情報処理学会論文誌　データベース　Vol. 3　No. 4　48–64 (Dec. 2010)

RerankEverything：ランキング結果閲覧のための柔軟な再ランキングインタフェース

山本岳洋[†1,†2]　中村聡史[†1]　田中克己[†1]

本稿では，さまざまなランキング結果をユーザのインタラクションに応じて自由に再ランキングできるシステム RerankEverything を提案する．既存の検索エンジンやウェブサービスでは，サービス側が用意した機能を利用することでしか，ランキング結果を変更することができない．RerankEverything では，ユーザはランキング結果に対して「この単語を含む結果を上位・下位に再ランキングしたい」であるとか，「この尺度でランキング結果を並べ替えたい」といった意図を簡単なインタラクションを介して伝えることができ，自らの興味や嗜好に合わせてランキング結果を再ランキングすることができる．また，本システムを利用することで，ユーザはウェブ検索結果だけでなく，動画・商品・ホテル・ニュースなどさまざまなランキング結果を再ランキングできるようになる．本稿ではシステムを（1）直接操作による再ランキングインタフェース，（2）タームクラウドによるランキング結果閲覧支援，（3）パーサ作成インタフェースと属性値の動的抽出，という 3 つの観点から設計し，ユーザ実験によりシステムの有用性を確認した．

RerankEverything: A Reranking Interface for Browsing Ranked Results Flexibly

Takehiro Yamamoto,[†1,†2] Satoshi Nakamura[†1] and Katsumi Tanaka[†1]

In this paper we propose a system called RerankEverything, which enables users to rerank search results in any search service. In conventional search services, interactions between users and systems are quite limited and complicated. In addition, search functions and interactions to refine search results differ depending on the services. By using RerankEverything, users can interactively explore search results in accordance with their interests by reranking search results from various viewpoints. The reranking interaction in our system is simple and unified independently of the various search services. To realize our system, we design and implement the system from three aspects: (1) a reranking interface with direct interaction, (2) a TermCloud to encourage users to browse and rerank search results from various viewpoints, and (3) an interactive wrapper generation interface and a dynamic attribute extraction method. We evaluated the usefulness of our system by conducting the user studies.

1. はじめに

近年，些細な調べ事から，物品の購入，旅行の計画に至るまで，インターネットは情報取得の手段として欠かせないものとなりつつある．ユーザは WWW 上のコンテンツを広く網羅しているウェブ検索エンジンや，画像や動画コンテンツを対象としたマルチメディア検索エンジン，物品販売やホテル予約などに特化した各種検索サービスを利用することで情報収集を行ったり，物品の購入やホテルの予約などをしたりする．一般にこうした検索サービスでは，ユーザの入力したクエリに基づき膨大なページや物品などをある尺度でランキングし，ランキング結果としてユーザに提示する．

しかし，下記に示す理由から，ユーザは満足のいく情報をランキング結果から得られないことが多々ある．

- ユーザにとって自らの求める情報をうまく表したクエリを作成することは容易ではない．また，初めから求める情報が具体的に決まっていないことも多く，ユーザが初めに入力するクエリは短く曖昧なものであることが多い[1),2)]．そのため，サービス側がクエリのみからユーザの検索意図を推定し，すべてのユーザが満足するランキング結果を返すことは困難である．
- 検索サービスによっては，ユーザ側でランキング結果を改善できるように，ランキング結果の絞り込みや，特定の属性を指定したソートといった機能を提供している．しかし，そうした機能は検索サービス側で用意されたものであるため，ユーザがある属性でランキング結果をソートしたいと考えたとしても，そのための機能がサービス側で用意されていなければ，ユーザはその属性でランキング結果を並べ替えることはできない．

このような問題点のため，現状ではユーザと検索サービス間のインタラクションは非常に限られている．ユーザの求める情報が，ランキング結果を閲覧中に「こんな結果をもっと見

†1 京都大学大学院情報学研究科社会情報学専攻
Department of Social Informatics, Graduate School of Informatics, Kyoto University
†2 日本学術振興会特別研究員（DC1）
JSPS Research Fellow (DC1)

てみたい」であるとか，「この観点で結果を並べ替えたい」というように移り変わっていったとしても，現状ではそうした意図をランキング結果に反映することは困難である．ユーザのその場の興味に応じてさまざまな観点からランキング結果を自由に並べ替えることができれば，より効率的に，また，より多くの観点からランキング結果中の情報を探索できると考えられる．

