平成 11 年度

筑波大学第三学群情報学類

卒業研究論文

題目：GUI を用いた
サーチエンジンのインタフェースに関する研究

主専攻	情報科学
著者名	八木 豊志樹
指導教員	電子・情報工学系 田中 二郎

要旨

既存のキーワード検索型サーチエンジンでは、キーボードとマウスと言う二つのデバイスを使い分ける必要である。また、ディレクトリ検索型サーチエンジンでは、検索できるページ数に限りがあり、膨張する Web の速さに追従しきれない。

そこで本論文ではそれに代わり得る、図形入力によるキーワード検索型サーチエンジンのフロントエンドの考案をした。それは、マウスのドラッグアンドドロップによって、検索式を作るというアプローチを取っている。これにより、ユーザはマウスのみでキーワード検索が出来るようになる。

目次

第 1 章

はじめに

現在存在する Web ページは国内だけでも 2950 万ページとも言われている [10]。その中から必要な情報を取り出すのは非常に大変な作業である。

その中で誕生、進化していったのがサーチエンジンである。サーチエンジンは、人が目的とするページを探す手助けとなっている。今では、サーチエンジンは WWW から必要な情報を取り出すのに必要不可欠な存在になっている。

しかし、そのインタフェースは必ずしも使いやすい物ではない。それは、マウスとキーボードという二つのデバイスの間を交互に使い分けなければならないからである。

そこで、本論文ではマウスを使った図形入力によるインタフェースを考案する。

第 2 章では既存のサーチエンジンについて述べ、第 3 章では関連のあるサーチエンジン Verno について述べ、第 4 章では図形入力によるサーチエンジンへのインタフェースについて述べ、第 5 章でまとめる。

第 2 章

既存のサーチエンジン

既存のサーチエンジンには、大きく分けてキーワード選択型サーチエンジンと、ディレクトリ型サーチエンジンがある。ここではこれらの特徴と問題点について考察する。

2.1 キーワード検索型サーチエンジン

キーワード検索型サーチエンジンとは、 Goo[3] や AltaVista[4] に代表されるサーチエンジンである。

このタイプのサーチエンジンでは、通常は、文字をキーボードから入力して検索を進める。また、このタイプのサーチエンジンでは、検索対象を大量にできるというメリットがある。また、作成日時などの付加情報で検索する事も出来る。

それは、エージェント、ロボット等と呼ばれるプログラムで自動的にデータを回収できるためである。ロボットによって回収された HTML ファイルは、インデクサと呼ばれるプログラムによってデータベースに置かれる。ユーザはインデックス化されたデータベースを間接的に扱う事によって、目的とするページを探す事になる。

しかし、このタイプのサーチエンジンでは、キーボードとマウスの操作を使い分ける為、ユーザに負担がかかる。また、検索式が複雑になれば、読みづらくなるという欠点もある。

2.1.1 事例: Goo

Goo では、検索条件として「全ての語を含む (AND)」、「いずれかの語を含む (OR)」、「フレーズ」、「リンク先 URL」、「人名」、「Boolean」のうちいずれかを選択し、検索を行う。フレーズ検索では、入力したキーワードの語順を守って、言葉と言葉の間に「；　：　、　・」等の記号が間に入っていても検索対象にする。また、リンク先 URL 検索は、入力した URL にリンクを張ってあるページを検索する。 Boolean 検索では、キーワード入力スペース内で AND、 OR、 NOT 検索ができる。例えば、インターネット AND (入門 OR 初心者) NOT セミナーとすると、インターネットの入門的ページを探しているがセミナー関係のページは除外したい、という場合役に立つ。

図 2.1: Goo で「筑波大学」をサーチした結果

図 2.1が、 Goo で「筑波大学」をサーチした結果である。

キーワード欄に「筑波大学」と入力し、「検索」ボタンを押す事でサーチが開始される。この場合、フォーカスをキーワード欄に合わせ、キーボードを使い文字を入力し、マウスで「検索」ボタンを押す事になり、面倒である。

goo の検索結果は、適合順、ドメイン順、日付順でソートできるようになっている。

goo では、検索結果を絞り込む事が出来る。図 2.1の「さらに条件を絞り込む」のリンクを辿ると、図 2.2の画面に移って、検索結果を絞り込む。追加キーワード、日付指定、検索先指定、データタイプで絞り込む事が可能となっている。追加キーワードでは、キーワード等を追加して絞り込み追加検索が出来る。日付指定では、ページが作成された日時で検索をする事が出来る。検索先設定の画面では、作成された地域やドメイン名で検索を絞り込む事も出来る。データタイプ指定では、 Image、 Video、 Audio などのデータタイプを指定する事が出来る。

2.1.2 事例:AltaVista

図 2.3が、 AltaVista の Advanced Search の画面である。

ユーザは、`(peanut AND butter) AND (jelly OR jam)` や `((marketing OR sales) AND business) AND recruit` のような複雑な検索式を入力する事が出来、`Language:` の欄から言語を選ぶ事によって、ページがどの言語で書かれているかを選ぶ事が出来る。また、`From:` 、`To:` の欄に日付を書き込む事によって、作成した日時による絞り込も可能になっている。しかし、検索式が複雑になると読みづらくなる。

