修　士　論　文

データマイニング技術を用いた
ゲノムデータベースの要約手法に関する研究

北陸先端科学技術大学院大学
知識科学研究科知識システム基礎学専攻

土橋 潤也

2002 年 3 月

修　士　論　文

データマイニング技術を用いた
ゲノムデータベースの要約手法に関する研究

指導教官　　佐藤 賢二　助教授

北陸先端科学技術大学院大学
知識科学研究科知識システム基礎学専攻

050057　土橋 潤也

審査委員：　佐藤 賢二 助教授（主査）
小長谷 明彦 教授
中森 義輝 教授
本多 卓也 教授

2002 年２月

目　次

図　目　次

表　目　次

第　1　章
は じ め に

1.1　研究の背景と目的

遺伝子やタンパク質など、生物に関するデータの爆発的な増加に対応するために、京都大学化学研究所と東京大学医科学研究所ヒトゲノム解析センターは、ゲノムネット(Genome Net)と名づけた情報インフラストラクチャの構築を行ってきた。ゲノムネットのデータベースサービスは、世界中に存在する生物学･医学関連の多様な知識･情報･データを、各研究者のデスクトップで統合して利用できる環境を目指した情報サービスである。**[1,2]**

ゲノムネットには各種の検索や解析を行うサービスがあるが、代表的なものとしてはキーワード検索･ホモロジー検索･モチーフ検索･パスウェイ検索などがある。キーワード検索は単語の出現に関するインデックスの検索、ホモロジー検索は遺伝子やタンパク質の配列相同性に基づいた検索、モチーフ検索はパターン辞書を用いた配列のパターンマッチング、パスウェイ検索はグラフ探索というように、それぞれ処理の内容は異なるが、検索結果として多数の文書からなる集合(ゲノムデータベースではエントリ集合)が得られるという点では、Google などのサーチエンジンサービスと同様である。**[3]** しかし、大量のエントリが検索結果として得られた場合、それが何を意味するかを理解するのは容易ではない。各エントリが何であるかを示す１行程度の簡単な記述が併記されることもあり、そこからいくばくかの情報を読み取れる場合もあるが、エントリ内に記述された情報を良く読まなければ分からない事柄もあるため、結局はエントリを１つ１つ表示して人間が確かめなければならないことが多い。

この問題を抽象化して言えば、「大量の文書集合(エントリ集合)をいかに要約し、そ

の意味を把握しやすくするか」となる。要約に関する研究は主に自然言語処理の分野において行われているが、どういう状況においてどういうタスクを達成するかという観点から、いくつかのカテゴリが考えられる。例えば、タスクに関しては要約結果として重要な文が抽出できればよしとする重要文抽出タスクや、もとの文書に直接含まれない新しい要約文を合成するタスクがあるし、要約の対象が単一の文書なのか複数の文書なのかによっても必要な処理が異なってくる。**[4]**

本研究では、ゲノムデータベースに対する既存の検索や解析サービスを高度化し、ユーザの知識発見を支援する目的で、エントリ集合の要約を行う方法について研究を行う。エントリは文書に対応するから、これは複数文書を入力とする要約の生成に分類される。しかし、主に自然言語で書かれたテキストの場合と異なり、ゲノムデータベースのエントリはフィールドと呼ばれる幾つかの領域にわかれている。エントリには、コメントなど自然言語で書かれたフィールドだけでなく、キーワードなどの言葉を列挙したフィールド、数値情報を表の形にまとめたフィールド、化学構造式を記述したフィールド、配列情報だけからなるフィールド、さらにこれらが混在した複雑な形式のフィールドなどが含まれている。よって、ゲノムデータベースを対象として要約を行う場合、各フィールドに対してどのような要約処理が必要かを考えなければならない。どのような種類のフィールドに対しては、どのような要約処理が必要か、要約する価値のあるフィールドとそうでないフィールドの区別はどうすべきか、要約結果をフィールド毎に表示すべきか否かなど、独自に検討すべき点が存在する。

一方、どのような情報を要約として残すべきかについても、検討が必要である。自然言語データから重要文を抽出する処理を参考にすれば、与えられたエントリ集合の中で共通性が高い情報(多くのエントリに出現する情報)を採用することにより、良い要約が生成される可能性が高くなる。しかし、その情報がデータベース内の他のエントリにも頻繁に出現するようなら、それは与えられたエントリ集合に固有の情報ではなく、要約として提示する意味は薄いことになる。例えば、あるエントリ集合のあるフィールドが共通に DNA というキーワードを含んでいても、同じデータベース内の他の全てのエントリが同様に DNA というキーワードを含んでいるなら、これを表示するのは無駄であり、良い要約になっているとは言い難い。しかし、ゲノムデータベースを対象とする場合、与えられたエントリ集合(データベースの部分集合)だけを見て要約を行う必要はない。有限の全体集合が仮定できる状況では、与えられたエント

リ集合とその補集合を比較することにより、与えられたエントリ集合に特有な情報だけを要約として残すことが可能になる。これは、WWW のサーチエンジンサービスのように、計算機上で扱えないほど巨大な全体集合を仮定している場合と比較して、よりよい要約を行える可能性があることを示している。

このように、大量の情報の組み合わせから、与えられたエントリ集合に関してなるべく共通に出現し(共通性)、その補集合にはなるべく出現しない(特殊性)という条件を満たすものを探す処理を行う場合、データマイニングの分野で研究されている相関ルール発見の手法を利用することができる。本研究では、ゲノムデータベースに含まれている各種の情報のうち、言葉が列挙されている情報(キーワードなど)を対象に、相関ルール発見手法を部分的に使用することにより、与えられたエントリ集合を要約し、ユーザが理解しやすい形で表示することを目指す。

図 1.1　サーチエンジン(Google)における検索結果

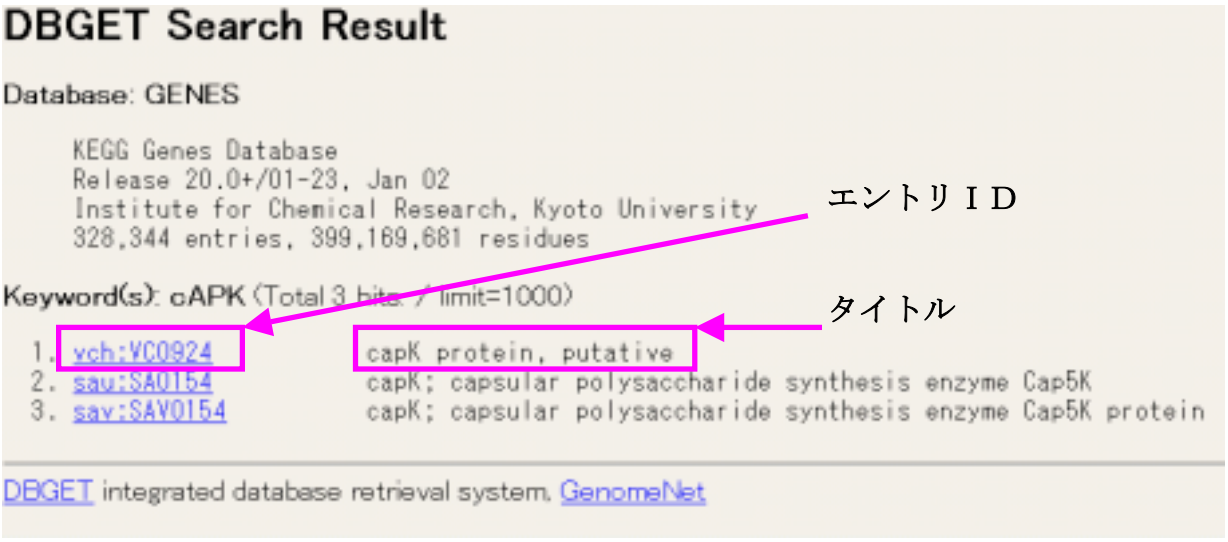


図 1.2　GenomeNet(GENES)における検索結果

1.2　本論文の構成

本論分は本章を含めて 7 章から構成される。第 2 章では、準備として自然言語処理とデータマイニングについて説明する。第 3 章では、ゲノムネットで提供されているゲノムデータベースの特徴と、各種検索サービスの特徴について説明する。第 4 章では、ゲノムデータベースの要約に必要なデータの加工などの処理について説明する。第 5 章では、前章までに述べてきたアルゴリズム、及びゲノムデータベースを用いて構築した要約システムについて、システムの構成や使用方法、及び要約結果の見方に関して説明する。第 6 章では、ゲノムネットで提供されている代表的な検索サービスを用いて、得られたエントリ集合の要約結果を確認し評価を行う。最後に第 7 章にて、研究のまとめ、及び今後の課題について述べる。

第　2　章
準備

本研究の目的である要約に関しては、自然言語処理の分野で盛んに研究が行われている。本章では、要約を行う際に参考とした研究例をまとめ、自然言語処理の分野で行われる要約に関して述べる。

また、本研究ではデータマイニング技術を用いて要約を行っている。このデータマイニングについて述べ、データマイニングのアルゴリズムについても触れる。

2.1　自然言語処理の分野における要約

要約とは、一般にあるひとまとまりのテキスト(文書)が表している意味や内容を非常に短いテキスト(文書)で簡潔に表現することを指す。人間があるテキストを要約する場合、そのテキストを読んで、その内容を理解し、それを再構成して直して簡潔な文章で表現する。人間が行う要約は、下記のような過程で行われるが、この過程を計算機でシミュレートすることは現在の自然言語処理技術ではほとんど不可能である。**[4,5]**

理解 ➔ 再構成 ➔ 文章生成

自然言語処理の分野における要約では、先に述べた人間による要約の過程で、内容を再構成して文章を生成することを諦める。これが具体的に意味するところは、「元の文章の中から重要な部分だけを残し、その他の部分を削除することによって要約を作成する」ということである。つまり、「文書の理解≒重要な部分の同定」と近似し、この処理だけで要約を作成する。重要な部分を同定する方法は大きく 2 つに分類できる。

1. 対象テキストの構造を積極的に利用する方法。
   ➔文書には、「起･承･転･結」や「序論･本論･結論」という構造を持っている。この結論部を抜粋して要約を行わせる。
2. テキストに含まれる各文の重要度を計算し、重要度の高い文だけを残す方法。
   ➔重要度の計算には以下のような特徴が利用される。
   (1) キーワードの出現回数
   (2) 特定の表現パターンの存在
   (3) 時制(現在，過去，未来)
   (4) 文のタイプ(主張，推測，事実.etc)
   (5) 全文との接続関係(理由，例示，逆説，対比，接続.etc)
   (6) 文章中の位置
   (7) 段落中の位置

また、要約の作成過程で、ユーザから要約の長さ(文字数や要約率)に関しての指定を受け、指定された範囲内で重要度の高い文を採用しなければならない。

$$要約率＝\frac{要約文の文字数}{原文の文字数}$$

以上のような点を考慮して、自然言語処理の分野で要約は行われている。要約の過程は、大きく次の 3 ステップに分けられる。**[6,7,8]**

１．原文の解釈
（形態素解析と構文解析結果の生成）
２．原文解析結果の要約を内部表現(意味解析)へ変換
（解析結果中の重要部分の抽出）
３．要約の内部表現(意味解析)から要約文を生成

現状の要約技術では、原文を読まず要約文だけで内容を理解できるレベルまで達して

いない。しかし、読むべき原文(本文)を探すための手段として有効であるといえる。

自然言語処理における要約は、単一のテキストを対象とした要約と、複数からなるテキストを対象とした要約が存在する。どちらの要約も、要約を行うための処理としては同一の処理を行う。要約したい部分を指定箇所として与え、先に述べた要約に関しての処理を行うことに変わりはない。

単一テキストの場合は、文書(テキスト)全部を指定箇所とすれば、文書全体に関する要約結果が得られる。要約を行うにあたり、文章中に共通に見られた重要な手がかりを探し出す。何度も頻出する形態素(単語)があれば、その形態素を重要語とみなして前後の文章を要約文として抽出する。

複数テキストの場合、複数からなる文書(テキスト)全部を指定箇所とすれば、複数の文書に書かれていた共通の文章が要約結果として得られる。基本的には、単一テキストに対する処理と同じであり、何度も頻出する形態素(単語)があれば、その形態素を重要語とみなして前後の文章を要約文として抽出する。その結果、複数の文書(テキスト)に共通に書かれていた文章を抽出することができるのである。

では、単一テキストと複数テキストを対象にした要約では何が異なるのであろうか。それは、複数テキストの場合、文書(テキスト)と文書(テキスト)の境が定まっている点にある。境があることで、それぞれの文書(テキスト)に共通して記述されていた情報や、一部の文書(テキスト)にしか記述されていなかった情報などが区別でき、要約結果に反映させることができる。

2.2　データマイニング

データマイニングとは、コンピュータを用いて、膨大な量のデータから、相関関係・パターンなどを高速に導き出すための技術や手法を指す。データマイニングでは、構造化されたデータベースから情報抽出を行うが、構造化されていないデータからの情報抽出はテキストマイニングといい、先に述べた自然言語処理分野で活発に行われている。**[9,10]**

近年、データマイニングが盛んに研究されるようになったのは、情報化社会において POS(販売時点情報管理)データシステムのデータや、顧客データなど多種多様なデータの蓄積が進み、それらを有効活用することが求められるようになったからである。このような大量のデータの間に成り立つ関係・規則・法則などを発見したいという要求からデータマイニングの研究は始まった。**[11]**

2.2.1　データマイニングの過程

データマイニングは以下の過程(**図 2.1**)で行われる。**[12,13]**

1. データの収集
   ユーザ(利用者)が得たいと考えている課題を明確に定義する。課題に見合ったデータをデータベースより収集する。
2. 前処理
   収集したデータからどのような項目を使用するか選択し、選択したデータの品質を確保するために、データの変換や欠損値処理を行う。またデータが大量に存在するときには、幾つかのデータをサンプリング(選択)して処理を行う。
   (1) データの変換
   データベースには様々なデータが存在する。各データの形式を変換して、コンピュータが処理しやすい形式に変換する。例えば成績表の場合、「優」・「良」・「可」・「不可」の 4 段階で表現される。このようなデータ表現では処理が困難な場合もある。そのような項目の表現を、数値形式に変換し

データ処理をしやすいようにする。

(2) 欠損値処理

選択したデータに欠けているものがあれば、何らかの方法でデータを補強、または排除し処理が行えるようにする。

(3) サンプリング

データが大量に存在する場合、幾つかのデータを選択して処理を行わせる。

3. 学習

変換されたデータからの規則や知識を発見する重要な過程。適切なデータマイニング手法を選択することを除けば、一般にこの処理は高速である。

(1) 学習モデルの決定･学習パラメータの決定

データからの規則パターンであるモデル(枠組み)を見つける。または、学習を行うために必用なパラメータを決定する。

(2) 学習の実行

データより定まったモデル･パラメータを用いて、マイニング処理を行う。

(3) 学習結果の検証

学習により一定の処理を経て得られたマイニング処理結果が、処理をしなかった場合のデータと比べ、どの程度の効果があるかを検証する。

4. 予測(評価)

学習で得られた規則に基づき、予測(評価)を行う。予想できるような結果が得られたのであれば、「取るに足らないルール」であったといえる。しかし、予想しなかったような結果が出た場合は、データの裏づけを検証し、その予測が偶然でないことを確かめ、新たな知識を得られたものであると考える。

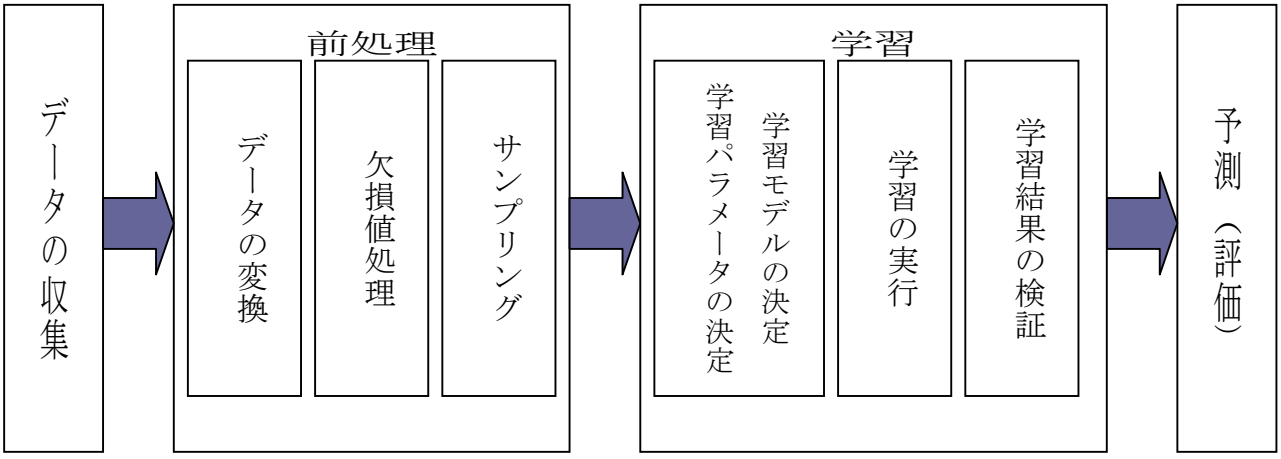


図 2.1　データマイニングの過程　（参考文献:**[12]**）

2.2.2　データマイニングの手法

データマイニングは、人工知能･統計手法･データベースの 3 分野の技術が融合して発展してきた。その結果、各分野の特徴に合わせて様々なアルゴリズムが開発されてきた。その中の代表的なデータマイニング手法について以下に示す。データの種類や予測したい分析結果などに合わせて手法を選択すれば、特徴ある予測が可能となる。**[14]**

- クラスター分析(クラスタリング)
  有効な分類軸がわかっていないデータをグループ化するための切り口を、自動的に探し出す手法。グループ化したデータの塊をクラスターと呼ぶ。データの分類軸を予め設定しないため、人間の予測や思い込みを超えた新たなデータ特性を見つける効果が期待できる。
- クラス分析(クラシフィケーション)
  過去に起きたデータの規則性を見いだして、それに基づき新規のデータを判別する手法。例えば、保険会社などで過去に自動車事故を起こした加入者のデータをこの手法で分析し、「事故を起こす可能性の高い人」に共通する特性を探って、新規加入者の保険料率の設定に役立てる。
- 相関ルール(アソシエーション)
  データ同士の相関関係の強弱を見る手法。膨大なデータを組み合わせて相関が高いパターンを検出する。米ウォールマーケット･ストアーズをはじめとする小売業者が「バスケット分析」と呼ぶ手法に応用されている。これは、POS(販売時点情報管理)情報をもとに、顧客が同時に購入する可能性の高い商品の組み合わせを発見するもので、「商品Ａを購入する可能性のある顧客は、42％の確立で商品Ｂも購入する」といった分析結果を返すため、店頭の陳列方法や接客時の提案活動に生かすことができる。
- 時系列分析
  相関関係分析に似ているが、時系列分析は同時にではなく、ある一定時間をおいて発生したデータの組み合わせや順序といった関係を調べる手法である。これにより、「商品 C を買った顧客は、次回の来店時に商品 D を購入するこ

とが多い」といった時系列を含む購読パターンをモデル化できる。こうしたモデルに基づいて、顧客のニーズに応じたタイムリーな販促活動ができるメリットがある。

- 決定木(デシジョン･ツリー)
  クラス分析を実施するための要素技術の一つで、特定の結果をもたらす要因拾い出すのに用いる。ある事象が起きるまでに、どのような分岐点があるかをシステムが自動的に探し出す。データを分類するルールを探し出すこうしたプロセスを木の幹や枝の形で表現することからこの名がある。
- ニューラル･ネットワーク
  過去のデータからパターンを学習して、それを基に新たなデータを分類したり、将来を予測する技術。人間の脳が学習するときの神経経路の働きを模している。クラス分析やクラスター分析のパターンに基づき、新たに得たデータが属するグループを判断するのに役立つ。

2.2.3 データマイニング手法の選択

データマイニングの手法は上記に述べた幾つかの手法が存在する。対象とするデータともに特徴があり、幾つか注意しなければならない点が挙げられる。**[11,13]**

- データマイニングの目的
  ユーザはデータマイニング結果により、どのような知識を得たいのか。
- データのタイプ
  対象とするデータベースには、どのようなタイプのデータが存在するか。例えば、過去からの膨大なデータが存在するか、細かな分類属性がありデータごとに様々な処理が要求されるか等。
- 適用の一般性
  どれだけ多くの問題解決やデータ型に適用することが出来るか。
- 適用の透過性
  データマイニングにより得られた結果が、わかりやすく理解しやすいか。このような結果は透過性があるといわれ、マイニングにより得られる結果は透

過性があるもの、無いものに分かれる。例えば決定木や相関ルールは明確なルールをもたらすため、わかりやすく透過性のある結果が得られる。しかし、ニューラル・ネットワークやクラスタリング分析では特定のモデルがなぜ得られたのか理解しにくく透過性の無い結果が得られる。

本研究ではデータマイニング手法として、透過性が優れており適用の一般性もある、相関ルールを用いた。それ以外にも、相関ルールでは分類属性を明確に定めなくてもよいので、分類属性を明確に定めることが出来ないデータを対象として、網羅的にマイニングが可能である。以上のような理由から、相関ルールを用いたデータマイニングを行う。

2.2.4 相関ルール抽出アルゴリズム

アイテム集合を $I=\{i_1, \cdots\cdots, i_m\}$、トランザクションデータベースを $D=\{t_1, \cdots\cdots, t_n\}$、$t_i \subseteq I$ とする。各要素 t_i をアイテム集合(itemset)と呼ぶ。そして個々のトランザクションにはユニークなトランザクション ID を割り当てる。その時に導出される相関ルールは次のように表現される。

$$X \Rightarrow Y\ ;\ X \subset I,\ \ Y \subset I,\ \ X \cap Y = \phi \qquad \text{式〔2.1〕}$$

相関ルールは支持度(Support)及び、確信度(Confidence)の 2 つのパラメータを持ち、これらの値は相関ルールの重要さを表す。相関ルールは、$X \Rightarrow Y$で表現される。相関ルールの $X \Rightarrow Y$の支持度は D 全体に対し、X, Yを共に含むトランザクションの割合($X \cup Y$)により定義され、確信度($X \Rightarrow Y$)は Xを含むトランザクションのうち、 Yも含むトランザクションの割合により定義される。書き換えると次のようになる。

$$\text{支持度} = \frac{\text{アイテム集合Xと Yを共に含むトランザクション数}}{\text{全トランザクション数}} \qquad \text{式〔2.2〕}$$

$$\text{確信度} = \frac{\text{アイテム集合Xと Yを共に含むトランザクション数}}{\text{アイテム集合Xを含むトランザクション数}} \qquad \text{式〔2.3〕}$$

相関ルールの抽出問題は、ユーザによって指定された最小支持度(minimum support)と最小確信度(minimum confidence)を満足する全てのルールを見つけることである。相関ルールは次の 2 ステップで抽出される。

Step1. 最小支持度を満足するアイテム集合を全て見つける。見つけたアイテム集合は頻出アイテム集合と呼ぶ。

Step2. **Step1** で求めた頻出アイテム集合から最小確信度を満たす相関ルールを導き出す。

相関ルール抽出処理のうち、**Step1** では基本的に、可能な全てのアイテム集合について支持度を調べる。このためアイテム数が多くなると組み合わせ論的にアイテム集合のバリエーションは増え、それに伴い計算が膨大になる。一方、**Step2** では、**Step1** で最小支持度を超えた頻出アイテム集合だけを対象に相関ルールの生成を行うため、**Step1** に比べると少ない計算量で処理できる。このため、相関ルールの研究では **Step1** の効率化が試みられている。apriori アルゴリズムは現在最も広く引用されている逐次アルゴリズムであり、本研究のシステムにもこれを用いている。

apriori アルゴリズムは 1994 年 IBM アルマデン研究所の R.Agrawal によって提案された。ここで、k 個のアイテムの組み合わせを k-itemset、長さ k の頻出アイテム集合を L_k、長さ k の候補アイテム集合を C_k とする。長さ k ＝1 の場合の処理は次のようになる。

1. トランザクションデータベース D から、長さ 1 の頻出アイテム集合 C_1 を作成する。
2. トランザクションデータベース D を探索し、支持度を求める。
3. 最小支持度を満足するものを取り出し、長さ 1 の頻出アイテム集合を L_1 とする。

長さ k ≧2 の場合の処理は次のようになる。

1. 長さ $k-1$ の頻出アイテム集合 L_{k-1} から、長さ k (k-itemset)の候補アイテム集合 C_k を作成する。
2. トランザクションデータベース D を探索し、支持度を求める。
3. 最小支持度を満足するものを取り出し、長さ k の頻出アイテム集合を L_k とする。

apriori アルゴリズムにおける候補アイテム集合の生成は、アイテム集合 L_{k-1} から次のアイテム集合 C_k を生成する。この候補アイテム集合を生成する時には最小確信度は考慮されない。**[11,13]**

- Join Step

insert into candidate k-itemset
select p.items$_1$, p.items$_2$, …, p.items$_{k-1}$, q.items$_{k-1}$
from large($k-1$)-itemset, p, large($k-1$)-itemset q
where p.items$_1$ = q.items$_1$, …, p.items$_{k-2}$ = q.items$_{k-2}$, p.items$_{k-1}$ < q.items$_{k-1}$;

- Prune Step

forall itemset c ∈ candidate k-itemset **do**
  forall ($k-1$)-subset s **of** c **do**
    if (s ∉ large($k-1$)-itemsets) **then**
      delete c **from** candidate k-itemset ;

図 2.2　apriori アルゴリズムにおける集合の導出（参考文献:**[13]**）

- Join Step

  L_{k-1}を自分自身と join して、C_kを生成。

- Prune Step :