このようなシステムを実現するため，我々はこれまでにユーザインタラクションに基づいてウェブ検索結果を再ランキングするシステムを提案してきた[3]．しかし，このシステムは一般的なウェブページを対象としたウェブ検索結果のみを対象としており，商品検索や論文検索，ホテル検索といった，ウェブページ以外のランキング結果を対象に再ランキングを行うことができず，システムの汎用性に欠けていた．また，ランキング結果の再ランキング機能を利用するためには，我々が実装したウェブサービス⋆1自体にアクセスする必要があった．そのため，たとえば，あるユーザが Google のウェブ検索結果を閲覧中にその検索結果を再ランキングしたいと考えたとしても，ユーザは我々のウェブサービスにアクセスする必要があり，普段のウェブブラウジング中に出会うランキング結果ページ内で再ランキング機能を利用することができなかった．

そこで本稿では，ウェブ上で公開されているさまざまな検索サービスのランキング結果を自由に再ランキングできるシステム RerankEverything を提案する．提案システムは通常のウェブブラウザの拡張機能として実装されており，ユーザは日常のブラウジングにおいて本システムの機能を利用することができる．本システムを利用することで，ユーザはウェブ検索結果ページや商品のランキングページといったコンテンツの種類を気にすることなく，どのようなランキング結果に対しても再ランキングを行うことが可能となる．また，検索サービスにかかわらず統一的なインタフェースを利用して，ユーザ側でランキング結果を再ランキングすることができる．

これまで，ユーザはクエリを入力しランキング結果をただ受動的に受け入れることしかできなかったが，RerankEverything により，ランキング結果を自分の興味に合わせて再ランキングしていくことで，積極的に情報を探していくことが可能となると考えられる．

2. 基本的なアイデア

本研究の目的は，さまざまな検索サービスの検索結果を自由に再ランキング可能とするシステムを実現することである．本章ではまず，システムの実現に必要な課題を整理し，その後，実際のアプローチについて説明する．

2.1 技術的課題

現状では，ユーザはランキング結果を絞り込んだり並べ替えたりするために，検索サービス側が用意した機能やインタフェースを利用しなければならない．しかし，そうした機能を利用するには，わざわざウェブページを先頭や最後までスクロールしたり，あるいは新しいページを開いたりしてから，新しいクエリを入力し直したり，価格順や人気順といった並べ替えリンクをクリックしたりする必要があり，ユーザにとって負担が大きい．また，インタフェースが複雑なだけでなく，サービスごとにインタフェースが異なっている点も，ユーザを困惑させてしまうと考えられる．White らは，複雑なインタフェースや検索機能を活用できるユーザは限られているという報告している[4]．そのため，より直感的，かつ柔軟にユーザがランキング結果を改善できるように，シンプルで統一的なインタフェースが必要である．

また，さまざまなサービスが何千，何万という膨大なコンテンツをランキングしユーザに提示する一方で，現状ではユーザはランキング結果の上位数件しか閲覧しない[5]．そのため，多くのユーザは受け取ったランキング結果のごく一部の情報しか閲覧しておらず，ユーザの興味を引くであろう情報が下位にある場合，そうした情報と出会うことができない．この問題を解決するためには，ユーザに再ランキングインタフェースを提供するだけではなく，クエリ入力時には気づかなかった潜在的な興味を引き出し，ランキング結果をさまざまな観点から閲覧させるための仕組みが必要である．

さらに，そうした再ランキング機能をさまざまなランキング結果ページ上で適用するためには，検索サービスごとに，HTML ソースからランキング結果の構造を認識するための仕組みが必要である．しかし，ウェブ上に検索サービスは無数に存在し，その構造は検索サービスによって異なるため，完全に自動認識することは難しく，システム作成者があらかじめすべての検索サービスに対して構造認識のためのパーサを用意しておくことは非現実的である．そのため，ランキング結果の認識を行うためのパーサを，システムを利用するユーザ側で手軽に作成できる仕組みが必要となる．また，ユーザがランキング結果をどのような属性で再ランキングするかどうかは，その時その時の検索意図によって変わってくる．そこで，ユーザの多様な再ランキング意図を満たすことができる柔軟なデータ抽出手法も必要である．

以上の課題を解決するため，本研究では下記の 3 つのアプローチからシステムを実現した．

⋆1 Rerank.jp. http://rerank.jp

図 1　ランキング結果に直接働きかけることによる検索結果の再ランキング
Fig. 1　Reranking search results using direct interaction.