AltaVista では、結果がヨーロッパ主要言語のページであれば、英語に翻訳するサー

図 2.2: goo の絞り込み画面 (EXPERT goo)

図 2.3: AltaVista の Advanced Search

図 2.4: Google のメインページ

ビスを行っている。また、結果ページに関連するページを再検索するサービスを行っている。

但し、関連ページを再検索するサービスは Advanced Search には無い。

2.1.3 事例:Google

図 2.4が Google[5] のトップページである。このサーチエンジンで目新しいのは “I’m feeling lucky” ボタンである。このボタンを押すと、検索式にとって最も適切であろうページにジャンプする。 Google では、検索キーワードに対するページの適合度に加え、リンク元ページの適合度も計算してスコアリングを行っている。そのように計算されたスコアリングの最も高いページに、 “I’m feeling lucky” ボタンでジャンプするものと考えられる。

Google のランキング方式は、質問とは独立していて、被リンク数の多さ、重要なページからのリンクの多さを用いている。検索キーワードが与えられた時は、そのキーワードを含むページを探し、既に決められたランキングによってそれらのページを並べている。そのため、応答が早い。

2.2 ディレクトリ型サーチエンジン

ディレクトリ型サーチエンジンとは、 Yahoo! JAPAN[7] や iNET Guide[8] に代表されるサーチエンジンである。また、 AltaVista、 NETPKAZA[6] 等のキーワード検索型エンジンにも付加される事が多くなった。

図 2.5: Yahoo!Japan のカテゴリ選択画面

このタイプのサーチエンジンでは、ユーザは漠然としたイメージから、マウス操作のみで操作できるというメリットがある。また、質の高い情報が得られるというのもディレクトリ型サーチエンジンの特徴である。

しかし、人手でデータ整理を行わなければならない為、見つかる情報量は限られている。

2.2.1 事例:Yahoo! JAPAN

図 2.5が Yahoo!JAPAN の「芸術と人文」というカテゴリを選んだ画面である。そしてそこからパフォーミングアート、パフォーマー、パフォーミングアーティストの順にカテゴリをたどって行くと図 2.6のようになる。このように、ユーザは漠然としたイメージからマウスのみで操作できる。また、キーワード入力型サーチエンジンのように、カテゴリやページを検索できるようになっている。

しかし、情報が古かったり、探していた物が無い場合もある。

図 2.6: Yahoo! Japan のパフォーマンスアーティストカテゴリ

第 3 章

全文検索エンジン Verno

Verno[1] は、早稲田大学で研究目的に開発されたサーチエンジンである。 Verno は、データベース層、ネットワーク層、インタフェース層の 3 層からなる。 3 層に分かれている事によって、データベースやインタフェースの追加や変更が容易になっている。本論文では、この特徴を生かして、新たなインタフェースを構築する。

3.1 データベース層

データベース層は、

- インデックスデータベース (通常の文字列検索等)
- プロパティデータベース (HTML ファイルのサイズや作成日、ドメインなどテキスト部分には現れない特徴に関するデータベース)
- タグデータベース (タグとして記述されている物に関するデータベース)

の 3 種類が用意されていて、検索すると URL が得られる。この特徴に着目し、検索をファイルのアクセスと同様に Open、 Read、 Close の 3 つに分けて行う。

3.2 ネットワーク層

ネットワーク層はデータベース層とインタフェース層の架け橋である。データベースの API に近い形で要求を与える事を理想とし、ネットワークのサーバとアクセスするための言語をもつ。その言語を検索言語と呼び、この検索言語を扱うサーバを検索言語サーバと呼ぶ。検索言語は、簡単なプログラミング言語であり、またネットワークプロトコルでもある。

ネットワーク層の存在により、データベース開発者は検索言語だけを意識して検索ルーチンを実装すれば検索可能となる利点がある。また、ユーザから検索言語を見ると、検索言語を介してデータベースとインタフェースを扱う事により、 Web 検索を実装する事ができる。

3.3 インタフェース層

インタフェース層は、ユーザがサーチエンジンにアクセスする架け橋である。 Web 上の CGI や、本論文で扱うフロントエンドもインタフェース層に属する。

3.4 検索言語サーバ

検索言語サーバとは、インタフェース層とデータベース層のつなぎを行うサーバである。検索言語サーバはネットワーク層に属する。

インタフェース層から検索要求が出されると、検索言語サーバは、検索を行うためのプロセスを 1 つ起動する。これにより、各種検索処理が並行して行われる。

インタフェース層からの検索言語を用いて記述された検索条件式をデータベースの API に見合うように処理し、データベースによる検索が必要な分だけを Open を用いて必要な分だけを各データベースに検索要求を出し、 DB ハンドルを得る。