  頻出でない任意の$(k-1)$アイテム集合は k アイテム集合の部分集合になれない。

  C_k　　: 大きさ k の候補アイテム集合

  L_k　　: 大きさ k の頻出アイテム集合

  L_i　　: {頻出アイテム群}

- Pseudo Code

for (k=1, $L_k \neq \phi$, k++) **do begin**

　C_{k+1}=L_kから作成された候補アイテム集合

　for each データベース中のトランザクション **do**

　　tに包含されている C_{k+1}中の全ての候補の係数を 1 増加する。

　　L_{k+1}=C_{k+1}中の最小支持度を満たす候補

　end ;

return $U_k \cdot L_k$;

図 2.3　apriori アルゴリズム (参考 Web:**[15]**)

- 頻出アイテム集合(最小支持度を超えるアイテム集合)を見つける。

1. 頻出アイテム集合の任意の部分集合は、再び頻出アイテム集合でなければならない。

   (例) もし、{A B}が頻出アイテム集合なら、{A}, {B}はともに頻出アイテム集合でなければならない。

2. 頻出アイテム集合を集合の大きさ順に 1 から k(k-itemset)まで順繰りに求める。

3. 頻出アイテム集合を用いて相関ルールを求める。

図 2.4　頻出アイテム集合のマイニング順序（参考 Web:**[15]**）

ここで、相関ルールを導出するアルゴリズムについて具体例を挙げて説明する。**表 2.1** の仮想データは、居酒屋にて顧客が頼んだメニューを仮定してある。

顧客番号 1　｛生ビール，枝豆，サラダ｝
顧客番号 2　｛冷酒，枝豆，冷奴｝
顧客番号 3　｛生ビール，冷酒，枝豆，冷奴｝
顧客番号 4　｛冷酒，冷奴｝

表 2.1 のような形式では、人間の感覚では非常に読み取りやすいが、コンピュータでは処理し難い。そこで、アイテム、及びトランザクションに関して、数値形式に変換して処理を行わせる**(表 2.2)**。この変換したデータをトランザクションデータという。データを前処理した上で、閾値についての設定を考えなければならない。今回は、最小支持度を 50%、最小確信度を 50%として実行するが、データマイニング実行により得たい知識に応じて閾値を変更する必要がある。実行例に関しては、**表 2.4** にまとめて示す。

表 2.1　仮想データ

DataBase D	
TID	*Items*
顧客番号 1	生ビール，枝豆，サラダ
顧客番号 2	冷酒，枝豆，冷奴
顧客番号 3	生ビール，冷酒，枝豆，冷奴
顧客番号 4	冷酒，冷奴

表 2.2　トランザクションデータベース

DataBase D	
TID	*Items*
1	1，3，4
2	2，3，5
3	1，2，3，5
4	2，5

表 2.3　頻出アイテム集合(閾値：最小支持度 50%，最小確信度 50%)

頻出アイテム集合	*支持度(Support)*
{1}	50%
{2}	75%
{3}	75%
{5}	75%
{1　3}	50%
{2　3}	50%
{2　5}	75%
{2　3　5}	50%

表 2.4　apriori による頻出アイテム集合の例（最小支持度 50%）

D	
TID	*Items*
1	1, 3, 4
2	2, 3, 5
3	1, 2, 3, 5
4	2, 5

C1	
ItemSet	*Support*
{1}	2
{2}	3
{3}	3
{4}	**1**
{5}	3

L1	
ItemSet	*Support*
{1}	2
{2}	2
{3}	3
{5}	3

C2
ItemSet
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

C2	
ItemSet	*Support*
{1 2}	**1**
{1 3}	2
{1 5}	**1**
{2 3}	2
{2 5}	3
{3 5}	2

L2	
ItemSet	*Support*
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C3
ItemSet
{1 3 5}
{2 3 5}

C3	
ItemSet	*Support*
{1 3 5}	**1**
{2 3 5}	2

L3	
ItemSet	*Support*
{2 3 5}	2

（補足）C_k　：大きさ k の候補アイテム集合

L_k　：大きさ k の頻出アイテム集合

L_i　：{頻出アイテム群}

Step1. 最小支持度(minimal support)を計算する。
与えられたデータのトランザクション数が 4、最小支持度が 50%であるから、最小支持度＝2 となる。

Step2. DataBase D より C_1 を生成する。
トランザクションデータベースを検索し、それぞれのアイテム集合がとトランザクションに含まれる回数を数え上げ、その結果を C_1 に格納する。

Step3. C_1 から L_1 を生成する。
C_1 のアイテム集合｛{1}，{2}，{3}，{4}，{5}｝から **Step1** で求めた最小支持度＝2 を満たすもののみ｛{1}，{2}，{3}，{5}｝を取り出し、これを頻出アイテム集合として L_1 に格納する。

Step4. L_1 から C_2 を生成する。
頻出アイテム集合 L_1＝{{1}，{2}，{3}，{5}}から 2 つのアイテムの組み合わせである、{{1 2}，{1 3}，{1 5}，{2 3}，{2 5}，{3 5}}を生成し、候補アイテム集合として C_2 に格納する。

Step5. C_2 から L_2 を生成する。
候補アイテム集合の C_2 の支持回数(それぞれのアイテムのトランザクションに含まれている回数)を、データベースを検索して数え上げ、長さ 2 の頻出アイテム集合{{1 3}，{2 3}，{2 5}，{3 5}}を求め L_2 に格納する。

Step6. L_2 から C_3 を生成する。
頻出アイテム集合 L_2＝{{1 3}，{2 3}，{2 5}，{3 5}}から 3 つのアイテムの組み合わせである、{{1 3 5}，{2 3 5}}を生成し、候補アイテム集合として C_3 に格納する。

Step7. C_3 から L_3 を生成する。
候補アイテム集合の C_3 の支持回数を、データベースを検索して数え上げ、長さ 3 の頻出アイテム集合{{2 3 5}}を求め L_3 に格納する。

Step8. L_3 から C_4 を生成する。
同様の手順で、L_3＝{2 3 5}から C_4 を生成しようとしても、長さ 4 の頻出アイテム集合は見つからないため、終了する。

以上により、最小支持度を満たす頻出アイテム集合は L_1，L_2，L_3 となる。

続いて、これらのアイテム集合の確信度を求める。最小支持度を満たしている頻出アイテム集合は、L_1，L_2，L_3であり、**表 2.3** より頻出アイテム集合は、

$$L_1 = \{\{1\}, \{2\}, \{3\}, \{5\}\}$$
$$L_2 = \{\{1\ 3\}, \{2\ 3\}, \{2\ 5\}, \{3\ 5\}\}$$
$$L_3 = \{\{2\ 3\ 5\}\}$$

と書き換えることができる。最小支持度(50%)と最小確信度(50%)を満たしているものとしては、合計 15 のルールが導出される。

1 <-	(50.0%/2, 50.0%)	生ビール<-
2 <-	(75.0%/3, 75.0%)	冷酒<-
3 <-	(75.0%/3, 75.0%)	枝豆<-
5 <-	(75.0%/3, 75.0%)	冷奴<-
1 <- 3	(50.0%/2, 66.7%)	生ビール<-枝豆
2 <- 3	(50.0%/2, 66.7%)	冷酒<-枝豆
2 <- 5	**(75.0%/3, 100.0%)**	**冷酒<-冷奴**
3 <- 1	(50.0%/2, 100.0%)	枝豆<-生ビール
3 <- 2	(50.0%/2, 66.7%)	枝豆<-冷酒
3 <- 5	(50.0%/2, 66.7%)	枝豆<-冷奴
5 <- 2	**(75.0%/3, 100.0%)**	**冷奴<-冷酒**
5 <- 3	(50.0%/2, 66.7%)	冷奴<-枝豆
2 <- 3 5	(50.0%/2, 100.0%)	冷酒<-枝豆 冷奴
3 <- 2 5	(50.0%/2, 66.7%)	枝豆<-冷酒 冷奴
5 <- 3 2	(50.0%/2, 100.0%)	冷奴<-枝豆 冷酒

以上のような閾値(最小支持度 50%，最小確信度 50%)の下で導き出されたマイニング結果としては、支持度 75.0%で、確信度 100%の「冷酒<-冷奴」と「冷奴<-冷酒」が注目できる。

- 居酒屋において、冷酒を注文する人は、冷奴を必ず注文する。
- 居酒屋において、冷奴を注文する人は、冷酒を必ず注文する。

という相関ルールが得られ、このようなマイニング結果より、新メニューとして、冷酒&冷奴のセット販売が考えられる。

2.2.5 相関ルール発見の問題点

相関ルール発見には、幾つかの問題点が挙げられる。ここでは、相関ルールを発見する過程において挙げられる問題点や、処理結果の分析における問題点を述べる。

- 相関ルールの組み合わせ問題

  相関ルールを求める、apriori アルゴリズムに関しては先に述べた。相関ルールの特徴は、与えられたデータのトランザクション数を数え、最小支持度を満たすデータを生成して、各頻出アイテム集合 L_kを生成していく。このときトランザクション数における、***k*-itemset(*k* 個のアイテムの組み合わせ)**により組み合わせは指数的に増大する。組み合わせを計算する計算量、及び計算時間は膨大なものになり、時として計算機の能力を超えてしまう。だが、指数関数的に増大する相関ルールの組み合わせは、最小支持度の閾値を上げれば問題は解消される。

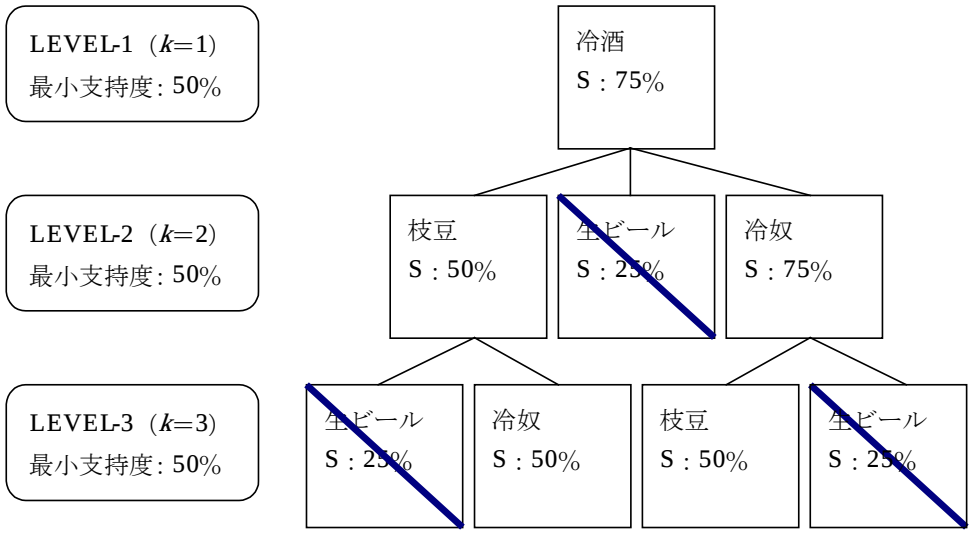
- 最小支持度における閾値の設定

  最小支持度に関しての閾値の設定は、***k*-itemset** に関わらず、一様に同値の最小支持度で計算を行う。しかし、実際にこの設定で計算を行えば、***k*-itemset** が増えていくにつれ、アイテムの出現頻度は少なくなる。これは、先に述べた相関ルールの組み合わせ問題にとっては良い解決策である。しかし、***k*-itemset** が増えていくにつれ相関を見逃してしまう可能性が生まれる。この解法を一様支持といい、全ての ***k*-itemset** において同じ最小支持度で計算を行うことをいう。これと反対に、***k*-itemset** が増えるにつれ、最小支持度を逓減させて計算を行う方法がある。この方法で行えば、***k*-itemset** が増えていくにつれ相関を見逃してしまうことは減少する。この解法を逓減支持という。

  一様支持と逓減支持における処理概要に関して、**表 2.2** のデータを用いて

図 2.5 に説明する。どちらの計算法ともに長所･短所があり、相関ルール発見において得たい情報に合わせどちらかの解法を選択する。**[15]**

1. 一様支持 ---全ての Level(*k*-itemset)で同一の最小支持度を用いた計算を行う。



2. 逓減支持 ---下位 Level(*k*-itemset の増加)につれ、最小支持度を逓減させて計算を行う。

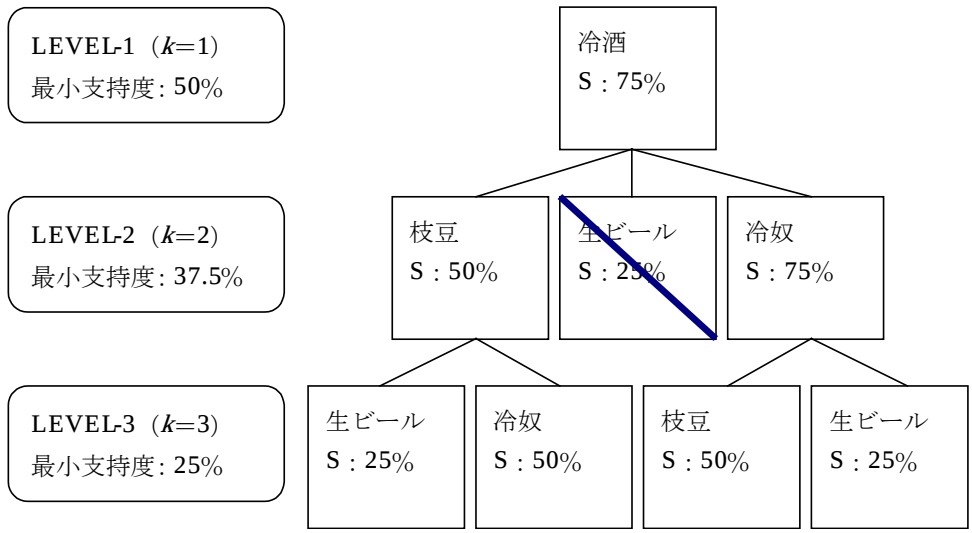


図 2.5　一様支持と逓減支持

- ルールの透過性

相関ルール発見では、最小支持度と最小確信度の 2 つのパラメータにより抽出されたルールを絞り込む。抽出されたデータの中には、稀に無意味なルールが含まれている場合がある。下記の 3 つの事例は、相関ルール発見でもたらされる 3 タイプを示す。**[11,13]**

- ✧ 木曜日にスーパーマーケットにおいてビールと紙オムツを一緒に買う人が多い。 ➔「有益なルール」
- ✧ 顧客は製品保証書を付けた大型家電製品を買う傾向がある。 ➔「取るに足りないルール」
- ✧ 日曜大工店の新規オープンでよく売れるものの一つがトイレットリングである。 ➔「説明不可能なルール」

「有益なルール」とは、質の高い有益な情報を示し、発見したルールを実行へと移すことが可能である。ビールと紙オムツの例では、木曜日の晩に子供のための紙オムツと父親用のビールを週末用に準備することを示している。この発見したルールより、店舗側では、紙オムツとビールを近くに並べて商品配置を行えば、2 つの商品の売り上げを更に伸ばすことも可能となる。更に発見したルールを発展させ、ビールの見える範囲に他のベビー用品を並べて商品配置を行うなどのマーケティングが考えられる。

「取るに足りないルール」とは、その業界の人なら誰でも既に知っている情報を意味し、マイニングの結果そのようなルールが発見されるのが当たり前である情報といえる。製品保証書と大型家電製品の例では、製品保証書と大型家電製品が別々に扱われることは少なく、一般的にペアで扱い販売されている。このようにして発見されたルールは、データ上では正しい発見であるが、使い道がない情報である。

「説明不可能なルール」とは、不可解であり理解不能な情報を示す。得られた情報結果の分析も難しく対応は取れない。新規オープン時にトイレットリングが売れる例では、新しい情報の発見と思えるかもしれない。しかし、この分析では、消費者行動や商品知識が欠けている為、誤解してしまっている。

開店セールの期間中、トイレットリングが他の商品に比べ安かった。または、少数店舗のみで起こった偶然の例外であったなどが考えられる。このようなルールは、原因がどのようなものであろうが、相関ルール発見に用いたデータからの追加分析でも確実に説明することは出来ない。「有益なルール」であるか「説明不可能なルール」であるかを見極めるには、事前に対象とするデータに対して、また関連のある分野に関する知識が必要である。

第　3　章
ゲノムネット(Genome Net)

遺伝子やタンパク質など、生物に関するデータの爆発的な増加に対応するために、京都大学化学研究所と東京大学医科学研究所ヒトゲノム解析センターは、ゲノムネット(Genome Net)と名づけた情報インフラストラクチャの構築を行ってきた。ゲノムネットのデータベースサービスは、世界中に存在する生物学･医学関連の多様な知識･情報･データを、各研究者のデスクトップで統合して利用できる環境を目指した情報サービスである。**[1,2]**

ゲノムネットのデータベースサービスは、インターネットを通じて誰にでも自由に利用できるサービスである。最も使いやすく、機能も豊富なのが WWW(Worle Wide Web)による利用である。インターネットによるサービスは、京都大学化学研究所バイオインフォマティクスセンターを中心に 3 ヵ所で行われている。

- 京都大学化学研究所バイオインフォマティクスセンター
  http://www.genome.ad.jp/
- 東京大学医学研究所ヒトゲノム解析センター
  http://www.tokyo-center.genome.ad.jp/
- 北陸先端科学技術大学院大学
  http://www.jaist.genome.ad.jp/

本章では、ゲノムネットで扱っているゲノムデータベースと、それらの検索や解析を支援する各種サービスについて説明する。

3.1　ゲノムネットのゲノムデータベース

ゲノムネットでは、生物学に関わりのある情報をデータベース化し、提供している。核酸配列データベース、アミノ酸配列データベース、タンパク質データベースなどは代表的なゲノムデータベースといえる。データベースの一部は、日々更新されており、最新の情報を手に入れることが可能となっている。**(表 3.1)**

表 3.1　ゲノムネットデータベースサービス（参考文献:**[1]**）

データベース	*内容*	*作成*	*日々更新*
GenBank	核酸塩基配列	米国 NCBI	*
EMBL	核酸塩基配列	欧州 EBI	*
SwissProt	タンパク質アミノ酸配列	ジュネーブ大学，欧州 EBI	*
PIR	タンパク質アミノ酸配列	ジョージタウン大学	
PRF	タンパク質アミノ酸配列	蛋白質研究奨励会	
PDB	タンパク質アミノ酸配列	ブルックヘブン国立研究所	*
PDBSTR	PDB アミノ酸配列	京都大学化学研究所	*
EPD	真核生物プロモータ	スイスがん研究所	
TRANSFAC	転写因子	ドイツバイオテクノロジー研究所	
PROSITE	タンパク質配列モチーフ	ジュネーブ大学	
LIGAND	酵素反応化合物	京都大学化学研究所	*
PATHWAY	KEGG パスウェイ	京都大学化学研究所	*
GENOME	KEGG ゲノムマップ	京都大学化学研究所	
GENES	KEGG 遺伝子カタログ	京都大学化学研究所	
OMIM	遺伝病	ジョンズホプキンス大学	*
PMD	変異タンパク質	蛋白工学研究所	
AAindex	アミノ酸指標	京都大学化学研究所	
LITDB	タンパク質関連文献	蛋白質研究奨励会	
Medline	医学･生物学文献	米国国立医学図書館	*
LinkDB	リンク情報	京都大学化学研究所	*

ゲノムネットで扱っているゲノムデータベースは、エントリと呼ぶ情報単位が集まった単純なファイル(フラットファイル)から成る。これは、エントリの集合によって、ゲノムデータベースが作られていることを意味する。**[1]**

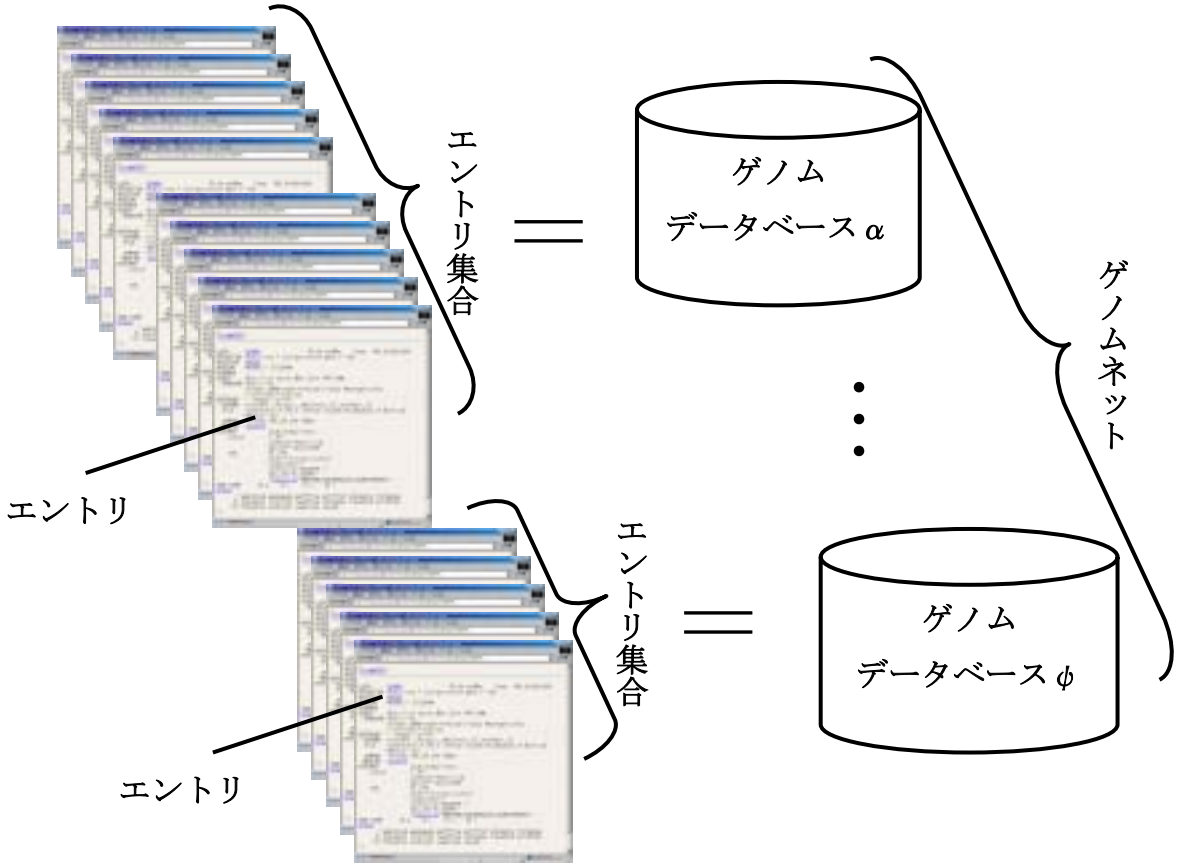


図 3.1　エントリとゲノムデータベースの関係

さらに、各エントリにはエントリ ID(またはアクセッション番号)と呼ばれる識別子が与えられている。従って、データベース名とエントリ ID の組を指定すれば、ゲノムネットに存在する数多くのデータベースを統合的に参照することが可能となる。**[1,15]**

ここで、エントリについて説明する。エントリはフィールドと呼ばれる領域にわかれている。フィールドはタイトルや、採取生物、塩基配列など各情報に合わせて作られており、何の情報が記載されているか判るようにフィールド名が付けられている。しかし、フィールド名に関してはデータベースごとに異なり、フィールド名の統合はされていないのが現状である。GenBank にでは、タイトルに関しての情報が記述さ

れているフィールドには「TITLE」というフィールド名が付けられており、採取生物に関する情報が記述されているフィールドには「ORGANISM」というフィールド名が付けられている。一方、EMBL では、タイトルに関しての情報が記述されているフィールドには「RT」というフィールド名が付けられており、採取生物に関する情報が記述されているフィールドには「OS」というフィールド名が付けられている。GenBank と EMBL は、どちらも核酸塩基配列に関するデータベースであるが、作成者が異なる。Genbank は米国 NCBI が作成し、EMBL は欧州 EBI が作成している。データベースの作成者が異なるので、このようなフィールド名に関しての統合はなされていないのである。また、他のデータベースに関しても同様の状況であるといえる。

フィールドに関しては、もう一つ特徴がある。フィールドには幾つかの記述形式が存在する。キーワードなどを列挙形式で記述した文字列からなるフィールド、自然言語で記述されたフィールド、数値情報を記述したフィールド、DNA やアミノ酸など配列情報だけからなるフィールドなど、様々な形式で記述されている。しかし、殆どのフィールドは、列挙形式で記述されている文字列からなるフィールドか、数値情報で記述されたフィールドか、配列情報で記述されたフィールドであり、自然言語形式で記述されているフィールドは少ない。

図 3.2, **図 3.3** は、GenBank と EMBL を対象として、エントリ ID には EBOMAY を組にしたときの、エントリについて例を挙げる。

GenBank：EBOMAY
EMBL：EBOMAY

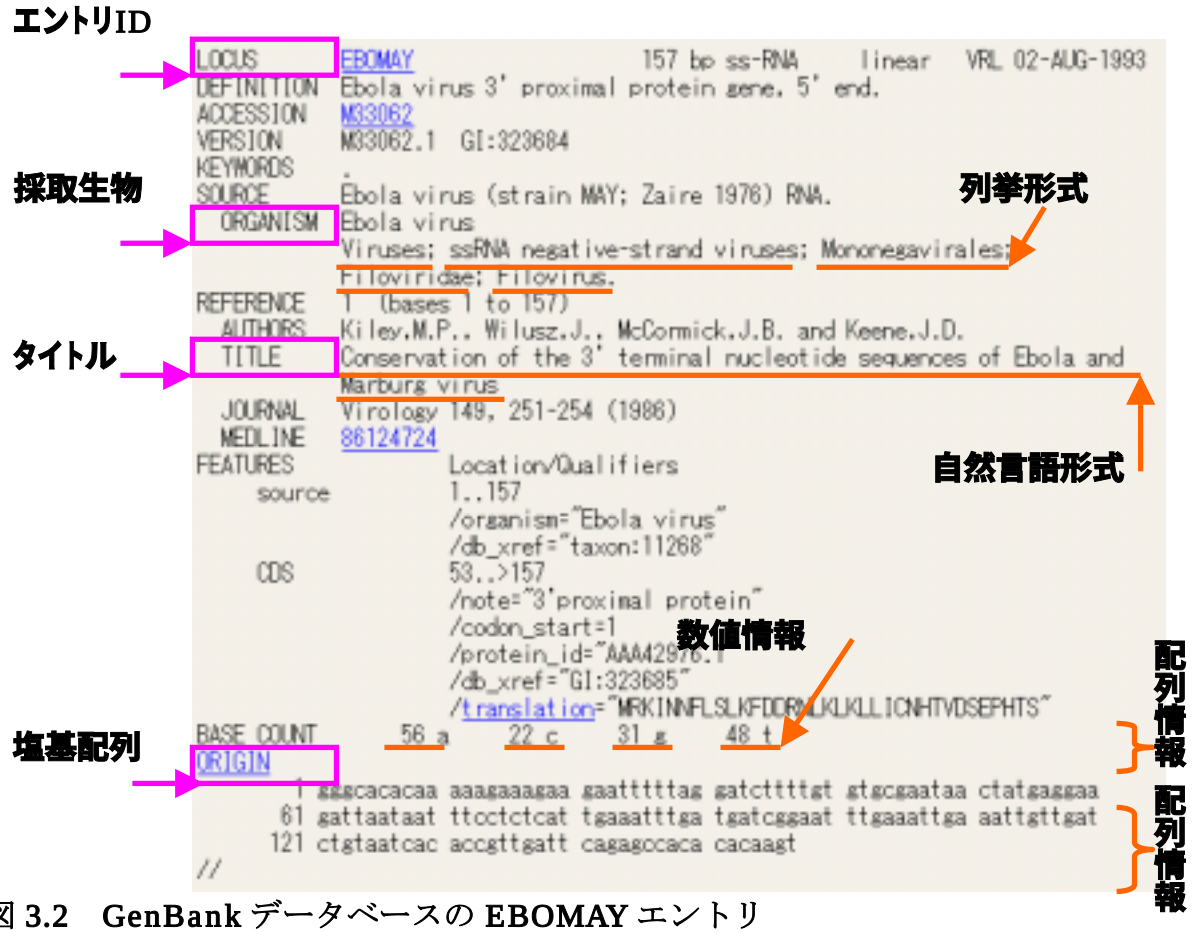


図 3.2　GenBank データベースの EBOMAY エントリ

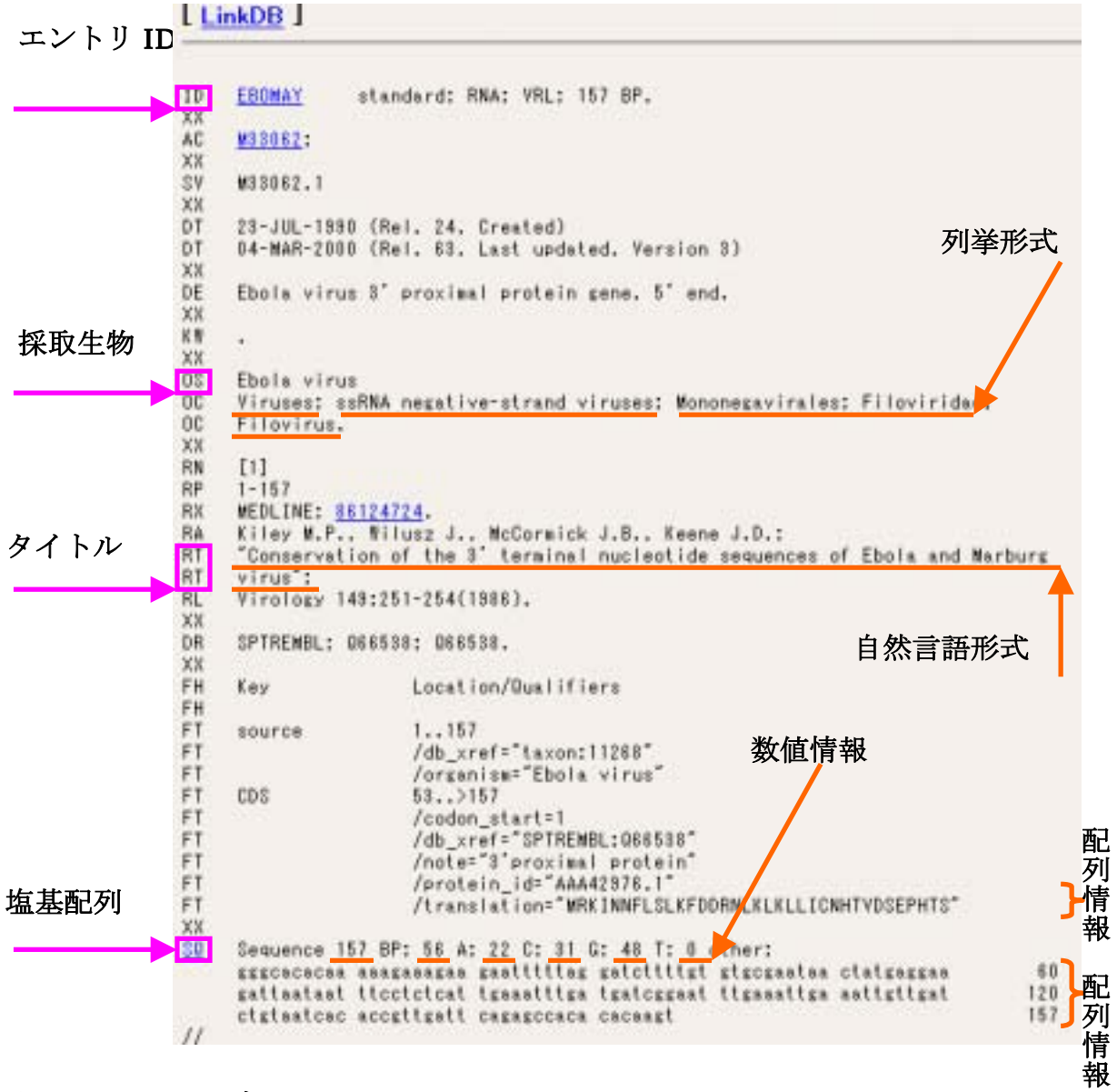


図 3.3　EMBL データベースの EBOMAY エントリ

3.2　ゲノムネットの各種検索サービス

ゲノムネットで提供されているデータベースの利用で最も基本となるのが、データベース検索システムである。データベース検索システムを利用することで、ユーザはエントリ情報を得ることができる。エントリ情報を得るためのデータベース検索システムの検索方法は、一般に

データベース名：エントリ ID

の組を与えれば、指定したデータベースの情報(エントリ)を得ることができる。また、ゲノムネットでは、異なるデータベースに関連するデータ(エントリ)があれば、相互参照することができるようになっている。例えば、文献データと文献に報告された配列データとの関連、塩基配列とそれを翻訳したアミノ酸配列関連をはじめ、異なるデータベースへのリンク情報を付加してデータベース化が行われている。この関連は、

データベース名 1：エントリ ID1 ➔ データベース名 2：エントリ ID2

の形で表現される。**[1]**

検索には、データベースとエントリ ID の組があれば、どのような情報も得られる中で、検索方法には、キーワード検索・ホモロジー検索・タンパク質モチーフ検索・パスウェイ検索の 4 つの種類が存在する。

キーワード検索とは、調べたいキーワードを入力し、エントリの取得を行う検索である。WWW サーチエンジンにおける検索と同様であり私たちには馴染みがある検索である。ホモロジー検索とは、配列データベースに対して、質問となる核酸配列やアミノ酸配列などを指定して相同な配列がデータベースに存在するか調べる方法である。タンパク質モチーフ検索も同様に、質問とアミノ酸配列を入力するが、入力した質問配列に対して配列情報の機能部位や特徴配列が記載してあるモチーフ辞書を参照し、パターンマッチングを行う。パスウェイ検索とは、細胞機能をタンパク質間相互作用ネットワークとして表現したパスウェイマップに対して、パスウェイマップからパスウェイマップへとグラフ探索を行う検索である。

ゲノムデータベースを検索するための検索サービスは、4 種類の検索方法によりエントリを得ることができる。代表的な検索サービスを**表 3.2** にまとめる。

- DBGET/LinkDB　　http://www.genome.ad.jp/dbget/
- KEGG　　http://www.genome.ad.jp/kegg/kegg2.html
- PATHWAY　　http://www.genome.ad.jp/kegg/kegg3.html
- BLAST　　http://blast.genome.jp/
- FASTA　　http://fasta.genome.jp/
- MOTIF　　http://motif.genome.jp/
- STAG　　http://stag.genome.ad.jp/

表 3.2　ゲノムネットで提供されている検索サービス

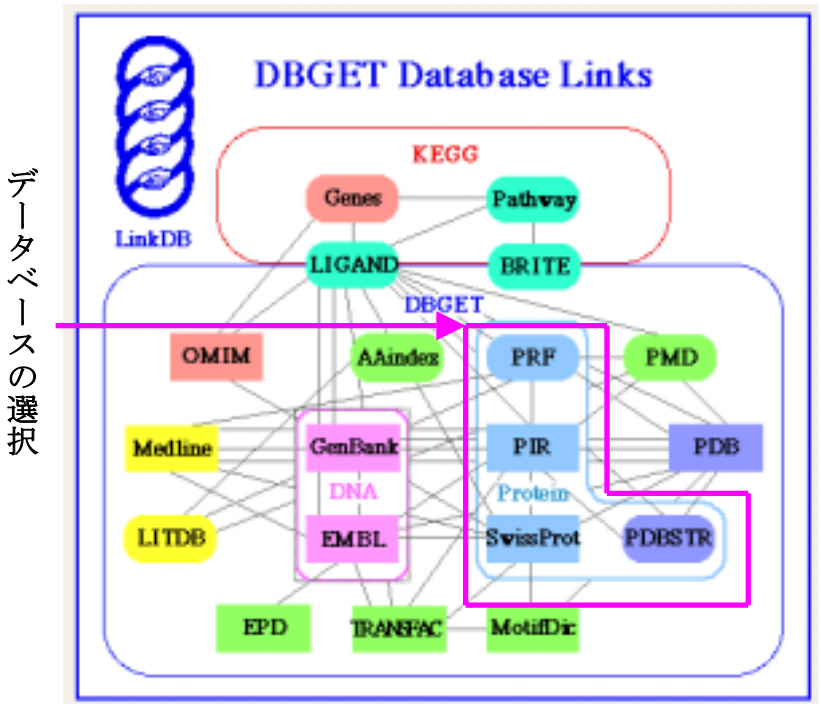
システム	*内容*	*作成者*
DBGET/LinkDB	統合データベース	京都大学化学研究所
KEGG	遺伝子･ゲノム百科辞典	京都大学化学研究所
PATHWAY	パスウェイ検索	京都大学化学研究所
BLAST	ホモロジー検索	NCBI
FASTA	ホモロジー検索	W.Pearson
MOTIF	タンパク質モチーフ検索	京都大学化学研究所
STAG	全文検索サービス	北陸先端科学技術大学院大学

3.2.1　DBGET/LinkDB による検索サービス

DBGET/LinkDB は、ゲノムネットで扱うゲノムデータベースを統合したデータベース検索サービスである。先の **3.1** でも述べたがゲノムネットで扱うゲノムデータベースは、それぞれ作成者が異なるゆえ、エントリの記述方式が異なる。また、データベースによっては日々更新が行われているため、データベースの記述を統合して扱うことは難しい。だが、データベースとエントリの組を指定すれば全てのデータベースを参照できる。また、エントリとエントリの関連付けも行われている。関連したエン

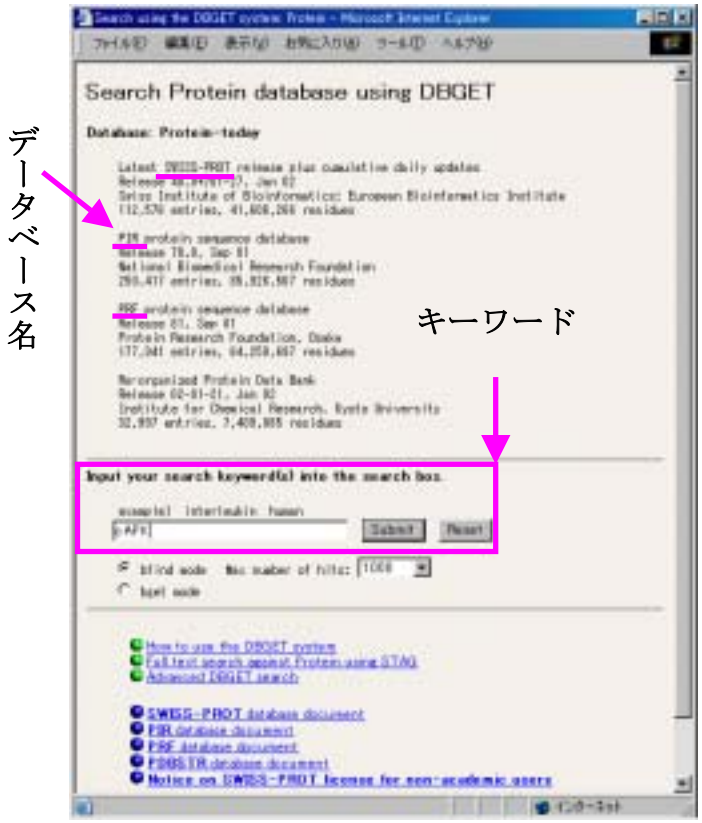
トリ間のリンク情報を利用して、データベースの統合を実現したのが、DBGET/LinkDB である。

DBGET リンク･ダイヤグラムでは、扱っているデータベース内容が類似しているデータベースに対して一度に検索が行えるようになっている。例えば、核酸塩基配列に関する情報をデータベース化してある、GenBank と EMBL に対しては、DNA という仮想データベースにより統合を実現している。アミノ酸配列に関する情報をデータベース化してある PRF と PIR と SwissProt と PDBSTR に対しては、Protein という仮想データベースにより統合してある。これを用いた検索例を以下に示す。



〔手順 1〕
アミノ酸配列データベースである PRF，PIR，SwissProt，PDBSTR に対してキーワード検索を行いたい場合、これらを統合してある仮想データベースの Protein を選択する。

図 3.4　DBGET リンク･ダイヤグラム



〔手順 2〕

検索したいキーワードを入力する。ここでは cAPK というキーワードを用いる。

〔手順 3〕

キーワード検索の結果、キーワードにマッチングするエントリ集合が表示される。

エントリ ID をクリックすることにより、特定のエントリを表示することができる。

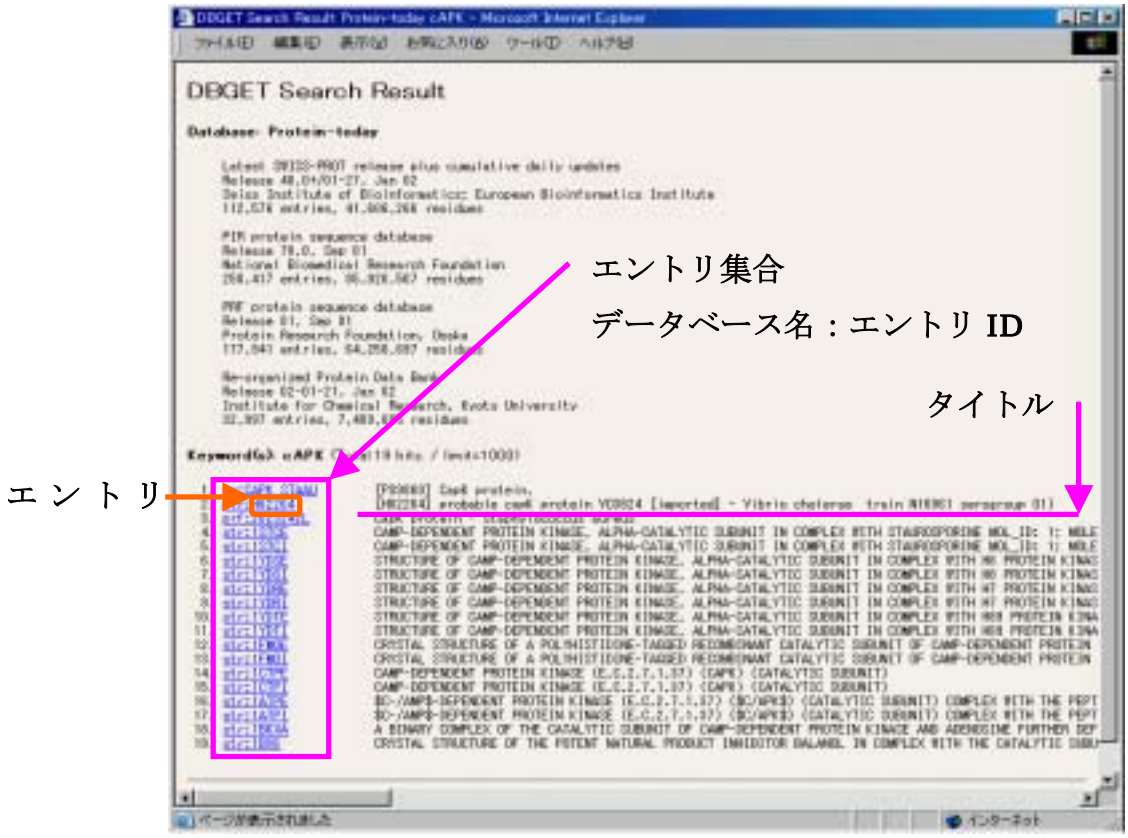


図 3.4　DBGET の検索画面と検索結果

3.2.2　STAG による検索

ゲノムネット上ゲノムデータベースに対する全文検索サービス STAG(Searching Texts in All over the Genomenet)は、1999 年 11 月から東京大学医科学研究所ヒトゲノム解析センターで開始した。このサービスは、フリーウェアの日本語全文検索システムである Namazu を用いることにより、検索の高速性と高いヒット率を実現している。**[16]** 現在では、北陸先端科学技術大学院大学でサービスを引継ぎ、更なるサービスの向上を目指して運用が行われている。

STAG で検索可能なゲノムデータベースは、AAindex, BRITE, COMPOUND, ENZYME, EMBL, EPD, GenBank, GENES, GENOME, LITDB, OMIM, PDB, PDBSTR, Pfam, PIR, PRF, PROSITE, PRINTS, PMD, RefSeq, SWISS-PROT TRANSFAC の 22 種類を対象としている。**(表 3.3)**

先の **3.2.1** で述べた、DBGET/LinkDB による検索では、ユーザはデータベースを一つ、もしくは内容が関連したデータベース群のみ選択してキーワード検索を行うことができた。この検索の方法では、ユーザが調べたいキーワードがどのゲノムデータベースに記述されているか予想がつかない時には、全てのデータベースを一つ一つ選択し、キーワード検索を行わなければならなかった。また、調べたいキーワードが複数のデータベース群 (内容が関連したデータベース以外も含めたデータベース群) に記述されている場合は、調べたいデータベースの数だけキーワード検索を行わなければならない。この検索では、ユーザは検索結果を得るのにかなりの人的な労力を使ってしまう。このような検索において、STAG の検索は威力を発揮する。ユーザが調べたいキーワードがどのゲノムデータベースに記述されているか不明なときは、STAG で検索を行えば、どのゲノムデータベースに記述されていたかが一度の検索で理解でき、情報が得られる。また複数のデータベースに記述されていることが既にわかっている場合にも、同じく一度のキーワード検索で情報が得られる。

また、STAG では検索対象となるデータベースを限定することもできる。以下では先に述べた DBGET と同様の例を用いて、STAG の検索手順を説明する。

表 3.3　STAG がサポートしているゲノムデータベース

検索区分	*ゲノムデータベース*
核酸配列(塩基配列)	GenBank，EMBL，EPD，RefSeq
アミノ酸配列	SWISS-PROT，PIR，PRF，PDBSTR
モチーフ検索	PROSITE，PRINTS
KEGG	COMPOUND，ENZYME，GENES，GENOME，BRITE
TEXT-LICH	OMIM，LITDB
その他	PDB，TRANSFAC，PMD，AAindex，Pfam

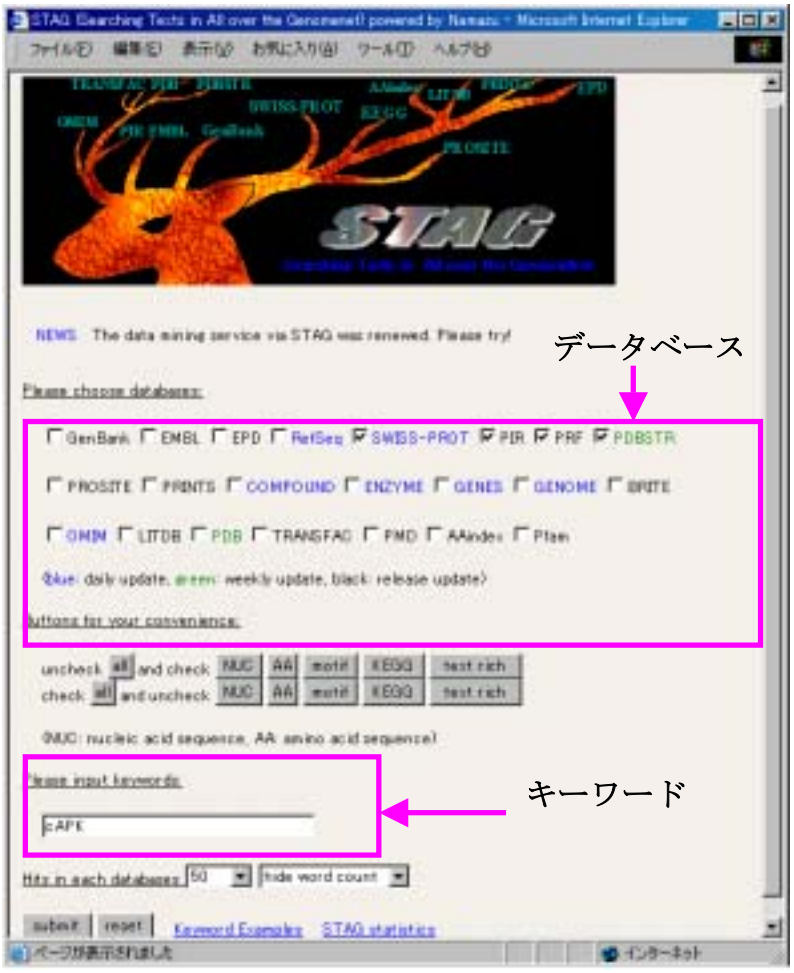


図 3.6　STAG の検索画面

〔手順 1〕
検索したいデータベースを選択する。

〔手順 2〕
検索したいキーワードを入力する。検索には、AND や OR を利用する論理型の記述によるキーワードの入力も可能。また、正規表現による曖昧な記述をしたキーワードの入力も可能。

〔手順 3〕
得られたエントリ集合から、確認したいエントリ ID を選択し、エントリを確認する。

図 3.7　STAG による検索結果

STAG による検索は、ゲノムデータベースを指定することから始まる。検索したいデータベースが不明な場合は、STAG で扱っている 22 種類のデータベース全てを選択すればよい。また、検索したいデータベースが既に決まっている場合は、核酸配列(塩基配列)、アミノ酸配列、モチーフ検索、KEGG、text-rich のいずれかを選ぶか、データベースを直接指定すればよい。続いて、検索したいキーワードの入力を行う。このとき、検索には AND や OR を利用する論理型の記述によるキーワードの入力も可能である。また、正規表現による曖昧な記述をしたキーワードの入力も可能となっている。

(例).　cAPK and human　←　論理表現(AND)によるキーワードの入力
　　human*　←　正規表現によるキーワードの入力

3.2.3　ホモロジー検索（BLAST）

ホモロジー検索とは、配列データベースに対して検索の対象となる核酸配列やアミノ酸配列などを指定して、相同な配列がデータベースに存在するかどうか検索を行う方法である。

現在、代表的なホモロジー検索プログラムとして BLAST と FASTA の２つが存在する。両者の違いは、配列内にギャップを入れるか入れないかの違いである。ギャップが入れば、配列に対して類似性が考慮され、質の高い情報が得られる。**[1]**

- BLAST
  NCBI(National Center for Biotechnology information)で開発されたプログラムで、ギャップを入れない部分配列のアライメント(一列に並べて整列させること)を複数含めて評価する手法を採用。また、閾値の設定を統計的計算により自動的に行う特徴をもっている。
- FASTA
  W.Pearson によって開発されたプログラムで、始めに文字の一致する領域について高速な検索を行う。最終的にはギャップを入れた完全なアライメントを行う方式を採用している。

速度については、一般に FASTA よりは BLAST のほうが高速である。BLAST はギャップを考慮しないので、ギャップが多く入った類似性を見落とす可能性があるが、多くの場合は十分な精度で検索が可能となっている。

ホモロジー検索を行うにあたり何点か注意しなければならない点が挙げられる。

1. 問い合わせ配列の作成
2. データベースの選択
3. プログラムの選択

注意すべき 1 点目は、問い合わせ配列の作成方法である。ホモロジー検索にかける配列のことを、問い合わせ配列と呼ぶ。問い合わせ配列は、以下のような形式(FASTA 形式)で記述する。**(図 3.8)**

```
＞名称 及び コメント
配列...
```