次に、論理演算などで DB ハンドル間の演算を行う。この時点では Read は行われず、また論理演算も行わない (遅延評価)。

最後に、検索結果の表示の要求など、検索結果が実際に必要になった時にはじめて検索言語サーバは各 DB ハンドルに対し Read を行い検索を行わせる。また得られた検索結果により、論理演算を施し結果をインタフェースに返す。

また、 Verno には Scheme を利用した検索言語も存在する。それは、記述性や柔軟性を考えると、 Lisp の S 式のように、任意の入れ子構造が指定でき、プログラムとデータが同一形式をもつ者が有利だからである。

3.5 システムの並列化

Verno では、全体のデータを分割してそれぞれの PE に分散配置し、検索の際には各 PE による検索結果を統合して出力する並列化方式を取っている。データを分散させてそれぞれのデータベースを作成する事により、作成時間の短縮が見込め、データの冗長性も小さくなる。

また、 Verno ではシステム全体の検索結果を得るために必要となる、各 PE における検索結果の統合は、各インタフェースによって個別に行うものとしている。ネットワーク層が、データベース層の並列実装を隠蔽してしまい、検索言語のプログラマに並列性を意識させない、という事は行っていない。

3.6 インデックスデータベース

Verno におけるインデックスデータベースとは、検索に関る条件をいれるとその検索条件を満たす URL の集合を返すと言う物であり、システムのデータベース層の核をなす物である。

Verno のインデックスデータベースの方針は以下の通りである。

- 3.1で規定したデータベースの API に準拠する
- 論理演算などは言語サーバで行うため、データベースでは 1 度の検索では 1 つのキーワードのみに対して処理すれば良い
- データベース部分では分散されている事は意識しなくても良い
- どんな文字列であっても検索できるようにする。文の一部や、一般には検索できないような語の一部といったようなフレーズでない物であっても検索可能にする

3.7 検索方式

Verno が採用している検索方式は、転置索引方式である。これは他の方式に比べデータベース化されている単語の検索が速くなるためである。 Verno は複雑な検索を行おうとしているのであり、基本的な検索ではなるべく高速である事を必要としているためである。

Verno では、 N=1 である N-gram 方式で単語を扱っている。これは新語や未定義語の対応が用意であり、複合名詞などの扱いは考えなくても良く、部分一致検索などの実装が用意であるためである。しかし、この方法では検索に膨大な時間がかかるため、全角では出現頻度の高い平仮名や片仮名においては N=3、それ以外の場合では N=2 とした。また半角においても形態素解析をせずに語尾辺かも考慮せず、語頭語尾が英数字であり、それ以外が英数字あるいは ‘-’ ‘=’ ‘/’ ‘.’ のいずれかである文字列を単語とした。

単語はそのままでは扱いにくいので、全角単語については全種類に一意の 32 ビットの値 (wordID) をつけて扱いやすくしている。半角の単語の場合、 32 ビットでは一意の番号を付すことができないため、単語の MD5 の値のうち、上位 32 ビットを wordID として用い、下位 96 ビットはハッシュ表におけるハッシュ値を求めるために用いる。

3.8 検索の流れ

インデックスデータベースは、ハッシュ表、データ参照ファイル、頻度データファイル、位置情報ファイルの 4 つから構成されている。

ある一単語を検索する流れは、

- 単語を wordID に変換する
- wordID をハッシュにかける
- ハッシュ表からデータ参照ファイルの該当位置をひく
- データ参照ファイルから頻度データファイル、位置情報ファイルの該当位置をひく

- データ参照ファイルから出現ファイル数を、頻度データファイルから URL No. 頻度を、位置情報ファイルからは位置情報を利用する

となっている。

また、複数の単語からなるキーワードを検索する時は、

- キーワードを構成する単語それぞれに対して、データ参照ファイルにあるデータを取得する
- 各単語の出現ファイル数を比較し、最小である単語の頻度データおよび位置情報を検索候補として主記憶上に置く
- 出現ファイル数が最小ではない残りの単語の打ち、キーワードの文字を全て含み、かつ重複が無いように絞り込用の単語を選ぶ
- 絞り込用の単語の頻度情報ファイルからの pageID および位置情報ファイルからの位置情報を用い、 pageID が一致しない、および位置情報の差分が一致しない検索候補を除外し、残ったものを検索結果として返す

という操作を行なっている。これを全走査と呼ぶ。

しかし、この方法では検索に時間がかかるため、絞り込に使う単語を少なくして、若干のノイズが入る事が考えられるがより高速な方法も提供されている。

第 4 章

検索式入力法の考察

本章では、上記をふまえ、新しい入力法を考察する。本論文では、なるべくマウスのみで操作するキーワード検索型サーチエンジンへのフロントエンドを考察する。

本研究では、キーワードを GUI 部品に置き換え、それをクリックして検索式を作る事に主眼を置いている。そこで、図 4.1の様なシステム、 Visual Search System(VSS) を考案した。