この問い合わせ配列の取得方法については、ゲノムネットでは、キーワード検索等を行いエントリを取得し、エントリの配列フィールドをクリックすれば得ることができる。

```
>gb:AF093232 [AF093232] Vibrio cholerae Vps70 (vps70) gene, partial cds.
tcacaacagtttatctcacaaaataaaccgtcagtggtggtgaaaggggcgaccagtggc
acaacgggtacacctctgactattttgcaagataggcattcggttattcgtgaacaagcc
tttgtcgcaagacagttggcttgggcgggatatcgtaaaggggataaacgagcgtggatc
cgtggcgatatggtggtgccattaagtt
```

```
>prf:2023242L CapK protein - Staphylococcus aureus
MLNYIYNHSPIIFQNLMVSIKGKIFMKQRYTKHYYEEIKRLRECNDLFELQNQRFEEFYN
YIKKNSEFYSEIIKKNNLSGKKITVANINQLPEITKDDIRKNVDKIITKKKNKLIKMGTG
GSTGKSMVFYTNAYDMSRKIAYLDYFKEQHGVYKGMKRVSVGGRKIVPIKQKKKVFWRYN
KPLNQLMISAYHADGENLKYYIKKLNKFQPETLDGYTTVIHRIARYILDNNIELSFTPIA
IFPNAETLTDLMRDDIEKAFNCPVRNQYASSEGAPFITENKEGELEINVATGVFECKQIH
GNIYELIVTGFYTTTTPLLRYKIGDSVELENELPVNYQQKDIKIKRIIGRNNDFLQSREK
GIVTNVNLSTAIRFVENDVIESQFVQNDIDNIIVYLVISNDADKNNIIKKLKYELKFRFG
TNTNFHFEFVNKIPSTPGGKKRFAINNIK
```

図 3.8　FASTA 形式で記述した核酸配列(上図)とタンパク質配列(下図)

注意すべき 2 点目はデータベースの選択についてである。問い合わせ配列が核酸配列であるか、タンパク質配列であるかにより対応しているデータベースは異なる。対

応しているデータベースに関しては、**表 3.4**，**表 3.5** に述べる。

表 3.4　BLAST がサポートしている核酸配列データベース（参考文献:**[1]**）

Nr-nt	GenBank，EMBL の最新リリース(EST も含む)、デイリー更新分を合わせたものから同一の配列を除いたもの。
genbank	GenBank 最新リリース(EST division を除いたもの)。
genbank-upd	GenBank のデイリー更新分。
EMBL	EMBL 最新リリース(EST division を除いたもの)。
EMBL-upd	EMBL のデイリー更新分。
Dbest	EST(Expressed Sequence Tag)配列を集めたデータベース。
EPD	真核生物プロモータ配列を集めたデータベース。

表 3.5　BLAST がサポートしているタンパク質配列データベース（参考文献:**[1]**）

Nr-aa	SWISS-PROT，PIR，PRF，GenBank のコード領域翻訳配列について、最新リリース、デイリー更新分を合わせたものから同一配列を除いたもの。
Swissprot	SWISS-PROT の最新リリース。
swissprot-upd	SWISS-PROT の最新リリース以降の更新分。
PIR	PIR の最新リリース。
PRF	PRF の最新リリース。
genpept	GenBank 最新リリースのコード領域翻訳配列。
genpept-upd	GenBank デイリー更新分のコード領域翻訳配列。
PDBSTR	PDB の最新リリースについて、鎖の単位で配列を収集したもの。
genes	KEGG 遺伝子カタログ。生物の遺伝子翻訳配列を集めたもの。

注意すべき 3 点目は、ホモロジー検索のプログラムの選択についてである。問い合わせ配列に対して、どのような情報で記述されたデータベースから配列比較を行うか選択する。（**表 3.6**）

核酸配列　　⇔ 核酸配列のデータベース

タンパク質配列 ⇔ タンパク質配列のデータベース

タンパク質配列 ⇔ 核酸データベースを翻訳したタンパク質配列

核酸配列を翻訳したタンパク質配列 ⇔ タンパク質配列のデータベース

表 3.6　BLAST のプログラムの選択

プログラム	*問い合わせ*	*データベース*	*備考*
blastp	タンパク質配列	タンパク質データベース	
blastn	核酸配列	核酸データベース	
blastx	核酸配列	タンパク質データベース	問い合わせ配列を翻訳して比較
tblastn	タンパク質配列	核酸データベース	データベースを翻訳しながら比較

核酸配列(塩基情報)が DNA の場合、アデニン(A)，チミン(T)，グアニン(G)，シトシン(C)から 3 つの塩基が 1 組となって 1 つのアミノ酸を指定する。更に、アミノ酸が複数の集合によって、タンパク質は構成されていることを知っていれば、BLAST のプログラム選択に対して理解できると思う。

以上のような点に注意して検索を行う結果、問い合わせ配列に類似した配列が含まれているエントリ集合を得ることができる。エントリ ID は、類実性の高い配列を順にして表示してある、類似性の評価指標として、P 値と呼ばれるものを利用する。P 値＝0 を示せば、配列相同性 100%であることを示し、値は小さければ小さいほど類似度が高いことを示す。P 値に関する目安は**表 3.7** にまとめる。

表 3.7　P 値の有意性（参考文献:**[1]**）

P 値	*有意性*
0.001 ≧ P	有意（ランダム配列ライブラリーでは滅多に出現しない）
0.001 ＜ P ≦ 0.1	微妙（ランダム配列ライブラリーでもたまに出現する）
0.1 ＜ P	有意でない（ランダム配列ライブラリーでも頻繁に出現する）

検索に関する問い合わせ配列に関しては、エントリ「prf：2023242L」のタンパク質配列に関する情報を記述してあるフィールドである「SEQUENCE」より取得した。類似しているタンパク質配列に関して検索を行うので、以下の点ような設定で検索を行った。

検索プログラム　：BLASTP
検索データベース：PRF
問い合わせ配列　：prf：2023242L

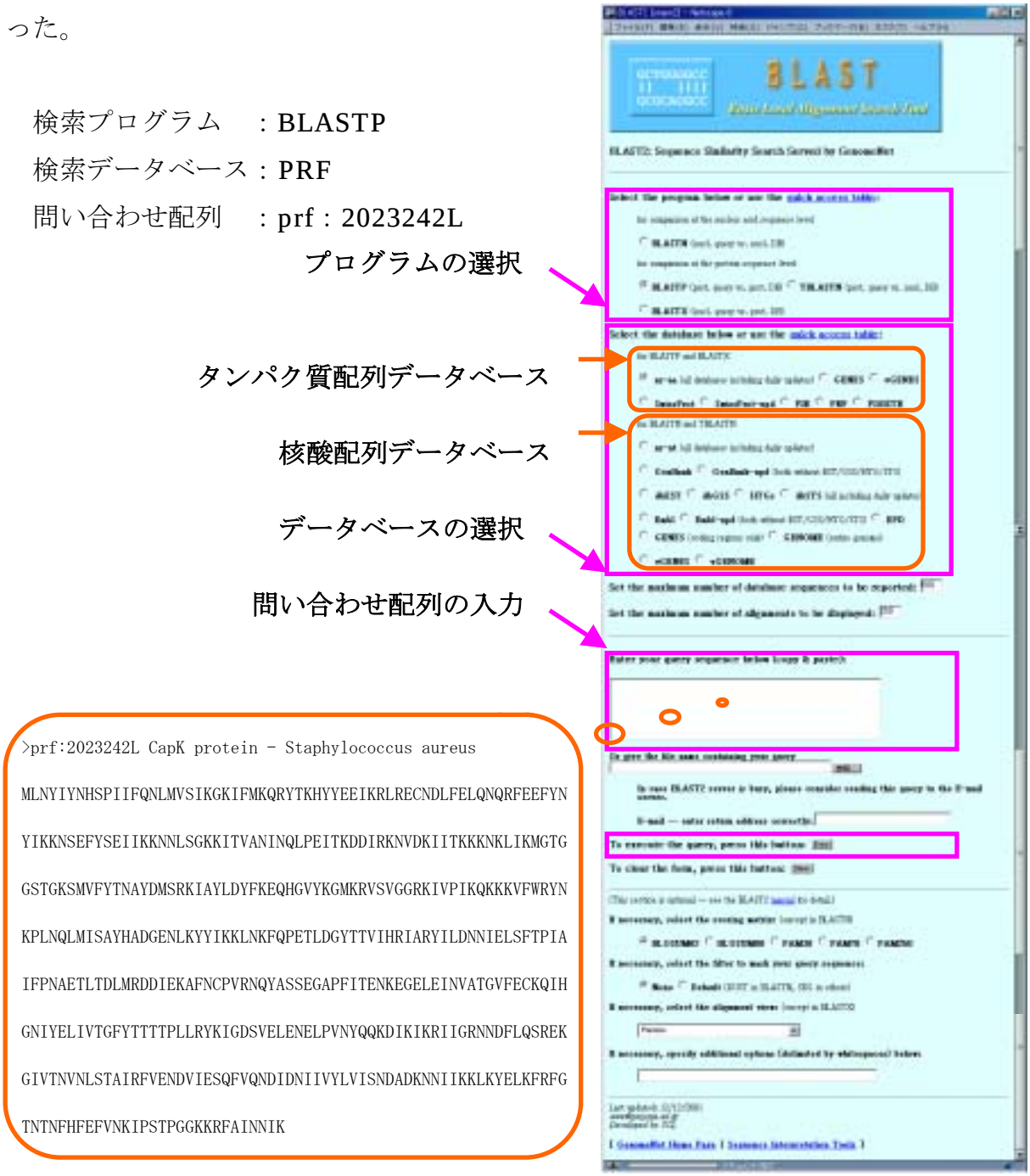


図 3.9　BLAST の検索画面.

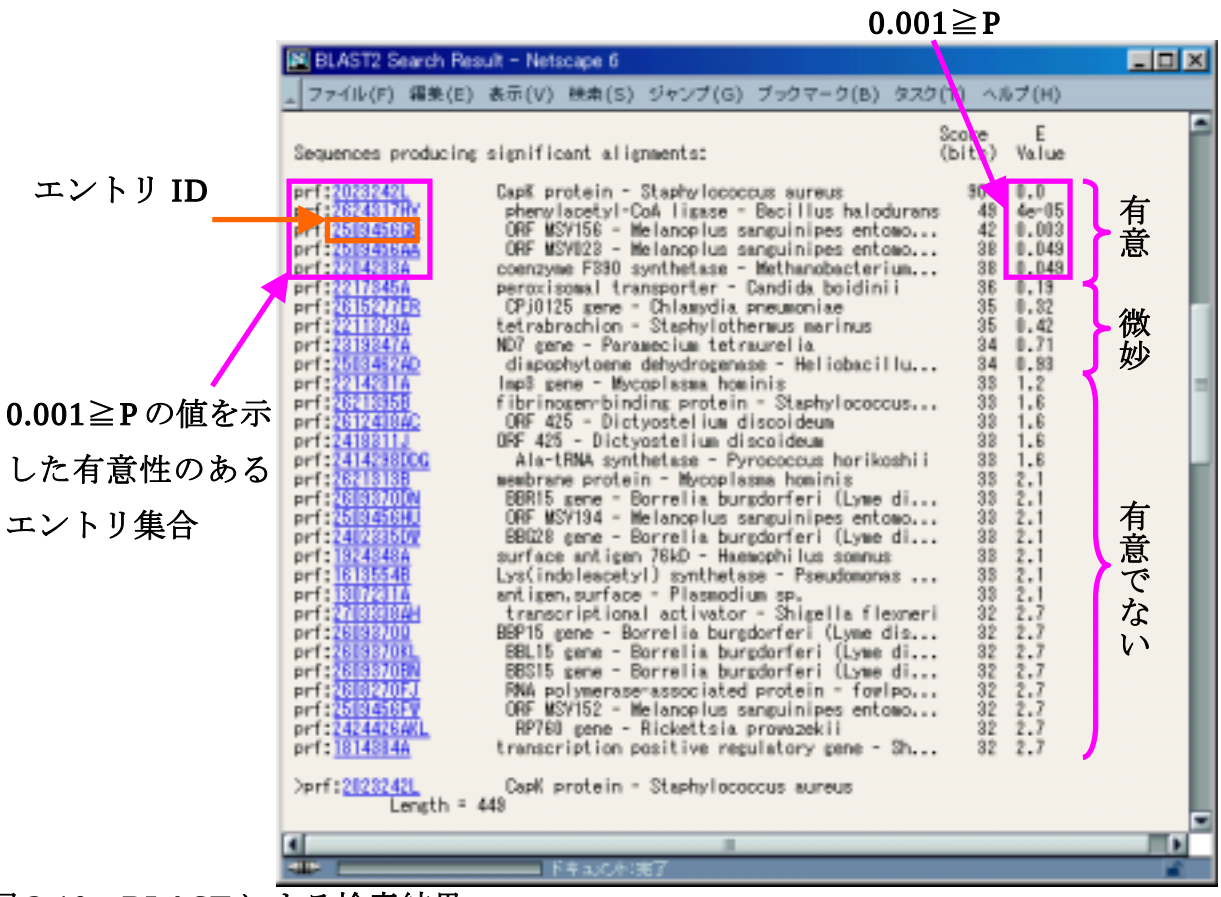


図 3.10　BLAST による検索結果

3.2.4　各種の検索における共通の特徴

ゲノムネットで提供されている各種の検索サービスの中から、DBGET/LinkDB によるキーワード検索、STAG によるキーワード検索、BLAST によるホモロジー検索などを例として挙げた。それぞれの検索サービスで提供されている処理の内容は異なるが、検索結果として多数の文書集合(エントリ集合)が得られる点で共通している。

ゲノムネットの各種検索により得られるエントリ集合は、ユーザが検索を行うにあたり入力した傾向(キーワード･配列情報.etc)と、密接に関係のある文書の集合である。だが、大量のエントリ集合が結果として得られた場合、その集合が何を意味するのか理解するのは容易ではない。エントリ ID と各エントリが何であるかを示す 1 行程度の簡単な記述の併記された情報を頼りにエントリを選択することも可能であるが、しかし、得られた集合からエントリを選択して確認するだけでは、検索結果を理解したとはいえない。検索結果である全てのエントリを確認し、ユーザが処理していたのでは負担が大きくなってしまう。ここでの処理とは、以下のようなことが想定できる。

- 得られたエントリ集合から目的のエントリを選択する。
- エントリ集合のどのような傾向かを確認する。
  (ⅰ). エントリ集合の各エントリがどのようなグループに分かれているか。
  (ⅱ). エントリ集合の各エントリは関連性があり、全く関連性のないエントリは混ざっていないか。

検索結果得られたエントリ集合から、どのような処理を求めるかはユーザにより異なる。だが、各種の検索において共通にいえることは、ユーザが検索結果の処理に、多大な時間を費やしている現状であることがいえる。

第　4　章
データマイニング技術を用いた
ゲノムデータベースの要約

4.1　本研究のアプローチ

ゲノムデータベースの各種検索サービスを用いた検索の結果、エントリ集合が得られることは前章で述べた。しかし、得られたエントリ集合が何を意味しているのか理解することは容易ではない。検索により得られたエントリ集合(文書集合)から、エントリ一つ一つを表示させ、人間が確かめなければ、エントリ集合の意味を理解することはできないのが現状である。この問題を抽象化していえば、「大量のエントリ集合(文書集合)をいかに要約し、その意味を把握するか」となる。上記の問題を解決するために、本研究では、ゲノムデータベースに対する既存の検索サービスを高度化し、ユーザの知識発見を支援する目的で、エントリ集合の要約を行う。

自然言語処理の分野において要約処理を行う場合、どのような状況においてどのようなタスク(課題)を達成するかについて検討する必要がある。この観点に基づき、ゲノムデータベースを対象にした要約について考えてみる。

まず、ゲノムデータベースの一つ一つのエントリは文書と見做せば、エントリ集合は複数テキストといえる。そこで、ゲノムデータベースから得られた検索結果(エントリ集合)を要約する際には、自然言語処理の分野の複数テキストの要約技術と同じ考え方で処理が行える。具体的な処理としては以下のものが挙げられる。

- 自然言語処理の分野の複数テキスト要約との関連
  - ➔ エントリとエントリの境が定まっており、一部のエントリのみに出現した情報、複数のエントリに出現した情報などエントリごとに情報のカウントが可能である。
- 自然言語処理の分野の重要文抽出との関連
  - ➔ 与えられたエントリ集合の中で共通性が高い情報(多くのエントリに出現する情報)に着目する。

この 2 点を参考に処理を行う。

さらに、ゲノムデータベースの特徴を活かした独自の方法も採用している。ゲノムデータベースでは、検索によりエントリ集合が得られるが、得られたエントリの補集合もデータベースに存在することから、以下の処理が考えられる。

- ゲノムデータベースならではの要約処理
  - ➔ 与えられたエントリの特異的に出現し、その補集合にはあまり出現しないような情報に着目する。

この方法を採用することで、エントリ集合の中で多く出現していた情報が、果たしてエントリ集合のみにしか見られない情報であったかを判断することが可能になる。その結果、与えられたエントリ集合に特有な情報だけを要約として残すことが可能になる。

また、エントリはフィールドと呼ばれる幾つかの領域にわかれている。フィールドごとに記述されている情報の内容が異なり、記述形式も異なる。そこで、各フィールドに対してどのような要約処理が必要かを考えなければならない。

- 各フィールドに対しての要約処理
  - ➔ フィールドの記述形式に合わせて、情報の抽出方法が異なる。
  - ➔ 各フィールドに記述されている情報の内容に合わせて、要約する価値があるか無いかを判断する。

上記で述べてきた処理を行うために、データマイニング手法が利用できる。なぜなら、与えられたエントリ集合に出現する情報とその組み合わせに関して、その集合に共有かつ特有なものを要約として残したいわけだが、これはデータ同士の相関関係を見ていることになる。そこで、各エントリの比較には、データマイニング技術の相関ルール発見手法を利用する。この手法を利用して、各エントリのそれぞれのフィールドにおける情報の相関関係を求めることができる。

- 相関ルール発見手法を利用したエントリ集合の要約
  - ➔ 各エントリに出現する情報の相関関係を求め、情報の出現回数、全体から支持される割合、情報の組み合わせに関して求める。その結果得られる、大量の情報の組み合わせから、与えられたエントリ集合に関してなるべく共通に出現し(共通性)、その補集合にはなるべく出現しない(特殊性)という条件を満たすものを探す。

さらに、要約結果の表示について考えなければならない。要約結果をどのように表示すれば、ユーザが把握しやすいかを考える。

- 要約結果の表示
  - ➔ 相関ルール発見手法を部分的に利用することで、重要情報の抽出ができる。そこで、要約結果として重要情報のリストを表示する。しかし、単に重要情報の表示だけでは、情報と情報の関係が理解し難い。そこで、情報間の関係を理解するために、グループ化の表示を行う必要ある。

4.2　要約の方針

本研究で行う、ゲノムデータベースを対象とした要約では、自然言語処理の分野における要約に関する研究を参考としている。例えば、ゲノムデータベースが、エントリという単位が集まった文書ファイルの集合で構成されていることから、複数テキストを対象とした要約処理の方法が利用できる。また、重要文抽出の処理方法に関しても、エントリ集合から共通な情報を抽出する際に利用できる。

だが、この処理を行う前には、まずデータベース中の全エントリから情報を規格化して切り出す必要がある。また、エントリはフィールドと呼ばれる幾つかの領域にわかれているため、フィールドごとに切り出しを行う必要がある。同じ研究室で構築している外延的オントロジー(キーワードなど、専門用語が列挙されているフィールドに着目して切り出しを行ったもの)を使用した。理由としては、各エントリとその情報の間の関係が、「存在する／存在しない」という単純な２値情報で表現できるため、相関ルール発見の枠組みで扱いやすいということが挙げられる。一方、数値や配列や構造など要約のしかたそのものを別途検討する必要があるものについては、本研究では扱わないことにした。

次に、与えられたエントリ集合について、どの専門用語を重要な情報として掲示すべきか検討する必要がある。本研究では、前節で述べた通り、以下の 2 つの観点から重要度を計算する。

- 共通性
  与えられたエントリ集合内になるべく共通に出現する。
- 特殊性
  与えられたエントリ集合内に出現し、その補集合にはなるべく出現しない。

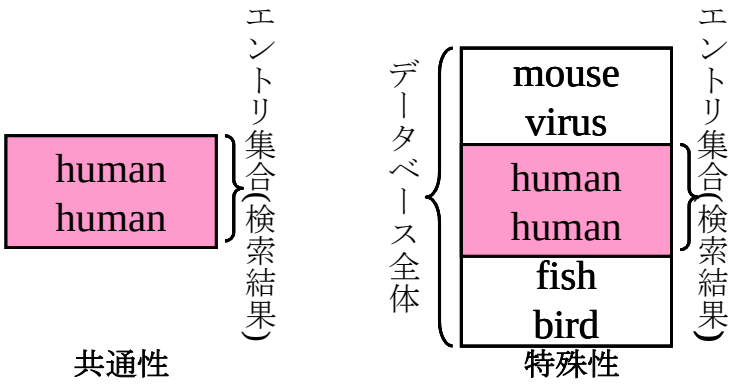


図 4.1　共通性と特殊性

これについては、具体的な例を用いて説明する。**図 4.1** では、検索により得られたエントリ集合の各エントリに記述されている情報が共通であるとはどのようなことか、特殊であるとはどのようなことかを示している。各行が 1 エントリを表し、行内の言葉はそのエントリがもつ情報を表している。

以下の例では、与えられたエントリ集合には、「human」しか見られない。「human」という情報がエントリ集合内に共通して出現している情報であるので、自然言語処理の分野における重要文抽出の処理では、重要語と位置付けることができる。さらに、そのエントリ集合内に出現した「human」という情報が、同じデータベース内の他のエントリにも出現しているかを見る。仮に、データベース全体に「human」という情報が見られているのであれば、仮に「human」という情報が出現しているのであれば、与えられたエントリ集合にも「human」という情報が出現するのは当たり前であり、重要語であるとは言い難い。しかし、**図 4.1** では、同じデータベース内の他のエントリには、「mouse」，「virus」，「fish」，「bird」という別の情報しか出現していない。よって、この例では、「human」という情報は、与えられたエントリ集合内に共通に存在し、かつ，その補集合には出現していない特殊な情報なので重要だといえる。

4.3　単なるデータマイニングとの違い

本研究では、指定されたエントリ集合に共通かつ特有な情報をユーザに分かりやすく提示することにより、エントリ集合の要約を行う。これに関連した研究として、主にゲノムデータベースのリンク情報を使用して、指定されたエントリ集合に共通かつ特有な情報だけを高速に提示するシステムを構築した例がある。**[11]** この例では、ゲノムデータベースが持つ巨大なリンク情報全体からデータマイニングを行うのではなく、指定された集合に属するエントリだけが持つ架空のアイテム（ターゲットアイテム）を導入し、それらのエントリのどれもが持たないアイテムを削除した上で、さらに同じ出現パターンを示すアイテム群を 1 つにまとめるチャンキング処理を行うことにより、指定されたエントリ集合に共通かつ特有なリンク情報をマイニングするのに十分な、小規模のデータテーブルを動的に切り出し、高速な相関ルール発見を実現していた。本研究においても、ユーザが着目しているエントリ集合(検索結果など)という情報を利用して、関連する情報だけを高速にマイニングするという点では、同じ手法を用いている。

しかしながら、布施田らの手法では、マイニングの結果をユーザに提示する際、相関ルールが持つ支持度と確信度だけを用いて重要度を決定していた。そうすると、例えば指定されたエントリ集合を表すターゲットアイテム x と、別のアイテム a との相関ルール(支持度 S，確信度 C)が得られた場合、S は全トランザクションで、x と a が両方成立するトランザクションの割合を表す。そのため、本研究で着目している共通性とは異なる。確信度 C の方は、相関ルールの方向が x⇒a なのか a⇒x なのかにより、異なってくる。さらに、相関ルール x⇒a の確信度は「x が成立するトランザクションにおいて a がどれだけ成立するか」を計算するため、x が成立しないようなトランザクション(つまり、指定されたエントリ集合の補集合)においてどのくらい a が成立しているかという情報を全く使用していない。逆に、相関ルール a⇒x の確信度は「a が成立するトランザクションにおいて x がどれだけ成立するか」を計算するため、x が成立するしないに関わらず、a が成立するトランザクションの情報しか見ていないことになる。これでは、本研究で着目している指標のうち、特殊性について計算できないことになってしまう。次節では、この問題を解決するために導入した計算式について説明する。

4.4　要約に用いた計算式

本研究では、要約を行うために共通性と特殊性を求める計算式を作成した。この計算式では、相関ルール発見手法の支持度を求める際の数式を参考にしている。相関ルールに関しては、先の **2.2.4** で既に述べているので、ここでは一部を省略した説明をする。

まず、以下の変数を定義する。

$x1$: 検索により得られたエントリ集合に含まれている
　　アイテムの組み合わせを含むトランザクション数
$x2$: 検索の対象となるデータベース内に含まれている
　　アイテムの組み合わせを含むトランザクション数
$y1$: 検索により得られたエントリ数
$y2$: 検索の対象となるデータベースのエントリ数

このとき、通常の相関ルール発見における支持度は以下のようになる。

$$Support = \frac{x1}{y2} \times 100 \qquad \text{式〔4.1〕}$$

しかし、本研究では着目する共通性は指定されたエントリ集合内で、アイテムが成立する割合なので、以下の式を用いた。

$$common = \frac{x1}{y1} \times 100 \qquad \text{式〔4.2〕}$$

一方、特殊性に関しては、指定されたエントリ集合内でアイテムが成立する割合と、全体集合(全トランザクション)でアイテムが成立する割合を比較することにより求められると考え、最初に以下の式を検討した。

$$special = \frac{\text{指定されたエントリ集合における共通性}}{\text{データベース全体における共通性}}$$
$$= \frac{x1 / y1}{x2 / y2}$$
$$= \frac{x1 \cdot y2}{y1 \cdot x2}$$

式〔4.3〕

式〔4.3〕では、全トランザクションを比べて、指定されたエントリ集合の割合が、$\left(y1 / y2\right)$が低い場合には、正しく機能する。しかし、そうでない場合は部分と全体の違いが小さくなるため、特殊性が曖昧になる。これを解消するため、以下の式〔4.4〕に変更し、違いを際立たせるようにした。

$$special = \frac{x1 \cdot (y2 - y1)}{y1 \cdot (x2 - x1)}$$

式〔4.4〕

さて、式〔4.2〕および式〔4.4〕で定義した２つの指標(*common*, *special*)は、どのような範囲を取り得るのだろうか。*common* に関しては、明らかに以下の数値が成立する。

$$100 \geqq common > 0$$

一方、*special* に関しては、極端な例でいえば、$x2 = x1$の場合(指定した部分集合だけにそのアイテムが出現する場合)は、分母である$x2 - x1$が 0 になり、式全体は無限大の値を持つ。逆に、$y2 = y1$の場合(指定した部分集合がデータベース全体の場合)は、分子である$y2 - y1$が 0 になり、式全体も 0 になる。よって、*special* に関しては以下の不等式が成立する。

$$+\infty \geqq special \geqq 0$$

一般にゲノムデータベースの要約では、共通であり特殊であるアイテムを重要アイテムとして抽出することが望ましい。このことから、下記の計算も行う。

$$common \times special \qquad \text{式〔4.5〕}$$

この指標については以下の不等式が成立する。

$$+\infty \geqq common \times special \geqq 0$$

以下では、共通性と特殊性に関しての理解を深めるため、幾つかの仮想データを用いて、共通性と特殊性に関する計算を説明する。(図 4.2) 各行は 1 エントリを表し、行内の文字列はそのエントリが持つアイテムを表す。また、着目するアイテムは「human」とする。各データの共通性，特殊性，共通性×特殊性について、計算結果と分析結果を挙げると以下のようになる。

共通性は、指定されたエントリ集合になるべく出現するアイテムであれば高い数値を示す。よって、指定されたエントリ集合のみを確認すればよい。(ⅰ)は、指定されたエントリ集合に「human」が全て出現している。(ⅱ)も、指定されたエントリ集合に「human」が出現しているが、「human」以外の情報である「virus」も出現している。(ⅲ)は(ⅰ)と同様に、指定されたエントリに「human」が全て出現している。(ⅳ)も、指定されたエントリ集合に「human」が全て出現している。そのような意味では、(ⅰ)と(ⅲ)と(ⅳ)に関しては、指定された集合に「human」は全て出現しているため、共通性は最高であるといえる。しかし、(ⅱ)に関しては、指定されたエントリ集合に、他のアイテムも出現しているため、共通性に関しては、(ⅰ)，(ⅱ)，(ⅲ)よりは低い結果となってしまう。

特殊性は、指定されたエントリ集合内に出現し、その補集合にはなるべく出現しないアイテムであればよい。よって、先ほどのアイテムが、補集合に出現していなければ高い値を示す。(ⅰ)は、補集合には「human」は出現していない。(ⅱ)も、補集合に「human」は出現していない。(ⅲ)は補集合に「human」が一つ出現している。(ⅳ)は、補集合全てに「human」が出現している。特殊性に関しては、他に「human」が出現していない(ⅰ)と(ⅱ)に関しては、最高であるといえる。また、(ⅲ)に関しては、補集合

にも「human」が出現しているため、特殊性は(ⅰ)と(ⅱ)に比べ下がってしまう。(ⅳ)に関しては、補集合全てに「human」が見られるため、特殊性があるとはいえず、特殊性は最低となってしまう。

最後に特殊性×共通性に関しては、先に求めた共通性と特殊性に関して両方を考慮した計算をしている。(ⅰ)は、「human」というアイテムが、指定されたエントリ集合にしか出現していない。共通性と特殊性は共に最高であることから、共通性×特殊性に関しても最高であることがいえる。(ⅱ)は、「human」というアイテムが、指定されたエントリにしか出現していないが、指定されたエントリ内に「virus」という他のアイテムも含まれている。しかし、補集合には「human」というアイテムは見られていないので特殊性は最高である。よって、特殊性に関しては、エントリ集合のみにしか見られないアイテムは正の無限大に発散するため、共通性×特殊性の検索をすると、先ほどの(ⅰ)と同様に、最高になる。(ⅲ)は、「human」というアイテムが、指定されたエントリにしか出現していないので、共通性は最高である。しかし、補集合にも「human」が一部出現しているため、特殊性は下がる。よって、共通性×特殊性の検索を行うと、(ⅰ)と(ⅱ)に比べ、低い値となる。

(ⅳ)は、「human」というアイテムが、指定されたエントリにしか出現していないので、共通性は最高である。しかし、補集合にも「human」が全てに出現しているため、特殊性は最低となる。よって、共通性×特殊性の検索を行うと、(ⅲ)よりも低い値となってしまう。

この結果をまとめると以下のようなことがいえる。

(ⅰ).「human」は指定したエントリ集合だけに出現している。

➔共通性は最高。特殊性は最高。共通性×特殊性は 1 位。

(ⅱ).「human」はエントリ集合だけに出現しているが、「human」以外の情報もエントリ集合に出現している。

➔共通性は中くらい。特殊性は最高。共通性×特殊性は 1 位。

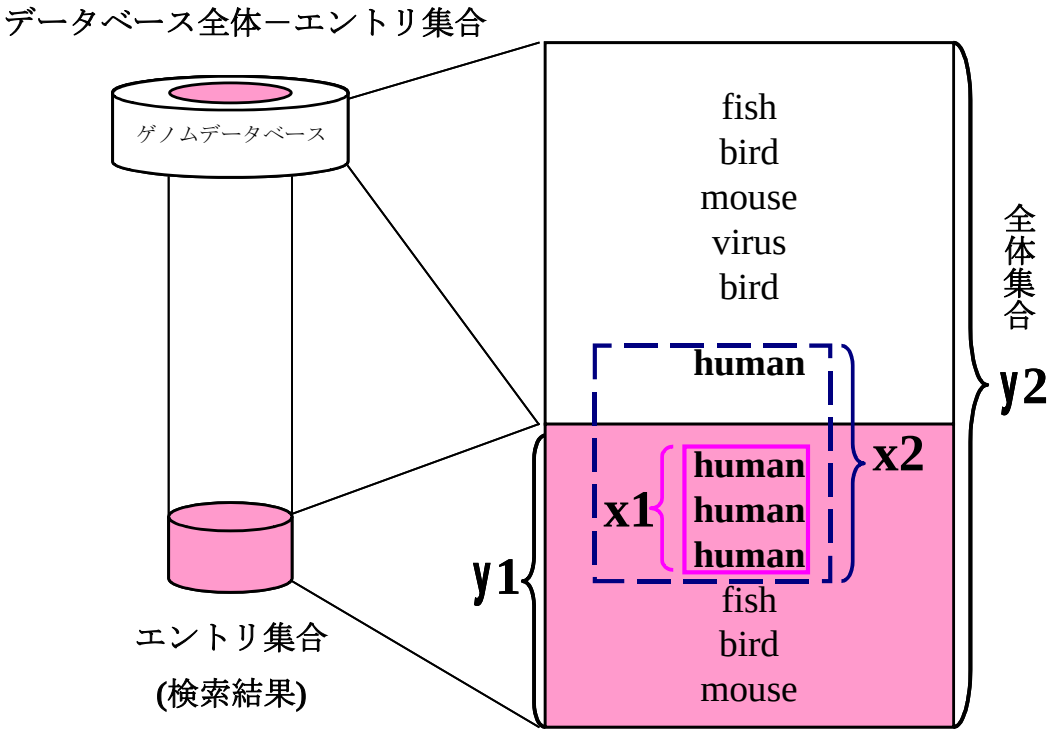
(ⅲ).「human」はエントリ集合に出現しているが、エントリ集合の補集合にも一部、「human」は出現している。

➔共通性は最高。特殊性は中くらい。共通性×特殊性は 2 位。

(ⅳ).「human」はデータベース全体に出現している。

➔共通性は最高。特殊性は最低。共通性×特殊性は 3 位。

以上のような結果から、エントリ集合の中で共通性の高いアイテムであり、データベースの他のエントリ集合内ではなるべく出現しないアイテムを要約結果として残すべきであることがいえる。



$x1$: 検索により得られたエントリ集合に含まれている
アイテムの組み合わせを含むトランザクション数

$x2$: 検索の対象となるデータベース内に含まれている
アイテムの組み合わせを含むトランザクション数

$y1$: 検索により得られたエントリ数

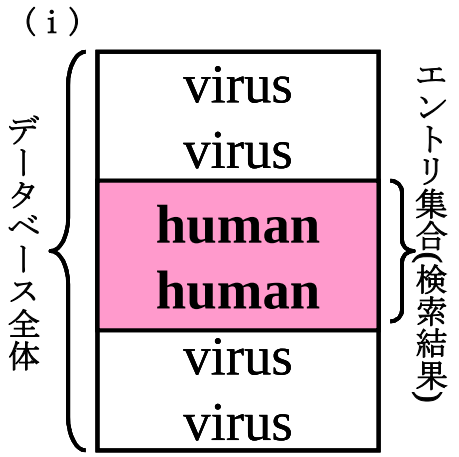
$y2$: 検索の対象となるデータベースのエントリ数

共通性　：$common = \frac{x1}{y1} \times 100$

特殊性　：$special = \frac{x1 \cdot (y2 - y1)}{y1 \cdot (x2 - x1)}$

共通性×特殊性：$common \times special$

図 4.2　共通性･特殊性･共通性×特殊性



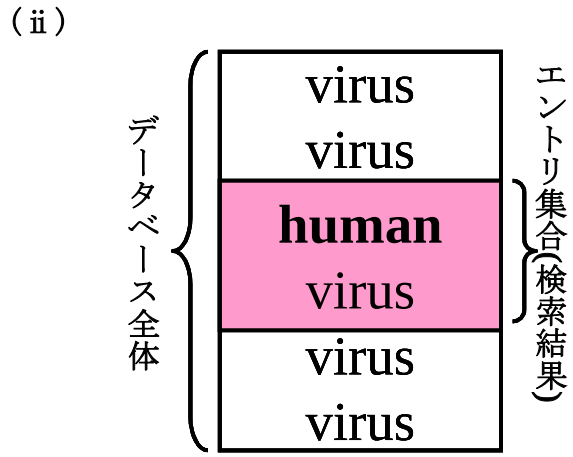
Human <-

X1＝2, X2＝2, Y1＝2, Y2＝6

Common＝100　共通性最高

Special＝+∞　特殊性最高

Common×*Special*＝＋∞



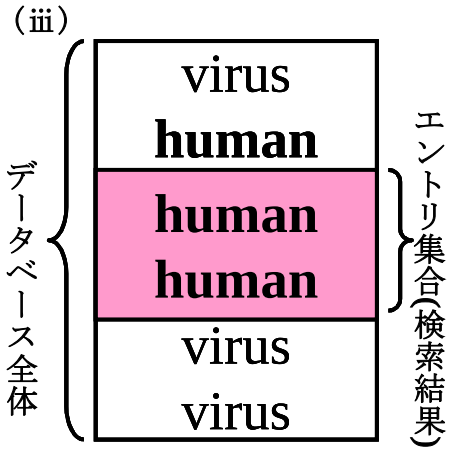
Human <-

X1＝1, X2＝1, Y1＝2, Y2＝6

Common＝50　共通性中くらい

Special＝+∞　特殊性最高

Common×*Special*＝＋∞



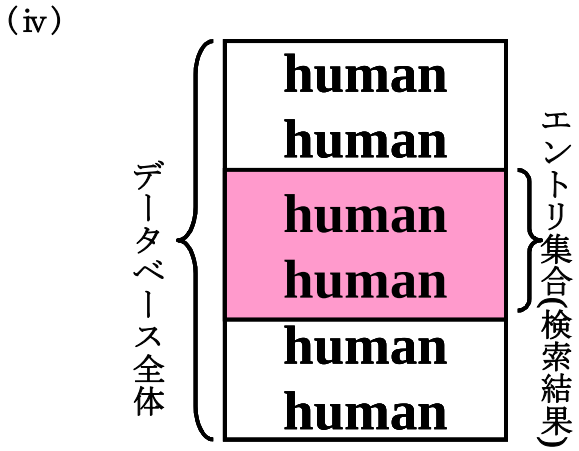
Human <-

X1＝2, X2＝3, Y1＝2, Y2＝6

Common＝100　共通性最高

Special＝4　特殊性中くらい

Common×*Special*＝400



Human <-

X1＝2, X2＝6, Y1＝2, Y2＝6

Common＝100　共通性最高

Special＝1　特殊性最低

Common×*Special*＝100

図 4.3　共通性と特殊性に関する指標

第　5　章
システムの構築

前章までに、要約システムの処理内容や計算式について述べてきた。本章では、これらを実装した際の処理手順、要約システムの構成、提供するサービスについて述べ、システムの利用方法および要約結果の見方について説明する。

5.1　システムの構成

本システムは、ユーザが入力したエントリ集合に出現する各アイテムに対して、相関ルール発見や各種計算を行うことで、入力されたエントリ集合を要約し、ユーザの知識発見を支援する。システムは Web 上で動作し、ブラウザがあれば実行できる。ユーザからの入力および結果の出力は全て Web 経由で行われる。ユーザが興味を持っているエントリ集合と、オプションの選択を入力すると、それらのパラメータは Web サーバ経由で CGI(Common Gateway Interface)に受け渡される。CGI は受け渡されたパラメータに従って、データの加工、データマイニング、各種計算を行い、その結果を Web サーバ経由でユーザ側のブラウザに表示する。

本システムは、データの準備部分、入力部分、処理部分、および出力部分にわけることができる（**図 5.1**）。本研究ではデータマイニング処理を行うために、Magdeburg 大学の Christian Borgelt がフリーウェアとして公開している apriori プログラムを使用している。**[13]**

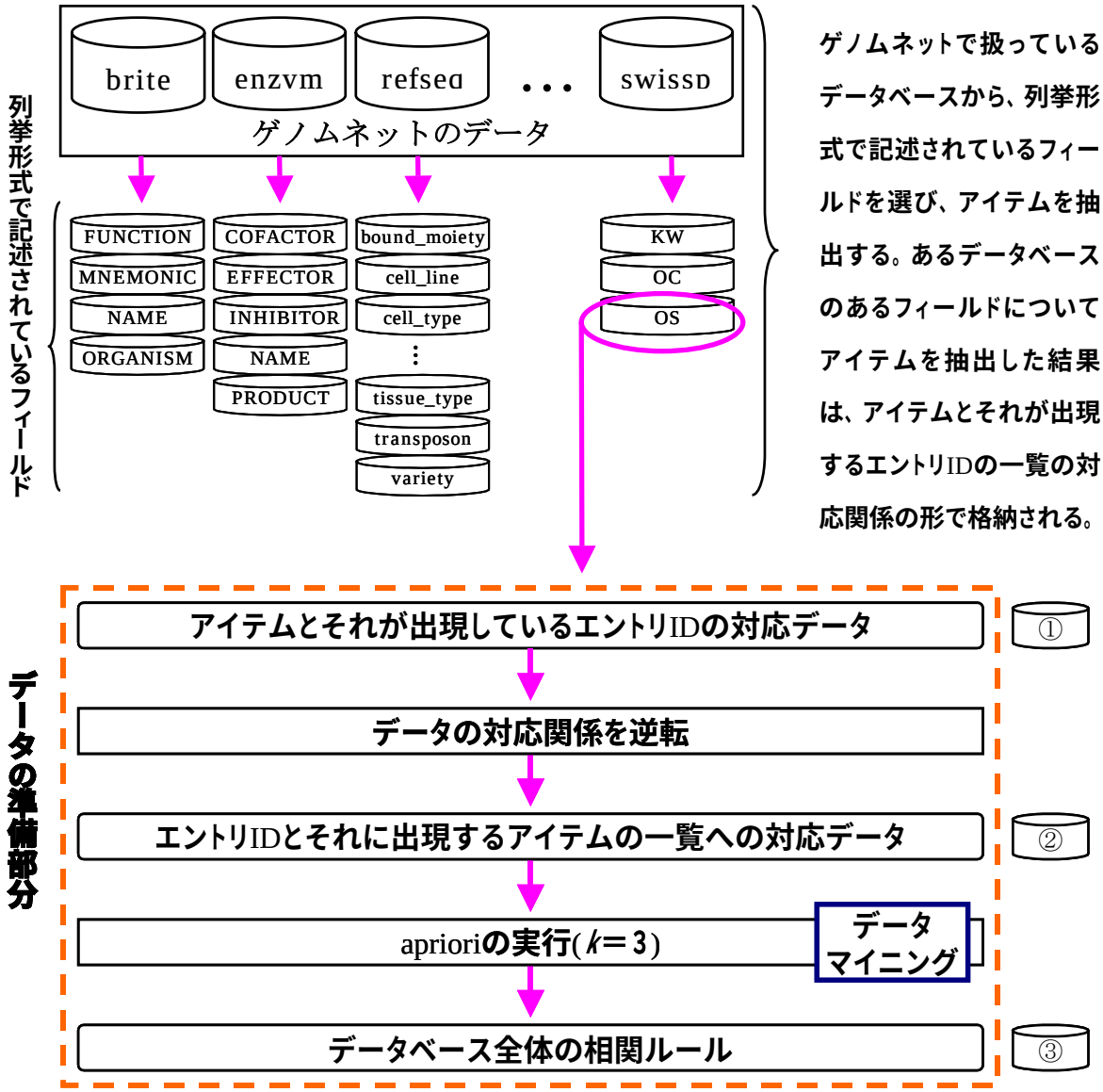


図 5.1　本システムの概要 (データの準備部分)

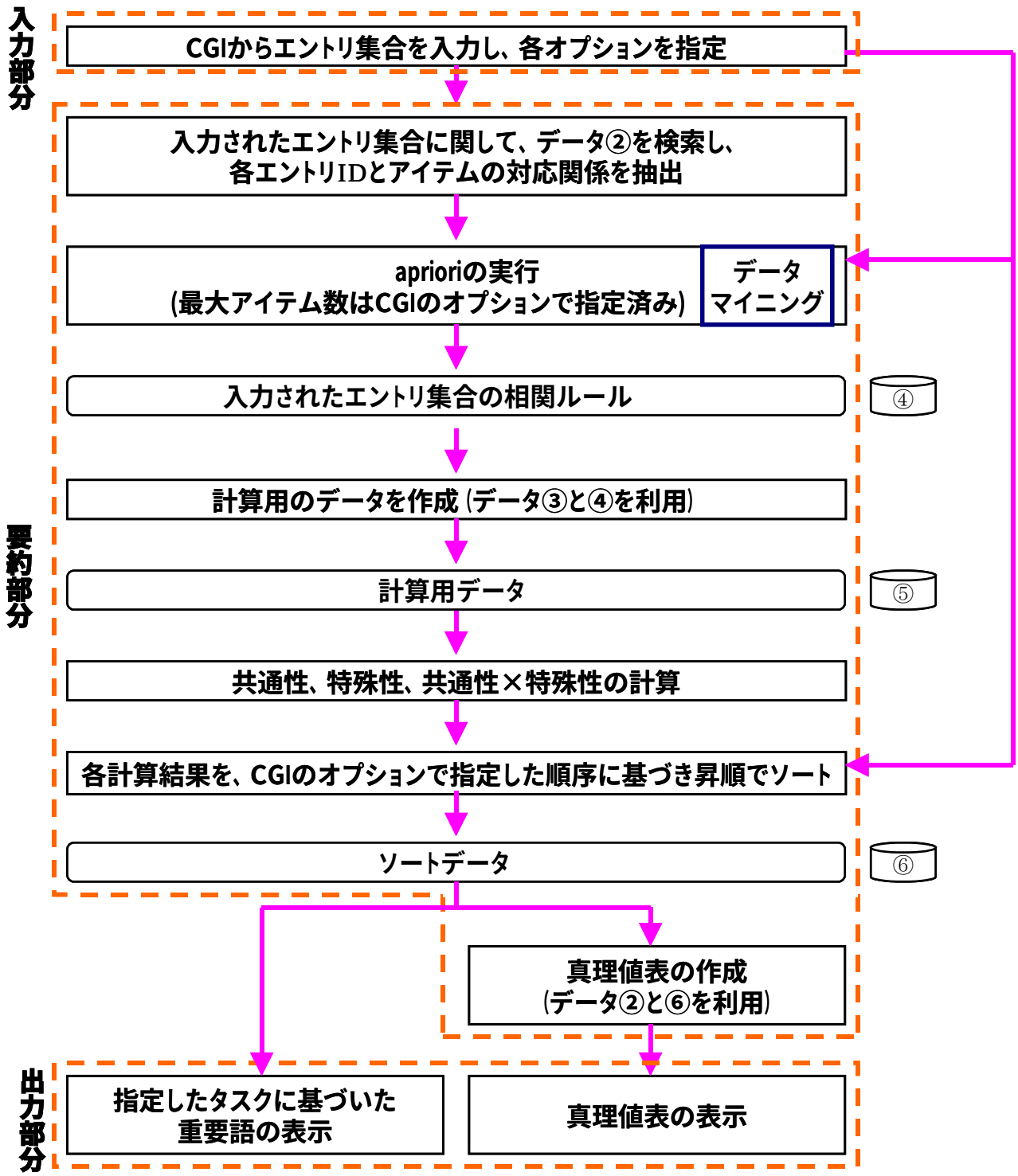


図 5.1 の続き　本システムの概要（入力部分，要約部分，出力部分）

5.1.1　データの準備部分

データの準備部分では、以下の手順で処理を行う。

Step1.　ゲノムネットで扱っているデータベースから、列挙形式で記述されているフィールドを選び、アイテムを抽出する。あるデータベースのあるフィールドについてアイテム抽出を行った結果は、以下のようにアイテムとそれが出現するエントリ ID の一覧との対応関係の形で格納される。

```
アイテム 1　エントリ ID1 エントリ ID2 エントリ ID3 エントリ ID4
アイテム 2　エントリ ID2 エントリ ID3 エントリ ID5
アイテム 3　エントリ ID1 エントリ ID2 エントリ ID6 エントリ ID7 エントリ ID8
…
```

(注意) ここまでのデータは、同じ研究室の別の研究で作成されているが、ここでは説明の都合上、手順の一部として記述している。

Step2.　関係を逆転させ、エントリからそれに出現するアイテムの一覧への対応関係を生成する。

```
エントリ ID1　アイテム 1　アイテム 3　アイテム 14　アイテム 16
エントリ ID2　アイテム 1　アイテム 2　アイテム 3
エントリ ID3　アイテム 1　アイテム 2　アイテム 13　アイテム 17
…
```

Step3.　apriori を実行し、データベース全体から相関ルールを求める。このとき、最大 3 アイテムまでの組み合わせについて、最小支持度 0%、最小確信度 0%のパラメータを用いて、全ての相関ルールを求める。

Step4.　データベースの各フィールドについて、以上の処理を行う。データベースとフィールドの一覧は。**表 5.1** を参照。

表 5.1　本システムで使用するデータベース名とフィールド名の一覧

データベース名	*フィールド名*
brite	FUNCTION MNEMONIC NAME ORGANISM
compound	NAME
enzyme	COFACTOR EFFECTOR INHIBITOR NAME PRODUCT SUBSTRATE
epd	KW OS
genbank	bound_moiety cell_line cell_type clone_lib dev_stage function gene lab_host organelle organism phenotype plasmid product rpt_family sex specific_host standard_name strain sub_clone sub_species tissue_type transposon variety
genome	DEFINITION LINAGE MORPHOLOGY NAME PHYSIOLOGY
litdb	KEYWORD
pdb	KEYWDS
pir	KEYWORDS
pmd	EXPRESSION-SYSTEM PROTEIN SOURCE
prf	NAME
refseq	bound_moiety cell_line cell_type clone_lib dev_stage function gene lab_host organelle organism phenotype plasmid product rpt_family sex specific_host standard_name strain sub_clone sub_species tissue_type transposon variety
swissprot	KW OC OS
transfac	DE OC OS

5.1.2　入力部分

入力部分では、以下の手順で処理を行う。

Step5.　入力画面より、要約を行いたいエントリ集合を入力する。入力したエントリ集合に応じて、使用するデータベースを決定し、要約に使用するフィールドを選択する(複数のフィールドを選択可能)。そして、各オプションを指定する。

Step6.　オプションの指定が正しくない場合は、エラー表示画面を出力する。➔**Step5** へ戻り、オプションの指定を変更。

入力部分は、ユーザから要約するエントリ集合を受け取る窓口である。ユーザは要約したいエントリ集合の入力を行い、各オプションを選択する。**(図 5.2)**

エントリ集合に関しては、STAG や BLAST などゲノムネットで提供されている検索サービスを利用すれば入手できる。本研究では直接 STAG や BLAST などのシステムと連動していないため、エントリ集合をカンマ区切りの形式で、手動で入力するようにしている。

次に、どのフィールドを使って要約を行うかを選択する。選択するフィールドが増えればより多くの情報を利用した要約ができるが、計算が遅くなるため、ユーザが必要としないものについては選択しないことにより、計算時間を短縮できる。

次に、発見する相関ルールの最大アイテム数を指定する。最大３個までの組み合わせを選択することができるが、アイテムの組み合わせを増やすと情報量が多くなり見難くなってしまうため、最初は１を指定して要約を行い、満足できなければ徐々に増やしていくような使い方が推奨される。

最後に、要約結果を表示する際に、個々のアイテム(またはアイテムの組み合わせ)の重要度を判定する方法について指定する。具体的には、3 つの指標(共通性、特殊性、共通性×特殊性)のうち、どれを優先的に用いてソートするかという順序を指定する。この組み合わせは以下の６通り存在する。

1 位：共通性,	2 位：特殊性	3 位：共通性×特殊性
1 位：共通性,	2 位：共通性×特殊性,	3 位：特殊性
1 位：特殊性,	2 位：共通性,	3 位：共通性×特殊性
1 位：特殊性,	2 位：共通性×特殊性,	3 位：共通性
1 位：共通性×特殊性,	2 位：共通性,	3 位：特殊性
1 位：共通性×特殊性,	2 位：特殊性,	3 位：共通性

デフォルトでは、1 位：共通性×特殊性，2 位：共通性，3 位：特殊性に設定している。

図 5.2　要約システム入力画面

5.1.3　要約部分

要約部分では、以下の手順で処理を行う。

Step7.　入力されたエントリ集合に関して、**Step2** で作成したデータを検索し、各エントリとアイテムの対応関係を抽出する。

Step8.　**Step7** で抽出した対応関係(すなわち、指定されたエントリ集合に限定した小規模なトランザクション集合)に対して apriori を実行し、各アイテムの相関関係を求める。このとき、最大アイテム数は、CGI のオプションで指定済みである(指定されていない場合はデフォルトで１に設定する)。