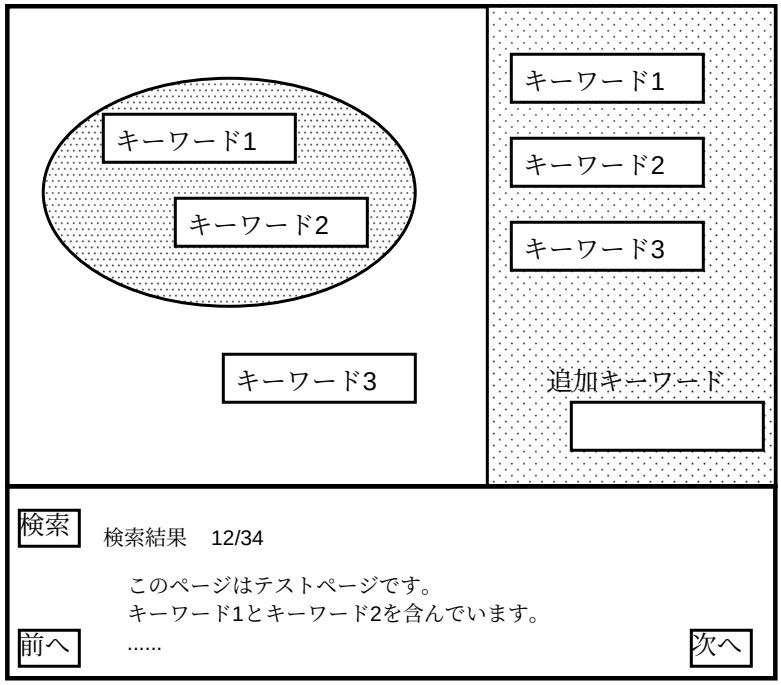

図 4.1: 今回考案したシステム VSS の概要

左側は、検索式作成部であり、ここで実際の検索式を作る。

右側は、キーワードプロフィール部であり、ここから検索式作成部にキーワードをドラッグアンドドロップして検索式を作る。

下は、検索結果表示部であり、検索式に当てはまる URL 等を表示させる。この部分に検索ボタンを設け、実際の検索を行なう。

4.1 検索式の作成

検索式は、キーワードプロフィール部から検索式作成部にキーワードをドラッグアンドドロップして作られる。また、範囲指定をし、グルーピングする事によってより複雑な検索式を作成する。図 4.1では、キーワード 1 とキーワード 2 が範囲指定されグルーピングされている。このようにして、キーワード 1 とキーワード 2 が AND で結ばれているとする。よって、図 4.1においての検索式は (キーワード 1 AND キーワード 2) OR キーワード 3 となる。

グルーピングには AND グルーピングと OR グルーピングを用意した。検索式作成部の最外角は OR グルーピングされているとみなす。

検索式作成部の、キーワードでない部分をドラッグすると、グルーピングを行なう。マウスの左ボタンでドラッグすると AND グルーピング、右ボタンでドラッグすると OR グルーピングされる。図 4.2は、 (キーワード 1 AND キーワード 2 AND (キーワード 3 OR キーワード 4)) を表す。