Step9.　**Step8** で生成した相関ルールをもとに、**Step4** でデータベース全体から生成した相関ルールとのマッチングを行い、相関ルールの支持度(比率と絶対数の両方)を抽出する。

Step10.　**Step9** で作成したデータから、共通性、特殊性、および共通性×特殊性を計算する。

Step11.　**Step10** で計算した結果を、CGI のオプションで指定した順序に基づいて、昇順でソートする。

➔**Step7** まで戻り、CGI のオプションで指定された各フィールドに対して、同一の処理を行う。指定されたデータベースの全てのフィールドに対して処理が完了したら **Step12** へ続く。

Step12.　**Step11** で作成したフィールドごとの計算結果を一つにまとめ、CGI のオプションで指定された順序に基づいて昇順でソートする。

Step13.　**Step11** のフィールドごとのデータより、ソートされた順にアイテムに番号を割り当てる。また、**Step2** で作成した、エントリとアイテム群の対応データより、アイテムを多く持っているエントリから順番に番号を割り当てる。この番号に従って、アイテムを横軸、エントリを縦軸とする真理値表(どのエントリにどのアイテムが出現しているかを０と１で表現した表)を作成する。

➔CGI のオプションで指定されたデータベースの全フィールドに対して、**Step13** を繰り返す。処理が完了したら **Step14** へ続く。

5.1.4　出力部分

出力部分では、以下の手順で処理を行う。

Step14.　**Step12** の結果を画面に出力する。この出力画面より、エントリ集合内に現れる重要アイテム(その集合に共通かつ特有な専門用)の確認ができる。

➔ユーザが処理結果に満足しない場合は **Step5** へ戻り、新たなエントリ集合やオプション選択を入力する。

Step15.　ユーザの求めに応じて、**Step13** で生成しておいた真理値表を表示する。

➔他の真理値表を確認したい場合は、**Step14** へ戻る。

➔新たなエントリ集合やオプション選択を入力したい場合は、**Step5** へ戻る。

出力としては、まず、重要度順にアイテム(またはアイテムの組み合わせ)のリストをユーザに提示する(**図 5.3**)。さらに、ユーザの要求に応じて、真理値表を提示する(**図 5.4**)。

図 5.3　出力結果 1（重要アイテムセットのリスト）

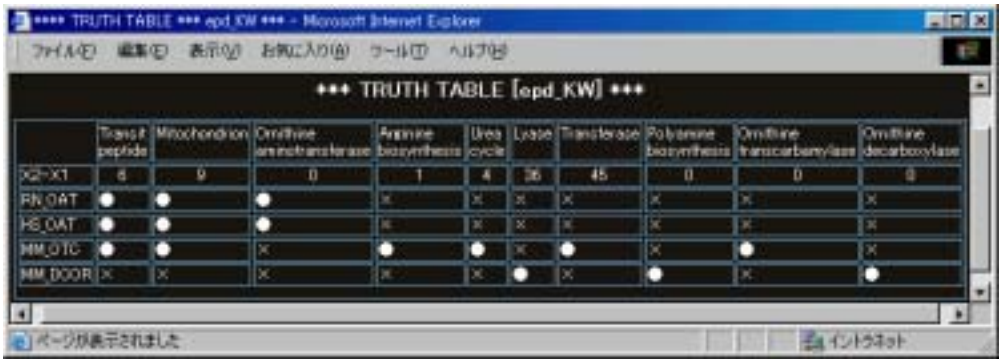
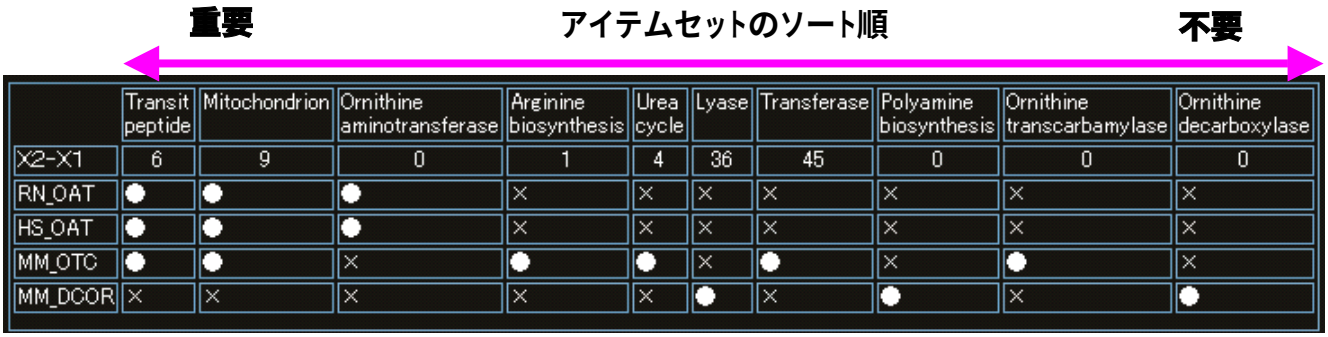


図 5.4　出力結果 2（真理値表）

ユーザが入力部分のデータベース、及びフィールドを複数選んだ場合、出力結果 1 の重要アイテムセットに関しては、ユーザが指定したデータベースの各フィールドの

情報を混ぜ合わせたソート結果を表示することができる。各フィールドを混ぜ合わせて表示した際には、どのフィールドから採取した情報であるか理解できるように、データベースとフィールド名も、アイテムの横に記述される。また、フィールドに関してはユーザが一目でわかるように、カラー表示で区別できるようにしている。

出力結果 2 の真理値表に関して、重要アイテムセットを表示する出力結果 1 の画面から、各フィールドの真理値表をユーザが選択して表示できるようになっている。真理値表は、画面の左側に重要であると判断されたアイテムが配置され、右側に重要でないと判断されたアイテムが配置される。この、重要度順に配置されたアイテムの真理値に従って、重要なアイテムの真理値がなるべく近くに揃うように、エントリが配置される。



	Transit peptide	Mitochondrion	Ornithine aminotransferase	Arginine biosynthesis	Urea cycle	Lyase	Transferase	Polyamine biosynthesis	Ornithine transcarbamylase	Ornithine decarboxylase
X2-X1	6	9	0	1	4	36	45	0	0	0
RN_OAT	●	●	●	×	×	×	×	×	×	×
HS_OAT	●	●	●	×	×	×	×	×	×	×
MM_OTC	●	●	×	●	●	×	●	×	●	×
MM_DCOR	×	×	×	×	×	●	×	●	×	●

図 5.5 真理値表におけるアイテムとエントリの配置関係

5.2　要約結果

要約の結果、重要アイテムセットのリスト表示と、真理値表によるエントリ集合のグループ化表示を出力することができる。両方の結果を見ることにより、ユーザが興味を持っているエントリ集合の意味を把握し、知識の発見につながることが期待される。本節では両方の結果の見方と分析の仕方について述べる。

5.2.1　重要アイテムセット(重要な専門用語の提示)

重要アイテムセットを確認することにより、ユーザは入力したエントリ集合に共通かつ特有な専門用語の有無を知ることができるが、相関ルール発見における最大アイテム数の指定により、得られる結果は異なってくる。最大アイテム数を 1 に指定した場合、ユーザはアイテム単体に関する理解しか得ることができない。しかし、最大アイテム数の長さを増やせば、複数のアイテムを組み合わせてエントリ集合を理解することができる。

以下の例は、本システムを使用して得られる重要アイテムセットのリストを用いて、最大アイテム数の長さを 1 にした場合と 3 にした場合との違いについて述べたものである。ここでは、STAG を用いて「ornthine」という言葉を持つ EPD データベースのエントリを検索した結果得られた 4 つのエントリを入力として、要約を行った。オプション等は以下のように指定した。

データベース	: EPD
エントリ集合	: RN_OAT, MM_OTC, HS_OAT, MM_DCOR
フィールド名	: KW
ソートに使用する指標	: 1 位:共通性×特殊性, 2 位:共通性, 3 位:特殊性

図 5.6　最大アイテム数の長さが 1 の時(上図)と 3 のとき(下図)に得られるアイテムの違い

最大アイテム数が 1 の場合、「transferase」というアイテムは最下位となった。この結果を見る限りでは、「transferase」というアイテムは、単体ではあまり重要でないという判断になってしまう。しかし、最大アイテム数の長さが 3 の場合では、「transferase」というアイテムと他のアイテムの組み合わせに重要なものがあると判断できる。このように、最大アイテム数を変えることで、重要なアイテムの組み合わせを発見できる可能性がある。

5.2.2　真理値表

真理値表を表示することにより、入力されたエントリ集合の中に注目すべきサブグループを発見できる可能性がある。もちろん、入力したエントリ集合がまとまった意味を持たない集合(例えば、ランダムに選んだエントリ集合等)であった場合は、エントリ集合に更なるグループを発見できる可能性は当然低くなる。しかし、本研究で要約を行うエントリ集合は、何かしらの検索(キーワード検索、ホモロジー検索、モチーフ検索等)により得られるものを想定しているので、少なくとも何かの関連性があることが期待される。それでも、与えられたエントリ集合を要約した結果、その集合の関連性(グループ)が判然としない場合もある(図 5.7)。このような場合、アイテム数が少なすぎたり、対象としたフィールドが悪かったり、エントリ集合の大きさが小さすぎた等、幾つかの理由が考えられる。逆に言えば、データベースのフィールドの選択や、入力するエントリ集合を得た過程や、対象とするデータベースのフィールドに存在するエントリの多さ(表 5.2)などの点に注目してエントリ集合を入力し、各オプションを適切に選択することにより、有用なグループを発見できる可能性が高い。

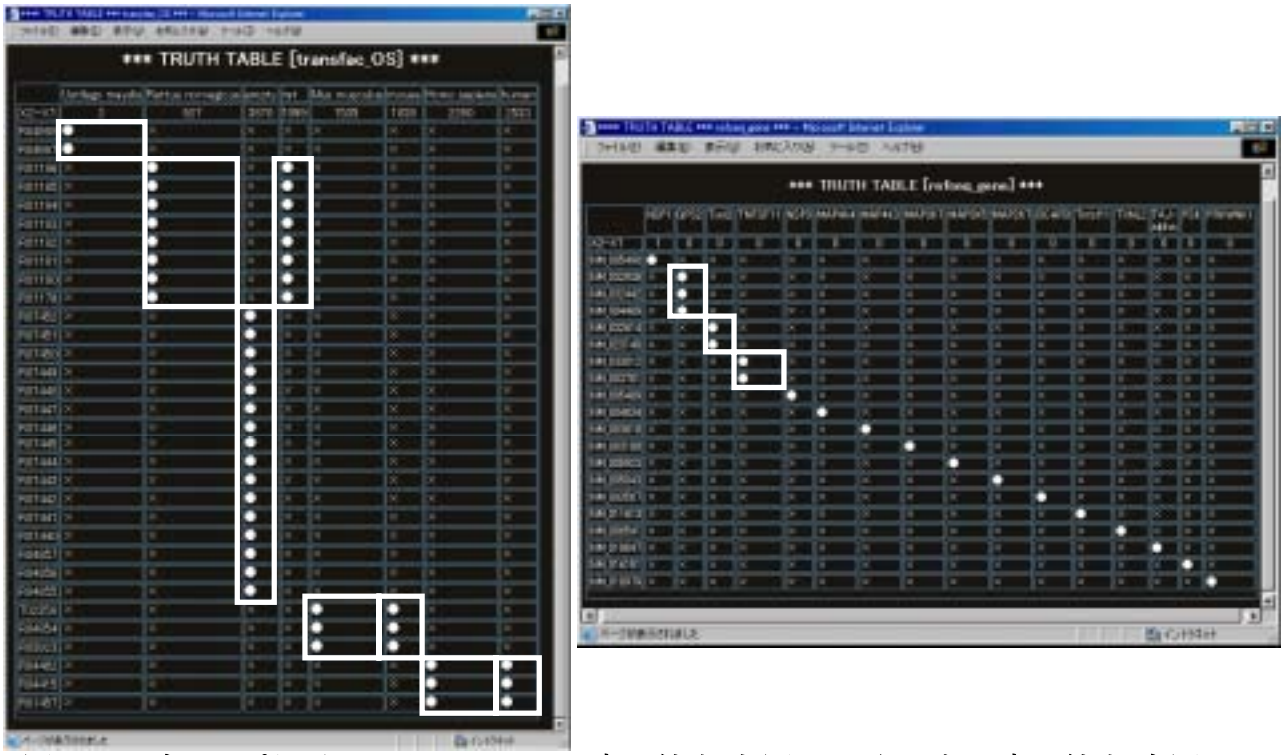


図 5.7　グループ化がはっきりしている真理値表(左図)とそうでない真理値表(右図)

表 5.2　本システムで扱っている各データベースのアイテム数とエントリ数

アイテム数	Y1	ファイル名
61	61	genome:NAME.txt
64	51	genome:PHYSIOLOGY.txt
1063058	391136	lindb:KEYWORD.txt
11929	12090	pdb:KEYWDS.txt
1072	73337	pir:KEYWORDS.txt
1370	9336	pmd:EXPRESSIONSYSTEM.txt
20886	23167	pmd:PROTEIN.txt
2842	23076	pmd:SOURCE.txt
83770	177742	prf:NAME.txt
99	80	refseq:bound:moiety.txt
1179	3001	refseq:cell:line.txt
900	3349	refseq:cell:type.txt
1510	6235	refseq:clone:lib.txt
621	4505	refseq:dev:stage.txt
3176	1730	refseq:function.txt
190279	25184	refseq:gene.txt
46	723	refseq:lab:host.txt
6	293	refseq:organelle.txt
1210	25739	refseq:organism.txt
23	14	refseq:phenotype.txt
285	288	refseq:plasmid.txt
94753	25519	refseq:product.txt
197	199	refseq:rpt:family.txt
18	1856	refseq:sex.txt
73	136	refseq:specific:host.txt
971	401	refseq:standard:name.txt
508	4674	refseq:strain.txt
24	20	refseq:sub:clone.txt
60	337	refseq:sub:species.txt
1006	9356	refseq:tissue:type.txt
12	10	refseq:transposon.txt
6	7	refseq:variety.txt
840	99274	swissprot:KW.txt
4974	101602	swissprot:OC.txt
9736	101602	swissprot:OS.txt
2433	10040	transfac:DE.txt
243	12494	transfac:OC.txt
319	13457	transfac:OS.txt

アイテム数	Y1	ファイル名
41	74	brite:FUNCTION.txt
30	145	brite:MNEMONIC.txt
182	278	brite:NAME.txt
7	204	brite:ORGANISM.txt
11770	7284	compound:NAME.txt
94	645	enzyme:COFACTOR.txt
35	38	enzyme:EFFECTOR.txt
169	111	enzyme:INHIBITOR.txt
6146	3829	enzyme:NAME.txt
2978	3407	enzyme:PRODUCT.txt
3103	3445	enzyme:SUBSTRATE.txt
763	1392	epd:KW.txt
214	1392	epd:OS.txt
2025	3875	genbank:bound:moiety.txt
9530	233006	genbank:cell:line.txt
4569	889140	genbank:cell:type.txt
25939	12126265	genbank:clone:lib.txt
6090	5445706	genbank:dev:stage.txt
19949	34362	genbank:function.txt
322632	410906	genbank:gene.txt
948	7722178	genbank:lab:host.txt
9	112073	genbank:organelle.txt
97622	13602261	genbank:organism.txt
5395	16231	genbank:rpt:family.txt
212	4532025	genbank:sex.txt
5250	27281	genbank:specific:host.txt
19244	19157	genbank:standard:name.txt
88325	4388446	genbank:strain.txt
1258	1497	genbank:sub:clone.txt
3529	26295	genbank:sub:species.txt
6659	5586636	genbank:tissue:type.txt
1779	5039	genbank:transposon.txt
1310	21187	genbank:variety.txt
61	61	genome:DEFINITION.txt
121	61	genome:LINEAGE.txt
20	53	genome:MORPHOLOGY.txt
61	61	genome:NAME.txt

第 6 章
要約結果の考察

この章では、ゲノムネットで提供されている各種サービスと本システムを結合した場合を想定して、**6.1**，**6.2** で述べた代表的な検索サービス(STAG，BLAST)を利用して得られたエントリ集合を本システムに入力した際の要約結果について考察する。また、**6.3** では、データベース ENZYME のエントリがもともと EC 番号という機能分類を持っていることを利用して、ある分類に属するエントリ集合を本システムに入力した際の要約結果について考察する。

6.1 STAG により得られるエントリ集合の要約

STAG による全文検索がうまく機能した場合、適切にフィールドを選択すれば、得られたエントリの中には検索に用いたワード(検索語)自身に加えて、検索語と関連のある用語(アイテム)も現れることが期待される。しかし、DBGET などに比べると STAG は、多くのエントリがヒットする反面、期待しなかったエントリまで出力される傾向がある。このため、有用なサブグループがエントリの一部でしか見られないことも予想される。

本節では、尿素の生成過程である尿素回路(オルニチン回路)を例に取り、実験を行った。例えば、ホ乳類はオルニチン回路を用いて肝臓でアンモニアを尿素に変えて排出しており、次のような反応が起こる。**(図 6.1)**

1. タンパク質の分解により生じたアンモニア(NH_3)と二酸化炭素(CO_2)がATPや各種の酵素の働きでオルニチンと結合し、シトルリンになる。
2. シトルリンは、さらにATPと酵素の働きで、もう1分子のアンモニアと結合してオルニチンになる。
3. アルギニンは、酵素(アルギナーゼ)の働きにより分解され、はじめのオルニチンと尿素が作られる。

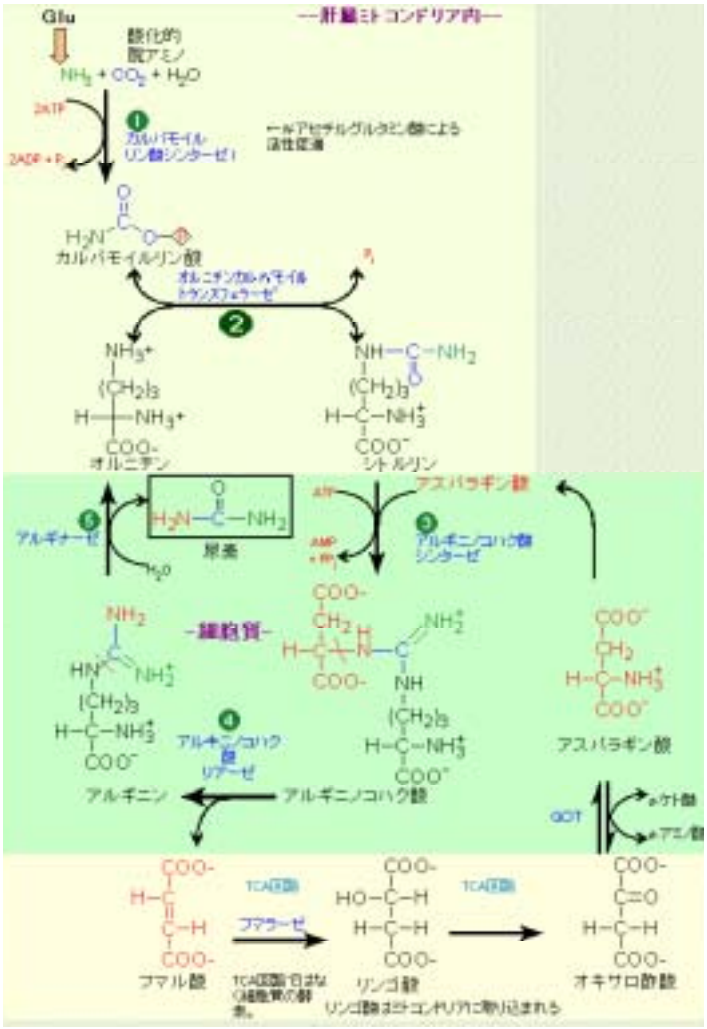


図 6.1　オルニチン回路（参考 Web:**[19]**）

オルニチン回路では、アルギニン，オルニチン，シトルリンが関連していることが分かる。よって、「ornithine」という検索語を入力したのであれば、得られたエントリ集合に「arginine」や「citrulline」というアイテムもしくはその組み合わせがあることが期待できる。また、エントリ集合の一部に、サブグループ(オルニチン回路に関するグループまたはそれ以外のグループ)が発見できるのではないかと考えられる。

以上より、本システムを利用してキーワード検索の結果得られたエントリ集合から、以下のような要約が得られるかを調べる。

- キーワード検索により入力した検索語が、要約結果に現れているか。

- 検索語と関連のあるアイテム(重要アイテム)またはその組み合わせが要約結果として現れているか。
- 与えられたエントリ集合から、サブグループの発見はできているか。

STAG に入力した検索語は「ornithine」とし、得られたエントリ集合を本システムに与えて、要約を行う。データベースは ENZYME を、フィールドは SUBSTRATE と PRODUCT をオプションにて選択する。SUBSTRATE フィールドには、基質に関する情報が記述されており、PRODUCT フィールドには、生成物質に関する情報が記述されているので、ユーザはこれらに関する情報を得ることができる。

データベース　　：ENZYME
エントリ集合　　：
3.1.1.48, 1.5.1.24, 5.1.1.12, 2.6.1.69, 2.3.1.35, 2.1.3.3, 2.3.1.127, 5.4.3.5, 4.1.1.17, 3.5.1.16, 2.6.1.13, 4.3.1.12, 2.6.1.68, 3.5.3.10, 3.4.13.4, 2.6.1.11, 3.5.3.1, 3.5.1.20, 2.3.1.109, 2.1.4.2, 2.1.4.1, 2.1.3.6, 1.5.1.19, 1.4.3.14, 5.4.3.1, 5.1.1.10, 5.1.1.9, 3.6.3.21, 3.5.3.12, 2.6.1.21, 1.14.13.59, 1.5.1.11, 1.4.3.3, 1.4.1.12, 1.2.1.12
フィールド名　　：SUBSTRATE，PRODUCT
最大アイテム数　　：k＝3（重要アイテムセットの表示の際）
k＝1（真理値表の表示の際）
ソートに使用する指標　：1 位:共通性×特殊性，2 位:共通性，3 位:特殊性

6.1.1　重要アイテムセットのリスト

重要アイテムセットは、ユーザが指定したソート順に基づき、個々のアイテム(またはアイテムの組み合わせ)を抽出したものである。よって、最大アイテム数の長さが 1 個であれば重要語が単体で理解できるし、最大アイテム数の長さを増やせば重要語の組み合わせを対象とした相関を見ることができる。実験結果を図 **6.2** に示す。

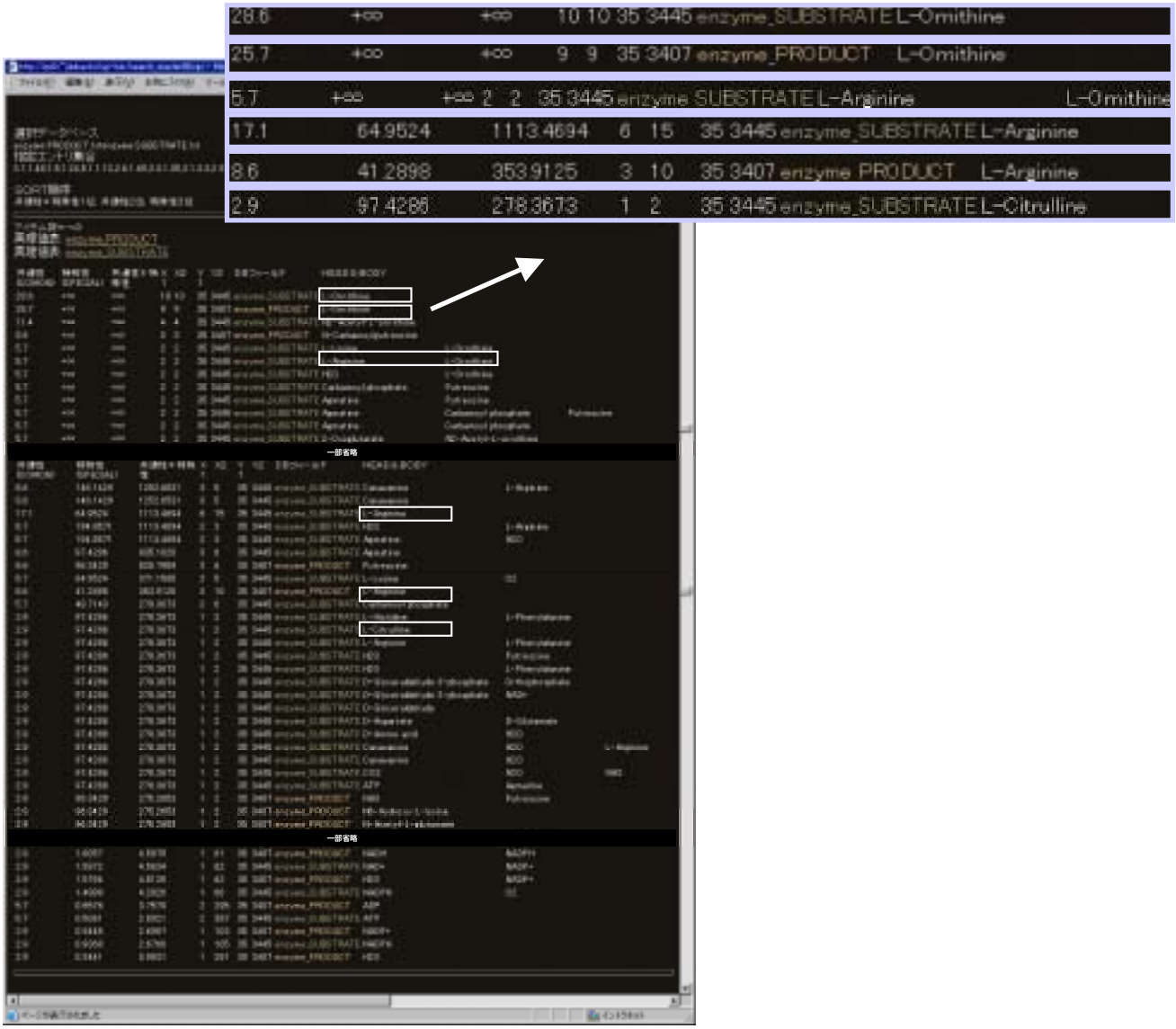


図 6.2　本システムの要約結果(重要アイテムセット)