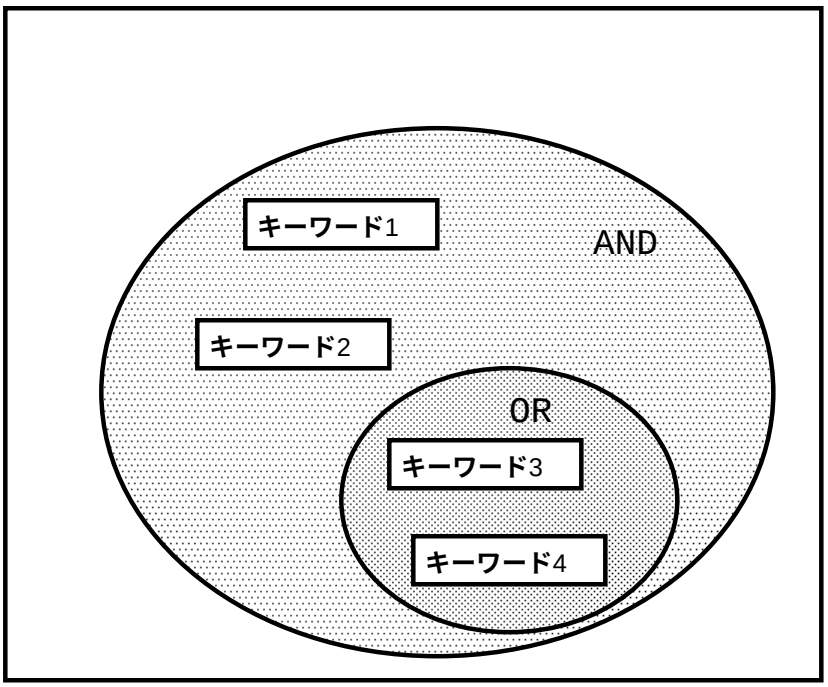

図 4.2: キーワード入力の例 1

また、図 4.3は、 (キーワード 1 AND キーワード 2) OR (キーワード 3 AND キーワード 4) を表す。

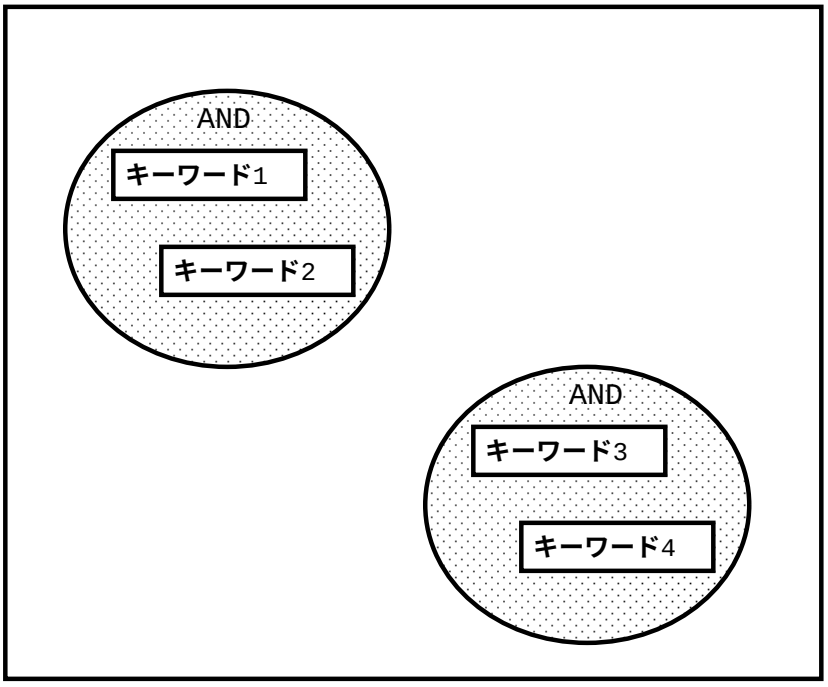

図 4.3: キーワード入力の例 2

キーワードは、ドラッグする事によって動かす事が出来る。また、キーワードを右クリックする事によって、キーワードの属性を NOT にしたりキーワードを削除する事も出来る。

このようにして、ユーザは検索式を作る。

4.2 キーワードプロフィール部

検索式作成部にキーワードを与える重要な役割を担っているのが図 4.1の右側にあるキーワードプロフィール部である。

ここから検索式入力部にドラッグアンドドロップして、ユーザは検索式を作っていく。

キーワードプロフィール部には、過去に検索する単語として使った物が蓄えられる。また、検索結果から抽出されたキーワードもここに蓄えられる。

ここには「追加キーワード」の欄があるが、これは実際に使いたいキーワードがキーワードプロフィール部に無い時に使用する。この部分にキーワードを入力すると、キーワードプロフィール部に新たなキーワードが登録される。

4.3 検索結果表示部

図 4.1の下側にあたる検索結果表示部は、ユーザの入力した検索式に対する結果を表示する部分である。

ここでは、結果は 1 つずつ表示される。

4.1節のようにして検索式を作成し、「検索」ボタンを押すと検索が始まる。

「前へ」のボタンを押すと前の候補に戻り、「次へ」のボタンを押すと次の候補に行く。

URL 部分をクリックすると、ブラウザにその情報が渡され、またキーワードプロフィール部に渡すためのキーワードを作るために自分自身も URL の先のファイルを読みに行く。

4.4 検索の流れ

キーワードプロフィール部からドラッグアンドドロップしたキーワードによって作成された検索式は、パーザにかけられ、 Verno の検索言語に直される。そして、 Verno の検索言語サーバにその検索式を渡し、結果を受け取る。受け取られた結果は、 1 件ずつ検索結果表示部に表示される。検索結果から、ユーザは自分に必要な情報を取り出す。すると、ブラウザに URL が渡され、システム自身もその URL の先を見に行く。システムは、見に行ったページからキーワードを抽出する。抽出されたキーワードは、キーワードプロフィール部に渡される。

この作業を繰り返す事によって、ユーザは目的とするページを探す。

4.5 検索の実例

ここでは、実際の検索について述べる。図 4.4は、検索を行う前のシステムの様子である。今回検索する以前に、 VSS において Linux と ThinkPad というキーワードを用いていたため、キーワードプロフィール部には Linux と ThinkPad というキーワードが存在する。ここで、 Linux をキーワードプロフィール部から検索式入力部にドラッグアンドドロップすると、図 4.5のようになる。そして、 ThinkPad を同じようにドラッグアンドドロップすると、図 4.6のようになる。今回は AND 検索をしたいので、左ボタンで、検索式入力部の 2 つのキーワードを囲むようにドラッグして、 2 つのキーワードを囲むと図 4.7のようになる。これで、 Linux と ThinkPad は AND で結ばれる。そして、検索結果表示部にある検索ボタンを押すと、検索がはじまり、図 4.8のように結果が帰って来る。結果のタイトル「Index of /WWW Sample With Access-Counter」をクリックすると、ブラウザに URL が渡される。また、 VSS 自身も URL の先のファイルを読み込み、キーワードを抽出する。そして、抽出されたキーワード「FreeBSD」、「Administration」は、図 4.9のようにキーワードプロフィール部に渡される。

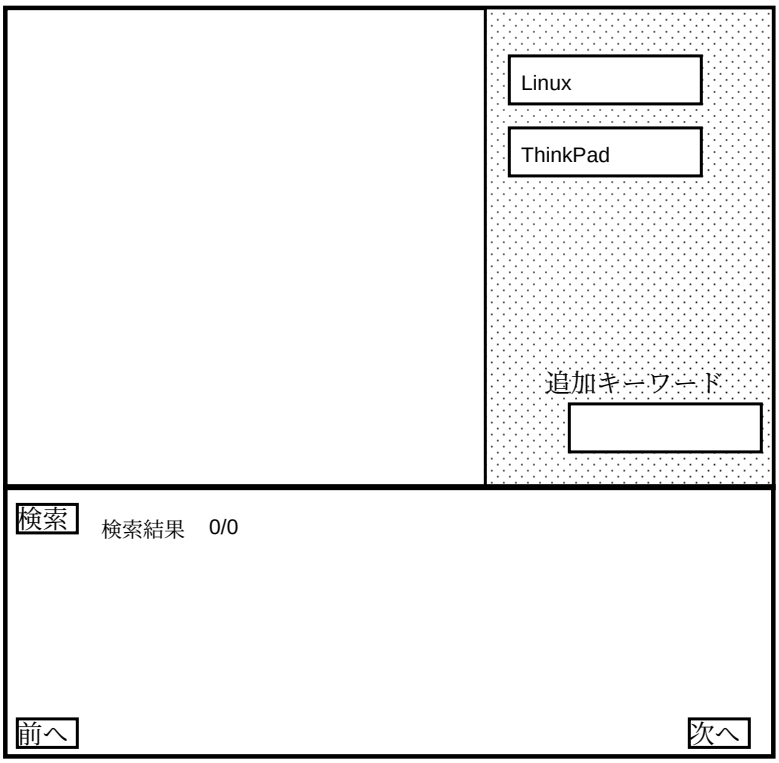

図 4.4: 初期画面

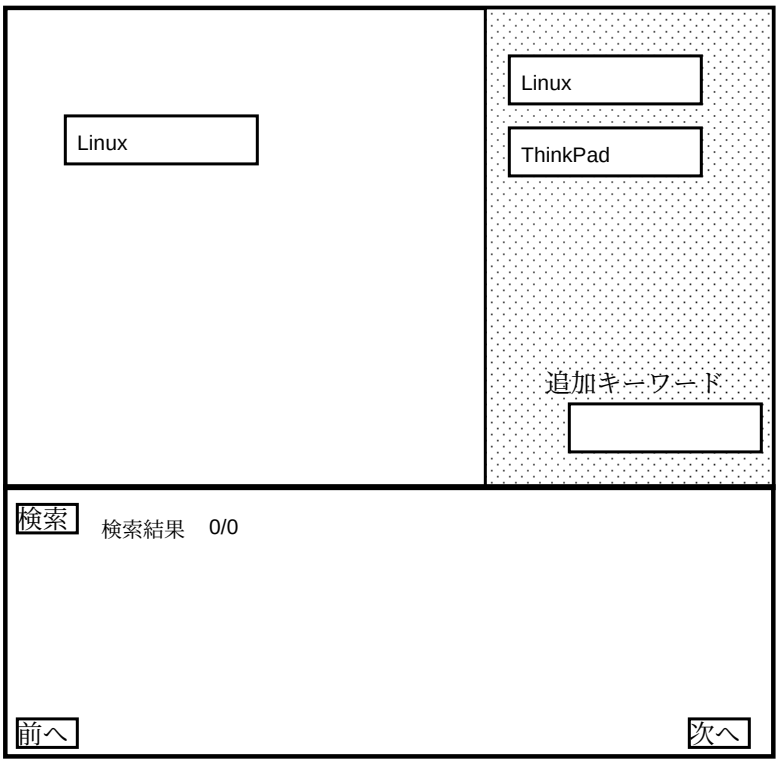

図 4.5: 「Linux」を検索式入力部に入力

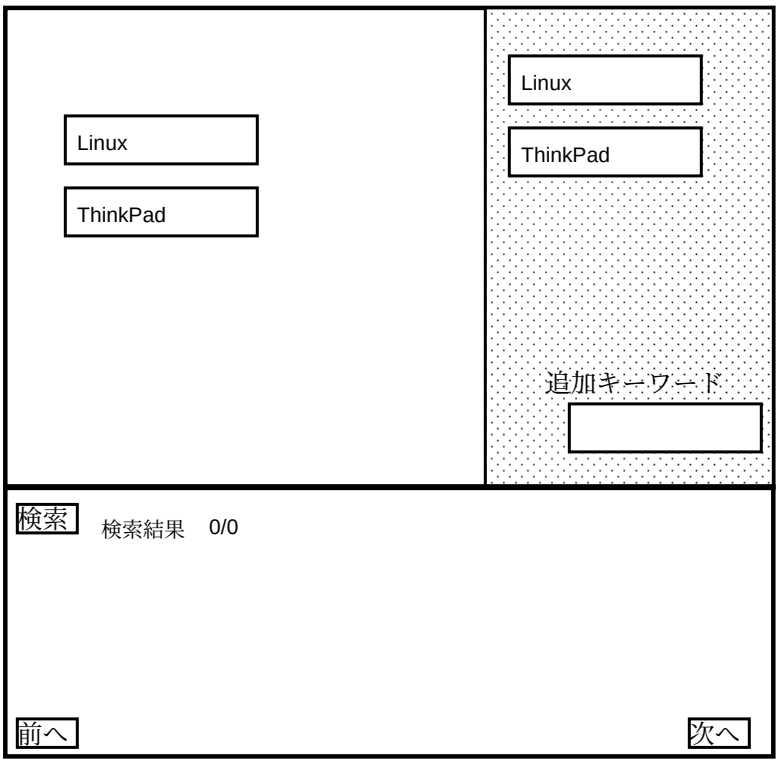

図 4.6: 「ThinkPad」を検索式入力部に入力

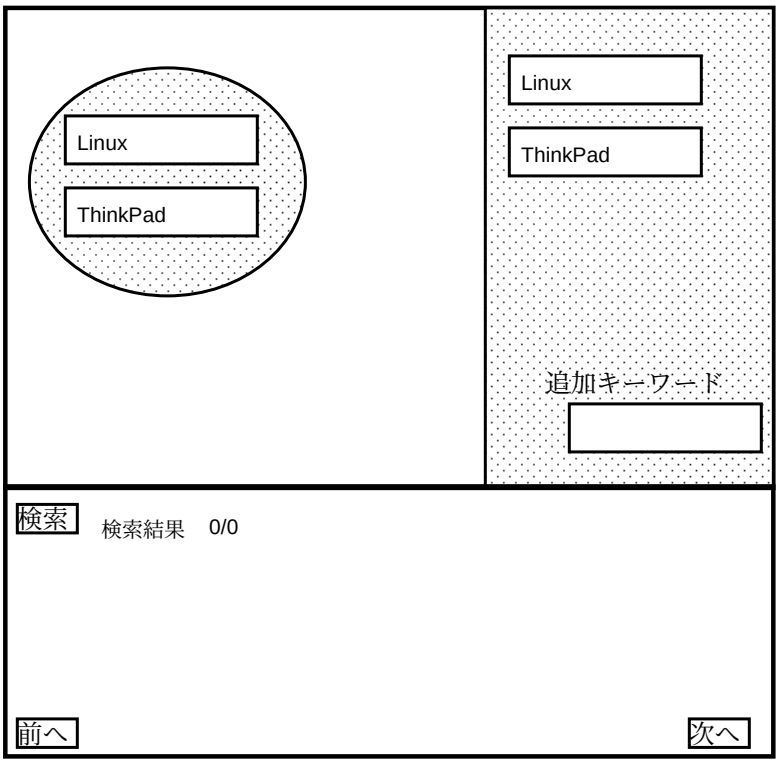

図 4.7: 単語を AND で結ぶ