キーワード検索により得られたエントリ集合を入力したので、検索語の「ornithine」に関しては、各フィールド(SUBSTRATE，PRODUCT)で、正の無限大に発散した結果が見られた。この結果から、そのエントリ集合内にしか「ornithine」は出現していないといえる。また、重要アイテムセットには、期待通りオルニチン，アルギニン，シトルリンが出現しており、指定した重要語抽出のタスクに基づき上位に位置していた。組み合わせに関しては、オルニチンとアルギニンの２組の組み合わせが上位に出現していた。しかし、オルニチン，シトルリン，アルギニンの３組の組

み合わせについてまでは出現してはいなかった。

以上の結果から、キーワード検索で得られたエントリ集合の要約結果には、検索語と関わりのある情報が要約結果として上位に現れるといえる。しかし、これは対象とするフィールドにより異なる。例えば採取生物に関するフィールドから、「ornithine」に関する情報が得られることはない。言いかえれば、フィールドの選択により「どの観点から要約を試みるか」が決定されるといえる。

6.1.2　真理値表

真理値表では、サブグループの発見が目的である。また、重要アイテムを選択し、それが含まれているエントリを個別に調べることも行える。

共通性 (COMON)	特殊性 (SPECIAL)	共通性＊特殊性	X 1	X2	Y 1	Y2	DBフィールド	HEAD＆BODY	
8.6	146.1429	1252.6531	3	5	35	3445	enzyme_SUBSTRATE	Canavanine	L-Arginine
8.6	146.1429	1252.6531	3	5	35	3445	enzyme_SUBSTRATE	Canavanine	
17.1	64.9524	1113.4694	6	15	35	3445	enzyme_SUBSTRATE	L-Arginine	
5.7	194.8571	1113.4694	2	3	35	3445	enzyme_SUBSTRATE	H2O	L-Arginine
5.7	194.8571	1113.4694	2	3	35	3445	enzyme_SUBSTRATE	Agmatine	H2O
8.6	97.4286	835.1020	3	6	35	3445	enzyme_SUBSTRATE	Agmatine	
8.6	96.3429	825.7959	3	6	35	3407	enzyme_PRODUCT	Putrescine	
5.7	64.9524	371.1565	2	5	35	3445	enzyme_SUBSTRATE	L-Lysine	O2
8.6	41.2898	353.9125	3	10	35	3407	enzyme_PRODUCT	L-Arginine	
5.7	48.7143	278.3673	2	6	35	3445	enzyme_SUBSTRATE	Carbamoyl phosphate	
2.9	97.4286	278.3673	1	2	35	3445	enzyme_SUBSTRATE	L-Histidine	L-Phenylalar

図 6.3　重要アイテムセット（一部抜粋）

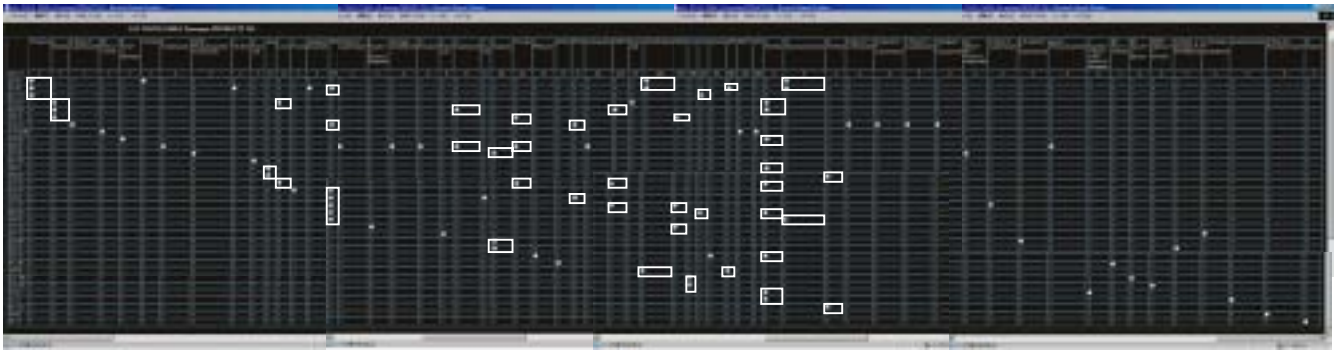


図 6.4　真理値表（ENZYME データベースの PRODUCT フィールド）

図 6.4 は、ENZYME データベースの PRODUCT フィールドに関して示した真理値表である。真理値表の一部にサブグループが形成されているが、サブグループが明確にわかれているとはいえない。同じく、SUBSTRACT に関しても同様に、サブグループの発見は明確とはいえない。これに関しては、ソート順のさらなる工夫により今後改善する可能性があると思われる。

重要アイテムセットの要約結果から興味をもった重要アイテムが含まれているエントリを、真理値表から探すこともできる。例えば、先ほどの重要アイテムセットから重要語である、「L-Arginine」に興味をもったとする。そのアイテムが、どのフィールドより出現していたかを確認し、そのフィールドに関する真理値表を確認する。今回の検索結果では、SUBSTRATE, PRODUCT のどちらにも「L-Arginine」は出現している（図 6.5）。しかし、SUBSTRATE に記述されている「L-Arginine」は基質に関する情報であり、PRODUCT に記述されている「L-Arginine」は生成物質に関する情報である。アイテム名に関しては同じであるが、記述されているフィールドは異なり、情報としての意味も異なる。よって、これらを含むエントリ集合も異なっている。ユーザは、どのような情報が欲しいかにより重要語のフィールドを選択し、重要語を選択する必要がある。この確認により、興味をもったエントリ ID がわかり、エントリ集合を直接選択できる。

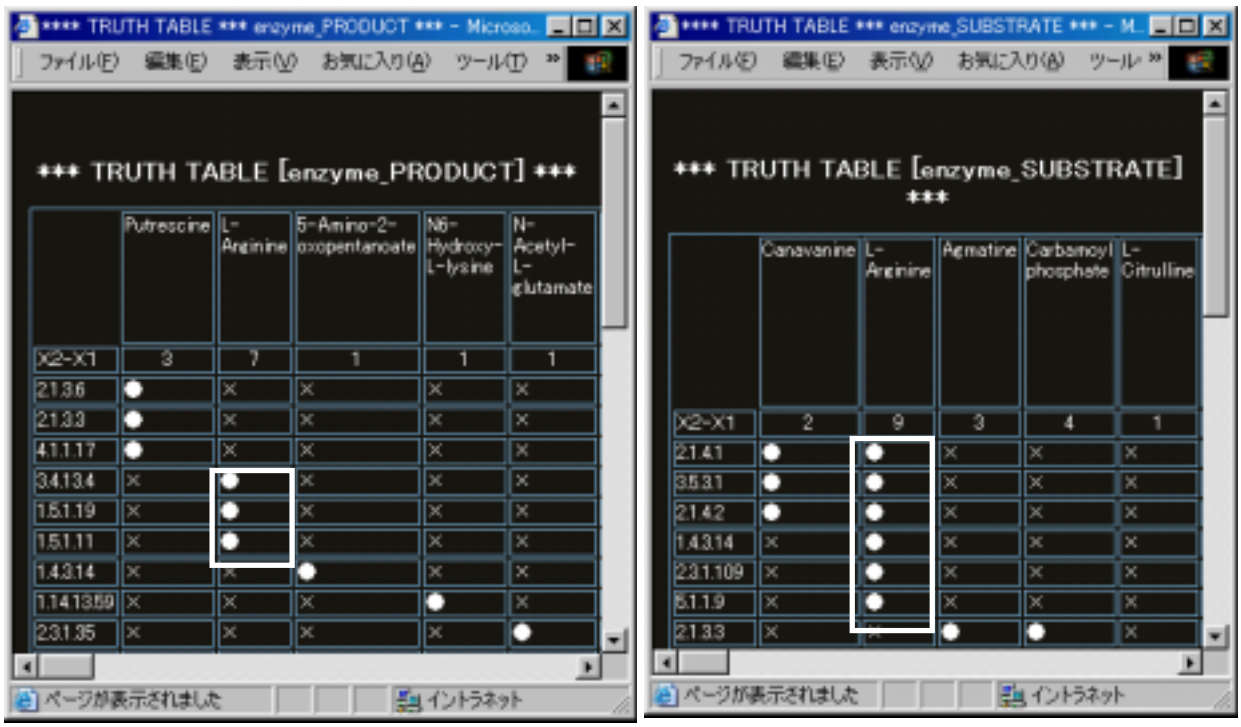


図 6.5　真理値表（一部抜粋）

6.2　ホモロジー検索(BLAST)により得られたエントリ集合の要約

ホモロジー検索では、質問配列に類似した配列が記述されているエントリ集合が得られる。ホモロジー検索にヒットするエントリ集合は、「配列が似ているならば機能も似ていることが多い」と考えられるため、全文検索から得られるエントリ集合よりも、意味的にまとまったものになることが予想される。しかし、質問配列全体と相同なものばかりがヒットするとは限らない。例えば、質問配列中に３つの特徴的な部分領域 A、B、C があったとすると、それぞれに相同な部分だけを持つ配列や、あるいは A と B、B と C について相同な配列がヒットすることも考えられる。質問配列がアミノ酸配列である場合、仮にこれらの特徴的な部分領域がタンパク質の機能と結びついているならば、ホモロジー検索でヒットした各配列は、どの領域について相同かによって、そのエントリが持つ専門用語情報(アイテム)が異なったり、あるいは真理値表においてグループができることが考えられる。

本節では、ホモロジー検索によって得られたエントリ集合を要約システムにかけた結果について考察する。ホモロジー検索を行うための質問配列としては、ユビキチンの配列を用いた。ユビキチンは同じ配列が繰り返されているため、この繰り返しを排除したポリユビキチンのアミノ酸配列(タンパク質配列)を問い合わせ配列とし、BLAST に入力した。得られたエントリ集合は、配列の類似性を示す評価指標 P 値が、0.001 以上を示したものに限定した。

データベース:エントリ ID　　: GenBank:M26880 からアミノ酸へ翻訳
問い合わせ配列(アミノ酸配列)　:
MQIFVKTLTGKTITLEVEPSDTIENVKAKIQDKEGIPPDQQRLIFAGK
QLEDGRLSDYNIQKESTLHLVLRLRGG
BLAST のプログラム　　: BLASTP
BLAST の指定データベース　: PIR
抽出エントリ　　: P 値≦0.001 の有意なエントリのみ抽出
エントリ集合　　:

S11248, S10319, S21083, S45359, I50438, I51568, I45964, S11296, A29584, JH0302, S42750, S28203, A26087, UQWO7A, UQFFR7, I52328, UQHUR7, I65237, UQHUR, UQHUC, S13928, A31560, UQHUB, S12583, I50437, S29853, UQHY, UQBO, UQHU, B48470, A49768, UQUYSF, T27638, JC5226, S43306, S32020, T16144, A30126, S25154, S34333, JN0790, S34334, S18535, JC5489, S40611, S25848, S34655, JT0492, S12114, UQKM, S53719, UQFFM, T51753, T50481, T39061, T52335, T52334, H86367, S40240, S40239, JS0657, C36571, S25305, JH0227, JH0226, D36571, T37547, S33633, S34662, S28420, A29456, A36571, E85063, B27806, D34080, S34285, A34080, H85066, S17435, S30151, S38669, T12035, T45526, T40261, B34080, C34080, A27806, T48345, T02358, G85036, C86439, S55242, JQ1728, S25164, S17436, PS0428, A25062, S16263, PS0380, S19799, S42643, S12161, UQTO7A, UQBYR7, UQDOR, S20925, UQBY, UQPM, UQMUM, S49332, S28426, UQFS, UQSY, UQOA, UQDOR7, UQJNI, T46664, S45304, UQUTRC, UQNC, UQUTC, S55245, S55244, C48111, UQUT, A56582, S34332, S62680, S29238, S31653, T04026, T28305, S62909, S62740, UQNCR, S62908, S55243, JQ2029, T10294, A29526, T41781, C72854, UQNVAC, T30390, JT0491, F96579, S43121, T07633, T22249, JN0710, A96580, T39965, G86254, S51867, T32805, S25001, S20863, JN0674, JN0673, S01625, H96579, B48766, I68527, A31084, A48766, T32806, T07649, H85134, T06496, T14336, T30561, T29404, A28138, T10540, F84903, S44346, A47416, T04150, JC1278, I48346, C84549, B84549, T14337, T01396, F96827, G86296, S60735, A35098, S54583, S44443, T51797, E85042, G85042, S26434, T32790, D86452, T39070, G85065, S19802, T29399, T51479, A49007, T00588, T38404, T40115

データベース　　：PIR

フィールド名　　：KEYWORDS

最大アイテム数　　：k＝3（重要アイテムセット表示の際）

k＝1（真理値表の表示の際）

ソートに使用する指標　　：1 位:共通性×特殊性，2 位:共通性，3 位:特殊性

6.2.1　重要アイテムセット

ホモロジー検索により得られるエントリ集合を要約した結果の重要アイテムセットは、特殊性の高い情報がほとんどであった**(図　6.6)**。これは、このエントリ集合に特有な情報が多く得られたという意味では、良い結果だと言える。この中には、単体では特殊性が低いアイテムだが、組み合わせると非常に特殊性が高くなるものも多く見られた。

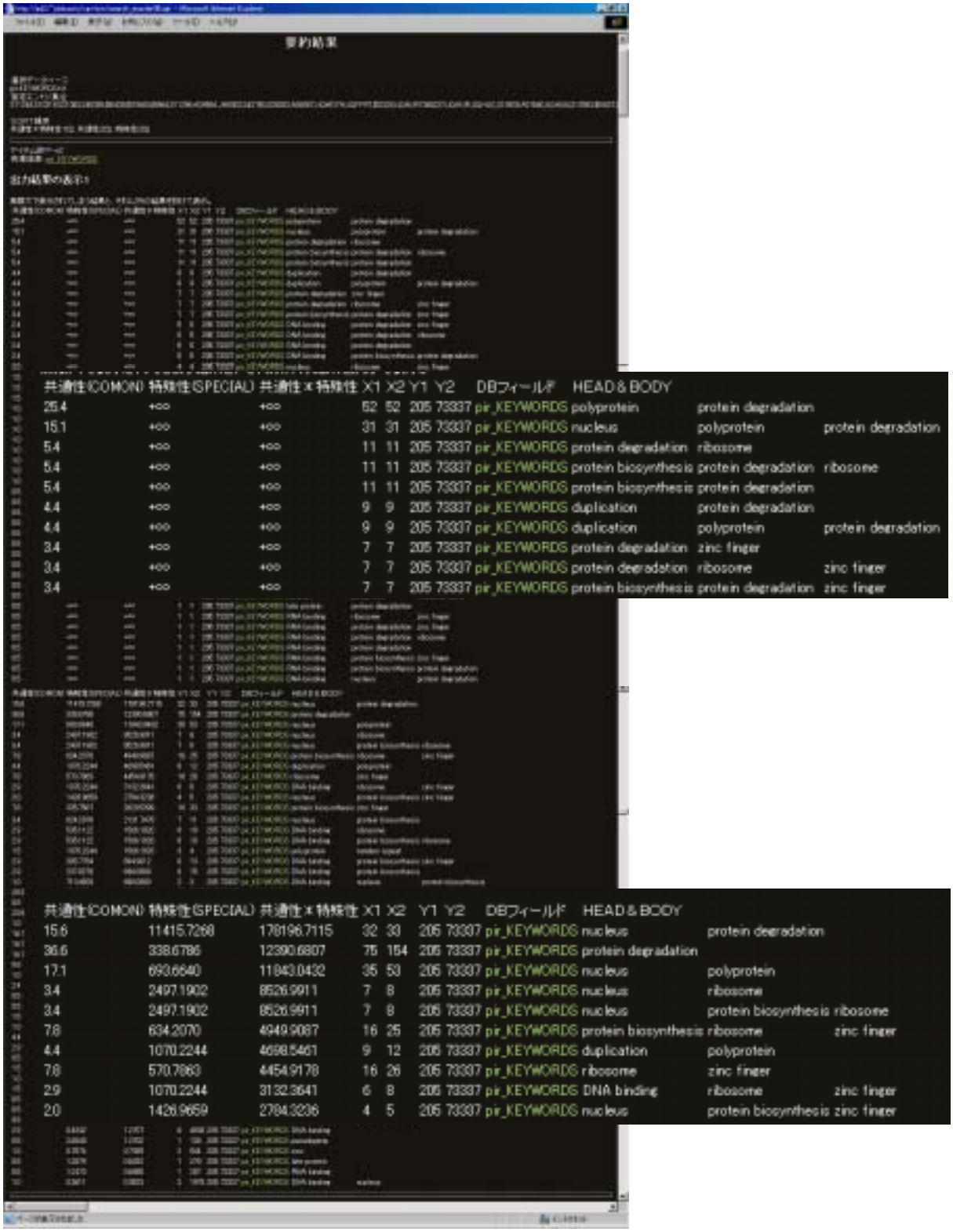
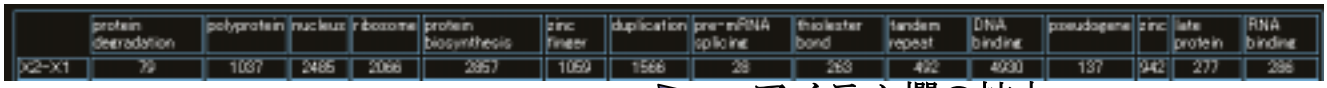


図 6.6　重要アイテムセット（PIR データベースの KEYWORDS フィールド）

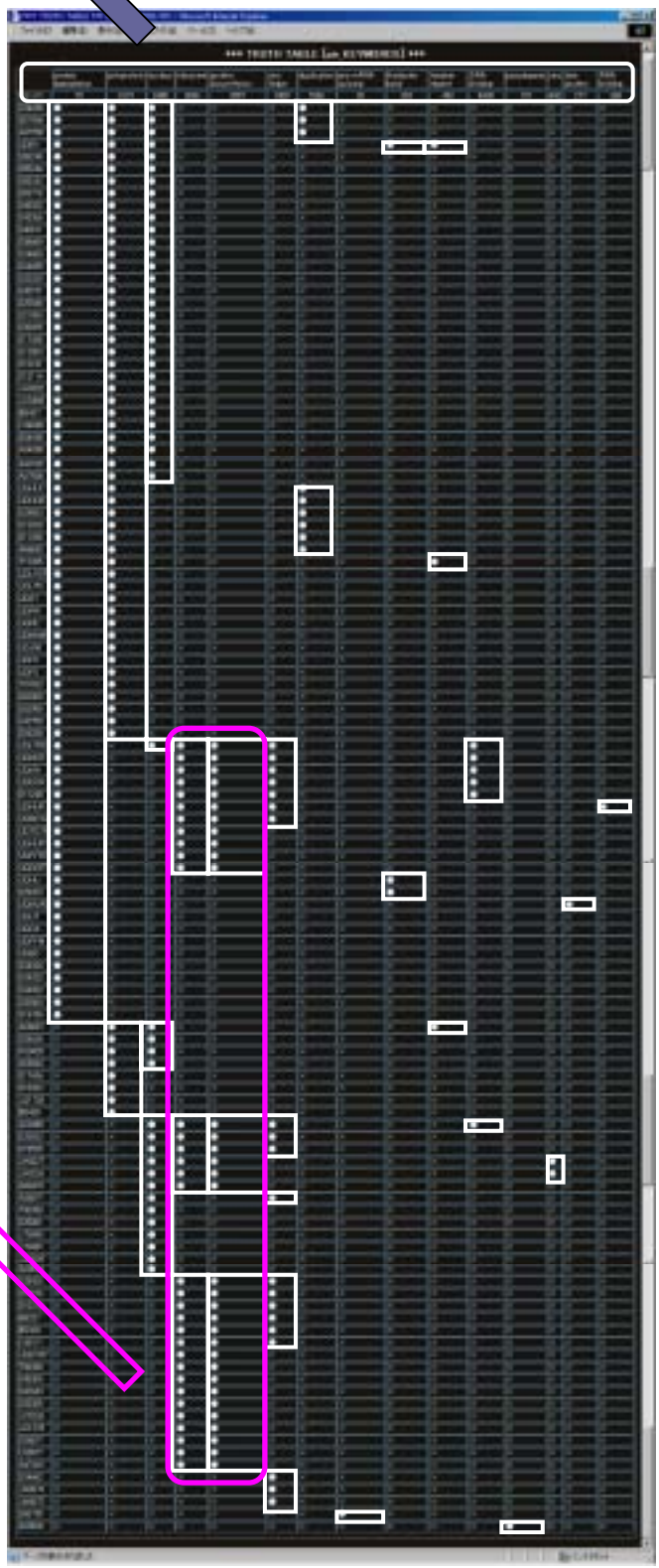
6.2.2　真理値表

この実験では、いくつかの明確なサブグループを真理値表から発見することができた。ここで、着目したいのは5列目と6列目のアイテムである「ribosome」と「protein biosynthesis」についてである。グループは双方ともに、同じエントリに見られている。このことから、ユビキチンのタンパク質配列(アミノ酸配列)を入力して得られたエントリ集合(ホモロジー検索結果)においては、双方向の相関が必ず成立することが分かる。実際、「ribosome(リボゾーム)」はタンパク質を合成する場であり、「protein biosynthesis」はタンパク質生合成を意味するので、両者に関連があることは妥当であり、適切なグループ化が行えたといえる。**[17]**

また、本節で最初に述べたように、同じ相同領域を持つもの同士がグループ化されているのもこの例である。**図 6.7** に示した、グループの配列情報を調べてみた結果を**図 6.8** に示す。**図 6.8** より、「ribosome」と「protein biosynthesis」が記述されているエントリのグループの配列情報は、全て同じ配列部分で一致していることが読み取れる。



アイテム欄の拡大



「r

UQUTRC	A29456	S28420
UQNCR	S33633	S25154
UQKM	S18535	S10319
UQDOR	JH0227	I65237
S11248	JH0226	I52328
UQHUR	B48470	C48111
UQWO7A		UQBYR7
UQTO7A		T06496
UQHUR7		S45304
UQFFR7		S42643
UQDOR7		S25305
		JC5226
		JC1278
		D36571
		C36571
		A47416

図 6.7　真理値表（ホモロジー検索からのエントリ集合）

```
>pir:S11248 ubiquitin /ribosomal protein CEP52 - mouse (fra  ment)
        Length = 94
 Score =  151 bits (381), Expect = 5e-37
 Identities = 76/76 (100% ), Positives = 76/76 (100% )
Query: 1  MQIFVKTLTGKTITLEVEPSDTIENVKAKIQDKEGIPPDQQRLIFAGKQLEDGRTLSDYN 60
        MQIFVKTLTGKTITLEVEPSDTIENVKAKIQDKEGIPPDQQRLIFAGKQLEDGRTLSDYN
Sbjct: 1  MQIFVKTLTGKTITLEVEPSDTIENVKAKIQDKEGIPPDQQRLIFAGKQLEDGRTLSDYN 60
Query: 61 IQKESTLHLVLRLRGG 76
        IQKESTLHLVLRLRGG
Sbjct: 61 IQKESTLHLVLRLRGG 76
```

```
>pir:S10319 ubiquitin /ribosomal protein CEP52 - fruit fly
          ros              n


           =                              6/76 (100% )
Query: 1  MQIFVKTLTGKTITLEVEPSDTIENVKAKIQDKEGIPPDQQRLIFAGKQLEDGRTLSDYN 60
           F                              NVKAKIQDKEGIPPDQQRLIFAGKQLEDGRTLSDYN
Sbjct: 1  MQIFVKTLTGKTITLEVEPSDTIENVKAKIQDKEGIPPDQQRLIFAGKQLEDGRTLSDYN 60
              KESTLHLV
       IQKESTLHLVLRLRGG
              ESTLHLVL
```

```
>pir:B48470 ubiquitin /ribosomal protein CEP52 - Eimeria bovis
        Length = 129
 Score =  150 bits (378), Expect = 1e-36
 Identities = 75/76 (98% ), Positives = 76/76 (99% )
Query: 1  MQIFVKTLTGKTITLEVEPSDTIENVKAKIQDKEGIPPDQQRLIFAGKQLEDGRTLSDYN 60
        MQIFVKTLTGKTITL+VEPSDTIENVKAKIQDKEGIPPDQQRLIFAGKQLEDGRTLSDYN
Sbjct: 1  MQIFVKTLTGKTITLDVEPSDTIENVKAKIQDKEGIPPDQQRLIFAGKQLEDGRTLSDYN 60
Query: 61 IQKESTLHLVLRLRGG 76
        IQKESTLHLVLRLRGG
Sbjct: 61 IQKESTLHLVLRLRGG 76
```

図 6.8 「 biosynthesis」が
記述されているエントリの配列情報

6.3　ENZYME データベースの EC 番号に着目したエントリ集合の要約

ENZYME は酵素に関する情報のデータベースである。ENZYME のエントリ ID は EC 番号(Enzyme Commission Number)と呼ばれる４桁の番号により分類されている。最初の数字が、酵素の機能を表す種類(大分類)を示し、２番目の数字は１番目の機能を細分類している。このようにして、４番目の数字までたどることで酵素の機能を一番小さなグループまで分類している。**[16,18]** どのように分類されているかに関しては、Web を利用して調べることができる(**図 6.9**)。

- Enzyme Nomenclature（IUBMD）
  http://www.chem.qmw.ac.uk/iubmb/enzyme/
- ENZYME Enzyme nomenclature database（ExPASy）
  http://kr.expasy.org/enzyme/