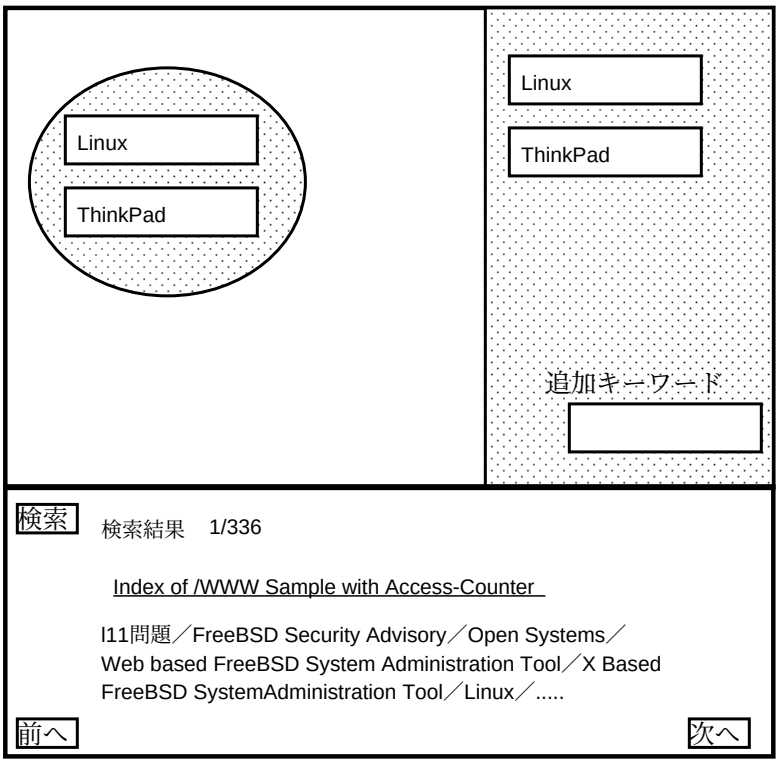
Linux
ThinkPad
Linux
ThinkPad
追加キーワード
検索
検索結果　1/336
Index of /WWW Sample with Access-Counter
l11問題／FreeBSD Security Advisory／Open Systems／
Web based FreeBSD System Administration Tool／X Based
FreeBSD SystemAdministration Tool／Linux／.....
前へ
次へ

図 4.8: 検索の実行

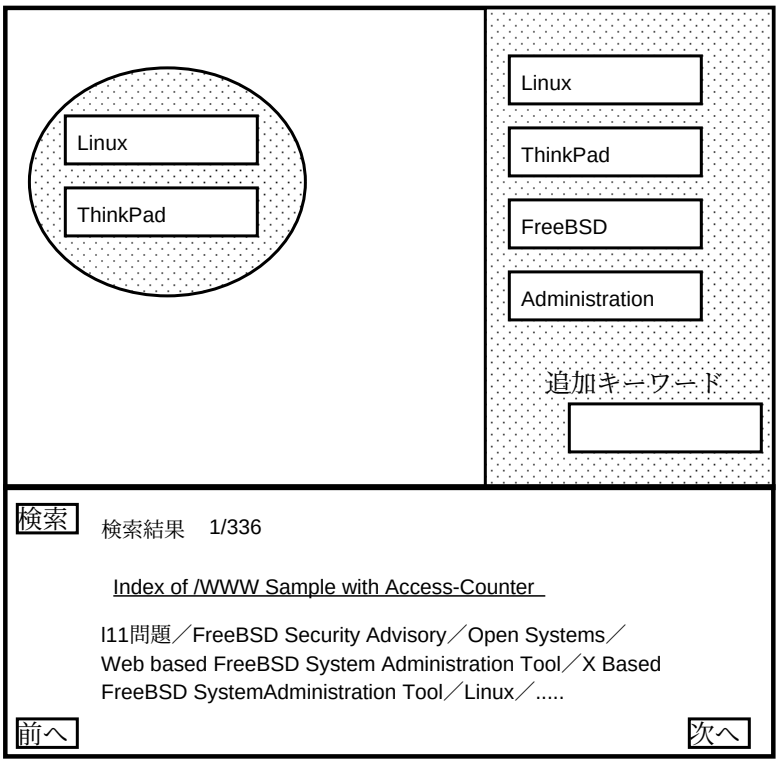

図 4.9: キーワードの抽出

第 5 章

結論

既存のキーワード検索型サーチエンジンでは、キーボードとマウスと言う二つのデバイスを使い分ける必要である。また、ディレクトリ検索型サーチエンジンでは、検索できるページ数に限りがあり、膨張する Web の速さに追従しきれない。

そこで本論文ではそれに代わり得る、図形入力によるキーワード検索型サーチエンジンのフロントエンドの考案をした。それは、マウスのドラッグアンドドロップによって、検索式を作るというアプローチを取っている。これにより、ユーザはマウスのみでキーワード検索が出来るようになる。

この研究は未だ構想の域を脱してはいないが、以後実装を進め、評価をする事によって、その有用性を確かめる予定である。

謝辞

この研究を進めるにあたり、終始ご指導下さった指導教官の田中二郎教授に心から感謝致します。また、忙しい中いろいろな御助言を頂いた田中研究室の皆様、特に奥村穂高氏に感謝致します。

参考文献

[1] 沼尻務, 竹岡厚, 渡辺高志, 芦川将之, 上田和紀: 全文検索システム Verno のアーキテクチャの設計, 第 2 回インターネットテクノロジーワークショップ論文集, 日本ソフトウェア科学会研究会資料シリーズ No.13, ISSN 1341-870X,5-1, pp.163–170(1999).

[2] 奥村穂高, 田中二郎: 重ねあわせノードによる Web のキーワード検索, 情報処理学会第 60 回大会,4U-02(掲載予定)(2000).

[3] Goo.
http://www.goo.ne.jp/

[4] AltaVista.
http://www.altavista.com/

[5] Google.
http://www.google.com/

[6] NETPLAZA.
http://netplaza.biglobe.ne.jp/

[7] Yahoo! JAPAN.
http://www.yahoo.co.jp/

[8] iNET Guide.
http://www.integ.com/

[9] 田中二郎, 神田陽治編: インタフェース大作戦, 共立出版, 1995.

[10] 平成 11 年度通信白書
http://www.mpt.go.jp/policyreports/japanese/papers/99wp/99wp-0-index.html