図 6.9　EC 番号の分類例（参考 Web:**[18]**）

この EC 番号を利用すれば、要約の結果として真理値表に現れるグループが有意味なものであるかどうか検証することができる。また、EC 番号の分類がどの程度まで細分類できているか(それ以上に細かいグループが存在するか)を確認できる。

以下の実験では、EC 番号の上位 3 桁が同じ数値であるエントリ集合と、上位２桁が同じ数値であるエントリ集合を使って、要約を行う。EC 番号に照らして同じ分類に属するエントリ集合には、機能的な共通性があることから、先のホモロジー検索の要約結果と同様に、良い要約結果が得られることが期待される。本節では、ホモロジー検索結果と似たような結果となることから、重要アイテムセットのリストについては省略する。

6.3.1　真理値表 1（EC 番号上位 2 桁のグループ）

EC 番号の上位２桁が等しいエントリ集合から、サブグループの発見を目指す。ENZYME データベースの PRODUCT フィールドから、EC 番号が「2.8.＊」であるようなエントリ全てを選択し、本システムに入力した(分類情報は以下)。

2.＊	Transferases（転移酵素）
2.8.＊	Transferring Sulfur-Containing Groups
2.8.1.＊	Sulfurtransferases
2.8.2.＊	Sulfotransferases
2.8.3.＊	CoA-transferases
2.8.4.＊	Transferring alkylthio groups

このとき発見されるサブグループは、EC 番号による上位 3 桁のグループと同じであることが望ましい。

データベース　　：ENZYME
フィールド　　：PRODUCT
対象 EC 番号　　：2.8.＊グループ
エントリ集合　　：
2.8.1.1 ,2.8.1.2 ,2.8.1.3 ,2.8.1.4 ,2.8.1.5 ,2.8.1.6 ,2.8.2.1 ,2.8.2.10 ,2.8.2.11,
2.8.2.12 ,2.8.2.13 ,2.8.2.14 ,2.8.2.15 ,2.8.2.16 ,2.8.2.17 ,2.8.2.18 ,2.8.2.19 ,
2.8.2.2 ,2.8.2.20 ,2.8.2.21 ,2.8.2.22 ,2.8.2.23 ,2.8.2.24 ,2.8.2.25 ,2.8.2.26 ,

2.8.2.27 ,2.8.2.28 ,2.8.2.3 ,2.8.2.4 ,2.8.2.5 ,2.8.2.6 ,2.8.2.7 ,2.8.2.8 ,2.8.2.9,
2.8.3.1 ,2.8.3.10 ,2.8.3.11 ,2.8.3.12 ,2.8.3.13 ,2.8.3.14 ,2.8.3.2 ,2.8.3.3 ,
2.8.3.5 ,2.8.3.6 ,2.8.3.7 ,2.8.3.8 ,2.8.3.9

最大アイテム数　　：k＝1

ソートに使用する指標　：１位:共通性×特殊性，２位:共通性，３位:特殊性

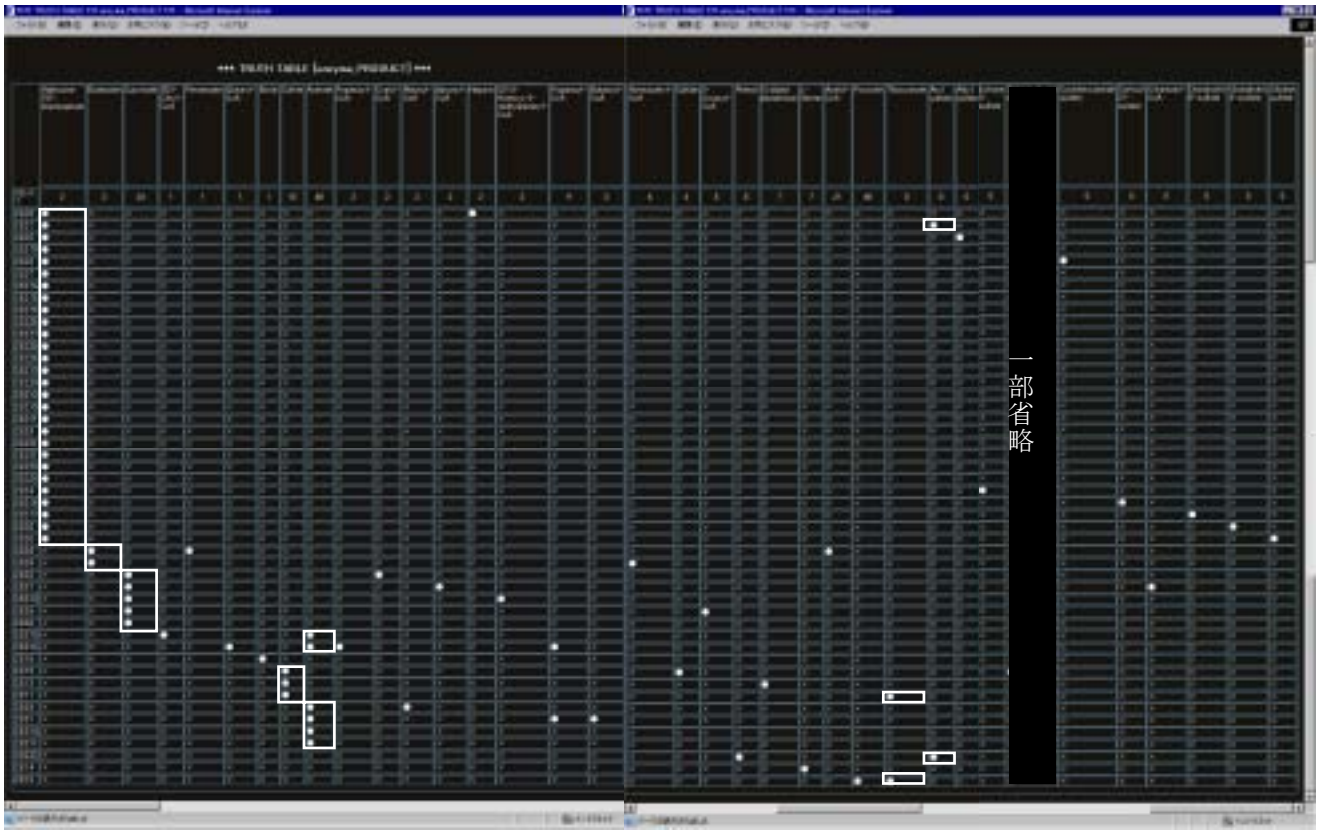


図 6.10　EC 番号 2.8 グループにおける真理値表

要約結果(**図 6.11**)は、１列目の「Adenosine 3',5'-bisphosphate」と 7 列目の「Aryl sulfate」においてサブグループが発見された。ここで発見されたグループは、EC 番号 2.8.2 グループの細分類に基づき分類されている。また、２列目の「Butanoate」と 3 列目の「Succinate」と 5 列目の「Acetate」においてもサブグループが発見された。ここで発見されたグループは、EC 番号 2.8.3 グループの細分類に基づき分類されている。他にも幾つか、EC 番号に基づいたサブグループが発見されている。他の実験に

よれば、EC 番号による細分類してあるエントリ集合を入力しても、必ずしも細分類に基づいたグループが見られるとは限らないが、今回のように EC 番号の細分類どおりにグループが発見される場合もあることがわかった。

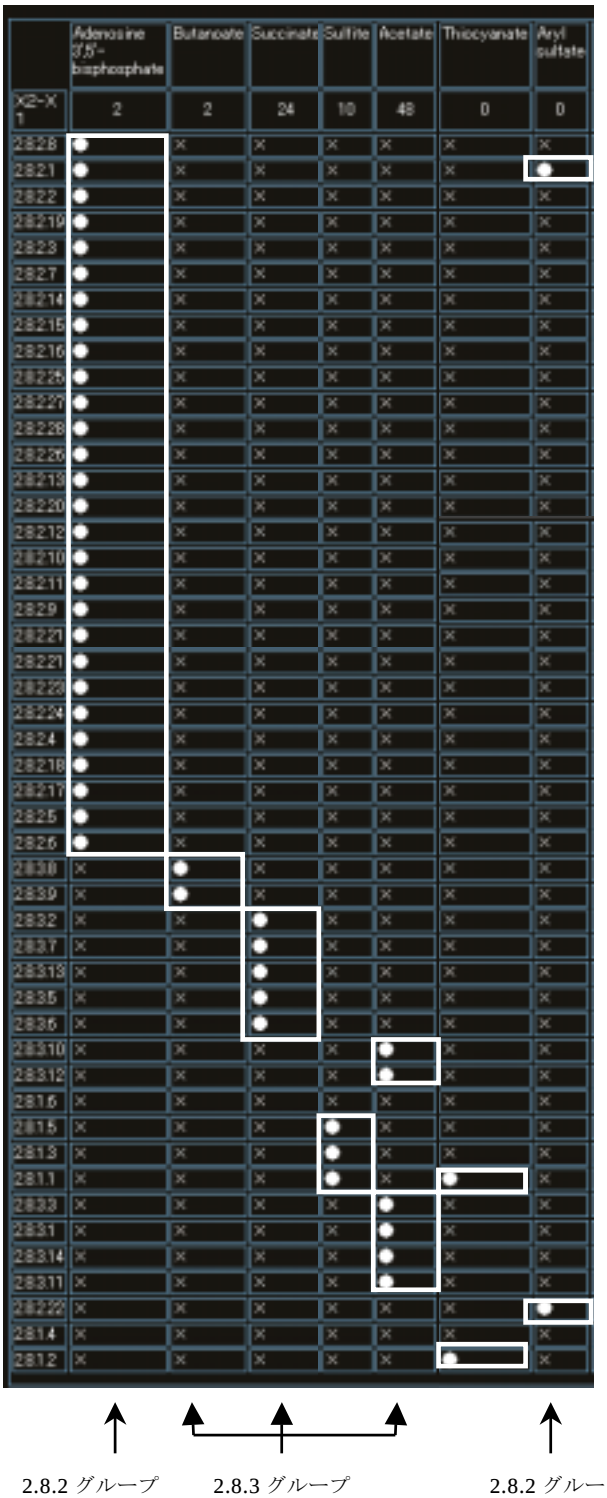


図 6.11　真理値表からの EC 番号に基づいたサブグループの発見

6.3.2　真理値表 2（EC 番号上位 3 桁のグループ）

EC 番号の上位 3 桁が等しいエントリ集合から、サブグループの発見を目指す。EC 番号「1.1.3.＊」をエントリを全て選択し、本システムに入力した(分類情報は以下)。

1.＊	Oxidoreductases.（酸化還元酵素）
1.1.＊	Acting on the CH-OH group of donors.
1.1.3.＊	With NAD or NADP oxygen as acceptor.

実際にシステムに入力したパラメータは以下の通りである。

データベース　　：ENZYME
フィールド　　：PRODUCT
対象 EC 番号　　：1.1.3 グループ
エントリ集合　　：
1.1.3.10, 1.1.3.11, 1.1.3.12, 1.1.3.13, 1.1.3.14, 1.1.3.15, 1.1.3.16, 1.1.3.17, 1.1.3.18, 1.1.3.19, 1.1.3.20, 1.1.3.21, 1.1.3.22, 1.1.3.23, 1.1.3.24, 1.1.3.25, 1.1.3.26, 1.1.3.27, 1.1.3.28, 1.1.3.29, 1.1.3.3, 1.1.3.30, 1.1.3.31, 1.1.3.32, 1.1.3.33, 1.1.3.34, 1.1.3.35, 1.1.3.36, 1.1.3.37, 1.1.3.38, 1.1.3.4, 1.1.3.5, 1.1.3.6, 1.1.3.7, 1.1.3.8, 1.1.3.9
最大アイテム数　　：k＝1
ソートに使用する指標　：１位:共通性×特殊性，２位:共通性，３位:特殊性

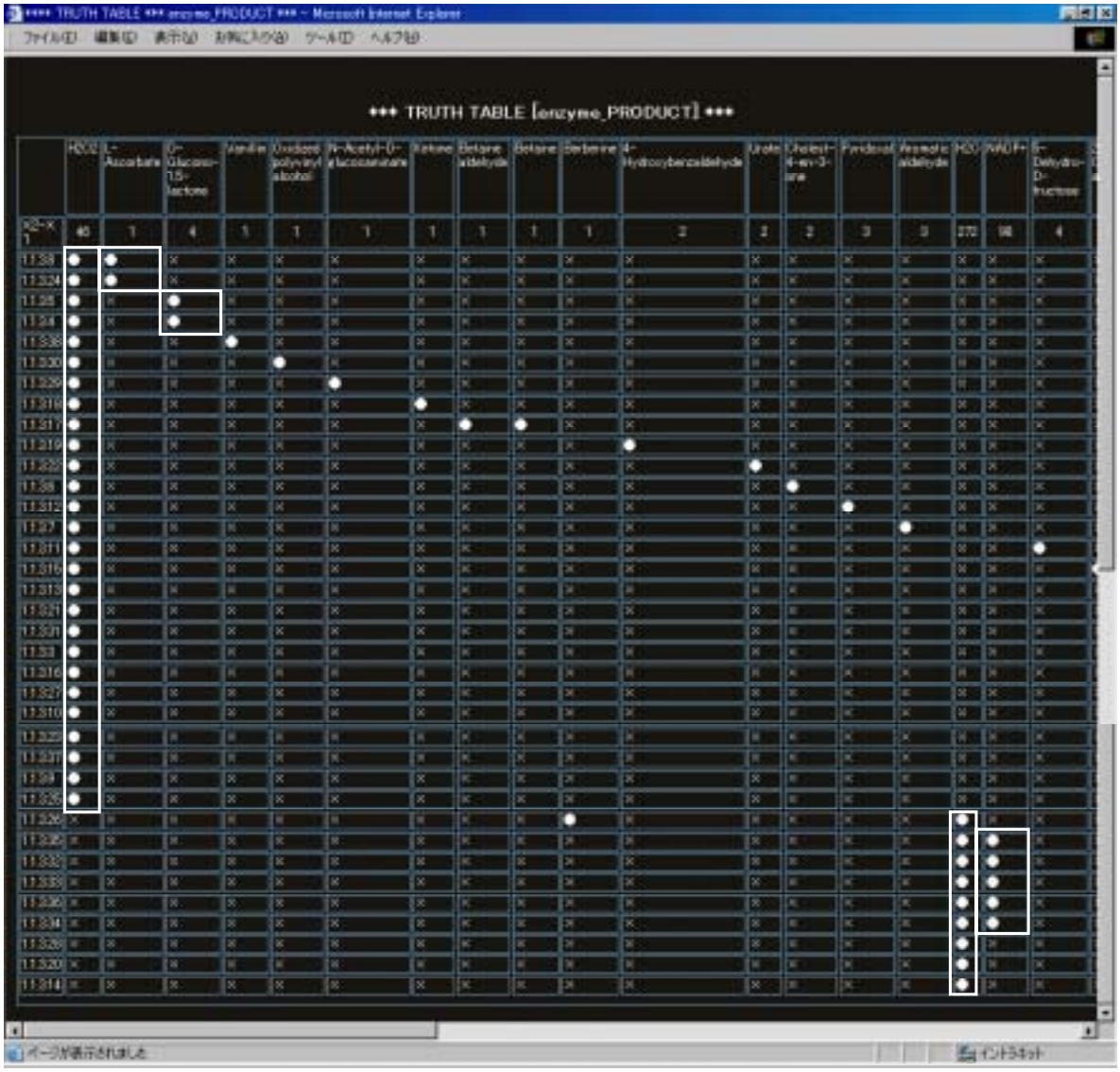


図 6.12　EC 番号 1.1.3 グループにおける真理値表（一部省略）

要約の結果、１列目の「H_2O_2」と 16 列目の「H_2O」により大きく２つのサブグループにわかれることが発見できた(**図 6.12**)。このことは、３桁の EC 番号を使って十分細かく分類されたエントリ集合に対しても、このようにグループの存在を提示できる可能性があることを示しており、ユーザの知識発見に結びつくことが期待される。

第　7　章
おわりに

7.1　結論

本研究では、ゲノムネットで提供されている各種検索サービスを利用して得られるエントリ集合を要約するシステムの構築を行った。本システムを利用することでユーザは、エントリを一つ一つ確認しなくてもエントリ集合の意味を把握できる。エントリ集合の意味を表示する要約結果として、２通りの方法を提供している。一つは、ユーザの入力したエントリ集合に共通かつ特有な専門用語の有無を確認することができる重要アイテムセットの表示。もう一つは、入力されたエントリ集合から注目すべきグループを発見するための真理値表の表示である。

この要約結果を出すために、ユーザが入力したエントリ集合に出現する各アイテムに対して相関ルール発見や各種計算を行っている。このとき二つの点に注目して、計算を行っている。一つは、与えられたエントリ集合の中で共通性の高いアイテム(多くのエントリに出現するアイテム)を抽出する。もう一つは、与えられたエントリの補集合にはなるべく出現しないアイテムを抽出する。この二点を考慮した計算を行うことで、与えられたエントリ内に共通に出現し(共通性)、その補集合にはなるべく出現しない(特殊性)という条件を満たしたアイテムを探すことができる。

本システムを利用した要約結果についていくつか実験を行った結果、以下のことが分かった。

- 全文検索の結果として得られるエントリ集合を要約した場合、検索語と関連のある重要アイテムセットが抽出できた。すなわち、検索語と関連のある情報が要約結果として得られた。しかし、全文検索の結果にはユーザが意図し

ないエントリも多く含まれる(ノイズが多い)ため、グループ化についてはあまり良い結果が得られなかった。

- ホモロジー検索の結果として得られるエントリ集合を要約した場合、指定したエントリ集合に特有な(特殊性が高い)アイテムが重要語として抽出された。また、エントリ集合内にはっきりとしたグループ化が認められた。

与えられたエントリ集合内に多く出現する重要語の抽出は行えた。また、重要語と判断されたアイテムは、キーワード検索の検索語と関連がある用語が結果として現れていることから、望ましい重要語抽出ができているといえる。このことから、重要語を確認することで、エントリ集合の内容を推測できるといえる。また、ホモロジー検索やモチーフ検索などのように、よくまとまった(ノイズが少ない)エントリ集合を本システムに入力すれば、特殊性の高い重要語を抽出し、有用なグループを発見できる可能性が高いことが示唆された。

全文検索とホモロジーの例は、特定のトピック(検索語や質問配列で表現されたユーザの興味)に関して、それぞれノイズが多いエントリ集合とノイズが少ないエントリ集合を要約した場合にシステムがどういう結果を返すかを示している。一方、エントリ集合内に複数のトピックが存在することが分かっている場合について調べるために、ENZYME データベースのエントリ ID を利用した実験を行った。ENZYME のエントリ ID は、EC 番号と呼ばれる酵素の機能分類に対応した４桁の数値で表されている。EC 番号を見れば、どのグループに属しているか判明する。そこで、EC 番号により細分類されたあるグループをエントリ集合として入力した場合、その分類に応じたグループが表示されるかどうかを調べた結果、以下のことが分かった。

- EC 番号の一部を揃えたエントリ集合を入力した場合、EC 番号の分類に応じたグループ化が認められる場合があった。さらに、EC 番号だけでは分からないサブグループが発見できる場合もあった。しかし、これらはどんな場合でも成立するわけではなく、良いグループ化が得られない場合もあった。

また、これら３種類の実験を通して、以下の知見を得た。

- 要約に使用するフィールドの選択により、表示される真理値表が異なるため、そこから得られるグループ化も異なってくる。これは、ユーザがどういう観点で要約を行いたいかにより、結果が違ってくることを示している。
- 要約を行うための基礎データとして十分なデータがない場合、要約結果が貧弱だったり、把握しにくいことが往々にしてある。
- 逆に、データが多すぎる場合、マイニングの結果多数の相関ルールが得られるので、リストや真理値表が長大になったり見にくくなったりする傾向がある。

7.2 今後の展望

現状の要約システム構築にあたり、明らかになった問題点と、現システムで改良及び追加を行う必要があると考えた点を以下に述べる。

- 自然言語形式で記述されたフィールドを使用した要約
  ゲノムデータベースの各エントリから列挙形式で記述されているフィールドを対象に、本システムは要約を行った。しかし、エントリには列挙形式以外にも自然言語形式、数値形式、配列形式など様々な形式で記述されているフィールドが存在する。このうち、自然言語形式で記述されているフィールドに対しては、次の処理を行うことで、本研究の場合と同様に要約を行える可能性がある。まず、形態素解析(形態素と呼ばれる意味を持つ最小の言語単位に分割し、形態素の構造上の意味や品詞の同定を行う)を行い、自然言語から形態素を抽出する。次に、抽出した形態素とゲノムデータベースの専門用語辞書(例えば本研究で使用した外延的オントロジー)を比較してアイテムの抽出を行う。自然言語から専門用語を抽出できれば、本システムと同じ手法で要約を行うことができる。その他のフィールドに関しては、要約の仕方そのものを別途検討しなければならないので、本研究の要約手法では処理が行えない。
- 他の検索サービスとの連携
  本システムでは、フォームから直接入力したエントリ集合に対して要約を行う。エントリ集合の入手には、他の検索サービスを利用する必要があるため、この

ままでは手軽に使えるサービスであるとは言い難い。そこで、結果としてエントリ集合を出力するような各種検索サービスと連動できれば、エントリを入力する手間が省けて使いやすくなる。

- 相関ルール発見の組み合わせ問題

  本システムでは、計算資源の限界により、膨大なエントリを持つデータベースの一部のフィールド(GenBank:gene，GenBank:organism，GenBank:strain，LITDB:KEYWORD，REFSEQ:gene，REFSEQ:product)に対しては、相関ルールの最大アイテム数 1 個でしか処理を行っていない。最大アイテム数が 1 個であるから、重要アイテムセットの要約結果では、アイテム単体の発見しかできない。前処理に用いる計算アルゴリズムの改良により、この問題を解決する必要がある。

謝辞

本研究を進めるに当たり、終始暖かく御指導戴きました北陸先端科学技術大学院大学 知識科学研究科 遺伝子知識システム論講座 佐藤 賢二助教授に心から御礼申し上げます。また、様々な面で御助言戴きました北陸先端科学技術大学院大学 知識科学研究科 遺伝子知識システム論講座 小長谷 明彦教授に心から御礼申し上げます。副テーマにおいて、本研究と関連した分析手法を御教授戴きました、北陸先端科学技術大学院大学 知識科学研究科 複雑系解析論講座 中森 義輝教授に心から御礼申し上げます。また、北陸先端科学技技術大学院大学 知識科学研究科 遺伝子知識システム論講座の助手である、Xavier Defago 先生、山本 知幸先生には、様々な側面から御助力戴き心から御礼申し上げます。

本研究で様々な側面から御助力下さいました佐藤研究室、小長谷研究室の皆様に厚く感謝致します。最後に、多くの方々の暖かい御協力により本研究を行うことが出来た事を感謝致します。

参考文献・参考 Web

[1] 高木利久, 金久實 : ゲノムデータベースの利用法[第 2 版], 共立出版(株), 1998 年 4 月.

[2] GenomeNet(ゲノムネット WWW サーバ 京都大学化学研究所バイオインフォマティクスセンター) : http://www.genome.ad.jp/Japanese/

[3] 山名早人・近藤秀和(早稲田大学 理工学部 情報学科) : サーチエンジン Google, 情報処理, 42 巻 8 号, pp775-780, 2001 年 8 月.

[4] 長尾真, 佐藤理史, 黒橋禎夫, 角田達彦 : 岩波出版ソフトウェア科学 15 自然言語処理, (株)岩波出版, 1996 年 4 月.

[5] 野口正一, 牧野武則 : com シリーズ 図解 自然言語処理, (株)オーム社, 1991 年 5 月.

[6] 奥村学, 難波英嗣(北陸先端科学技術大学院大学 情報科学研究科) : テキスト自動要約に関する研究動向(巻頭言に代えて), 1998 年 11 月

[7] 奥村学, 難波英嗣(北陸先端科学技術大学院大学 情報科学研究科) : テキスト自動要約に関する研究動向(改訂), 1999 年 4 月

[8] 奥村学(東京工業大学), 難波英嗣(北陸先端科学技術大学院大学) : テキスト自動要約に関する最近の話題, 2000 年 7 月.

[9] 市村由美, 長谷川隆明, 渡部勇, 佐藤光弘 : テキストマイニング事例紹介, 16 巻 2 号, pp192-200, 2001 年 3 月.

[10] 那須川哲哉, 河野浩之, 有村博紀 : テキストマイニング基盤技術, 人工知能学会誌, 16 巻 2 号, pp201-211, 2001 年 3 月.

[11] 布施田敏樹(北陸先端科学技術大学院大学 知識科学研究科 知識システム基礎学専攻)：ゲノムデータベースにおける柔軟なデータ加工及びマイニングシステムの構築に関する研究，2000 年 2 月.

[12] 月本洋：実践データマイニング 金融・競馬予想の科学，(株)オーム社，1999 年 12 月.

[13] 内藤隆宏(北陸先端科学技術大学院大学 知識科学研究科 知識システム基礎学専攻)：マイクロアレイにより得られる遺伝子発現情報からの知識発見に関する研究，2000 年 3 月.

[14] 花沢祐二，力竹尚子：(特集２)本番！データマイニング 顧客攻略の法則が見える，日経情報ストテラジー，pp46-59，No99，2000 年 7 月.

[15] データマイニング法 2001 春学期，古川康一(慶應義塾大学 環境情報学部)：http://bruch.sfc.keio.ac.jp/course/DM01/

[16] 片岡孝雄(北陸先端科学技術大学院大学 知識科学研究科 知識システム基礎学専攻)：ベクトル空間法を用いてゲノムデータベース全体から関連性を抽出する手法に関する研究，2000 年 3 月.

[17] EleanorLawrence，荒木忠雄，清水碩，藤森高嶺：ヘンダーソン生物学用語辞典，(株)オーム社，1996 年 5 月.

[18] 栗長醤油 酵素入門：http://www.awacho.co.jp/main/0502micro.htm

[19] アミノ酸の代謝 尿素回路(福岡大学 理学部 化学科 生物化学第１研究室)：http://www.sc.fukuoka-u.ac.jp/~bc1/Biochem/Ureacycl.htm