- 直接操作による再ランキングインタフェース
- タームクラウドによるランキング結果閲覧支援
- パーサ作成インタフェースと属性値の動的抽出

2.2　直接操作による再ランキングインタフェース

本システムでは，ユーザは“単語ベースのフィードバック”と“数値ベースのフィードバック”の 2 種類の操作により検索結果を再ランキングすることができる．ユーザは，ランキング結果の中から単語を選択し，単語ベースのフィードバックをシステムに伝えることで，「この単語を含んだ結果を上位に表示してほしい」であるとか，「この単語を含んだ結果は必要ない」といった意図をシステムに伝達し，ランキング結果を再ランキングすることができる．また，ランキング結果から属性値（価格，日付，時間など）を選択し，数値ベースのフィードバックをシステムに伝えることで，「ランキング結果を価格の順に並べ替えたい」といった意図でランキング結果を再ランキングすることができる．

本システムではユーザはこの 2 種類のフィードバックを，“ランキング結果から興味のある箇所を選択する”という同じ操作で行うことができる．**図 1** はユーザが単語ベースのフィードバック，数値ベースのフィードバックのそれぞれをシステムに与え，ランキング結果を再ランキングする様子である．たとえば，ユーザが静岡県の温泉に関するランキング結果を閲覧中に“熱海温泉”という単語に興味を持ち，“熱海温泉”に関連する結果を閲覧したいと思ったとする．すると，ユーザはランキング結果から“熱海温泉”という単語を選択する．その場合，システムはランキング結果を再ランキングするためのボタン（上昇ボタン・下降ボタン）をユーザに提示する．ユーザは上昇ボタンをクリックすることで，“熱海温泉”を含んだ検索結果を上位に再ランキングすることができる．同様に，ユーザがランキング結果を価格順で並べ替えたいと思った場合，ランキング結果から価格が記載されている箇所

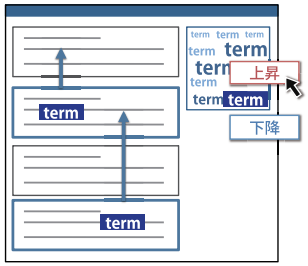

図 2　タームクラウドによる検索結果の再ランキング
Fig. 2　Reranking search results using terms in a TermCloud.

を選択する．すると，システムはランキング結果をソートするためのボタン（昇順ボタン・降順ボタン）をユーザに提示する．ユーザが昇順ボタンをクリックすることで，ランキング結果を価格の順にソートすることができる．このように，ユーザはランキング結果ページに対して直接インタラクションを行うことで，検索サービスにかかわらず同じインタフェースでランキング結果を自らの興味に応じて再ランキングすることができる．

2.3　タームクラウドによるランキング結果探索支援

ランキング結果をさまざまな観点から閲覧することを支援するために，本研究ではランキング結果中に出現する単語をいくつか抽出し，タグクラウド形式でユーザに提示する．本稿では，この提示方法をタームクラウドと呼ぶ．**図 2** に示すように，ユーザはタームクラウドに表示された単語を用いてランキング結果を再ランキングできる．

タグクラウドは，あるコンテンツ集合に付与されたさまざまなタグをひとかたまりの領域に並べて表示し，利用頻度や人気度といったタグの重要度に応じてフォントの大きさや色を変更することでコンテンツの可視化を行う手法である．タグクラウドを表示することの利点として，シンプルなリスト形式よりもタグクラウドの方が，ユーザの欲しい情報が明確に決まっていない際の情報探索を効率的に支援できるという点があげられている[6]．単語をユーザに明示的に提示することで，上位のランキング結果を閲覧するだけでは気付かなかった観点でランキング結果を閲覧することが可能となると考えられる．

2.4　パーサ作成インタフェースと動的な属性値抽出

本研究ではプログラミングの知識を必要とせず，簡単なマウス操作のみで視覚的にランキング結果ページのパーサを作成可能な機能を提案する．パーサを作成する一般的なアプローチは，HTML ソースからランキング結果のような目的の要素を抽出するスクリプトを記述

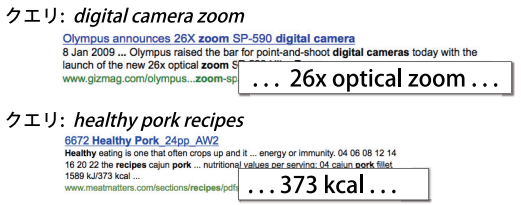

図 3　クエリに依存する属性値の例
Fig. 3　Numerical attributes depending on queries.

することである．しかし，一般のユーザにとって，スクリプト言語を用いてパーサを記述することは困難である．また，スクリプト言語に慣れたユーザであっても，それぞれのサービスごとに数行〜数十行のプログラミングを繰り返すことは負担である．一方，ページを閲覧しているユーザにとって，ページ中のどの部分がランキング結果にあたるかを判断することは比較的容易である．そこで，本システムではユーザのマウス操作に応じて，システムが今現在認識している構造情報をページ上に視覚的なフィードバックとして提示することで，パーサ作成を支援する．ユーザはマウスを動かしながらシステムからのフィードバックを得ることで正しいパーサを直感的に作成できると期待される．

また，ランキング結果をさまざまな属性で並べ替えるためには，ランキング結果中にどのような属性が存在し，それぞれの属性がランキング結果ページ中にどのような構造で出現しているかといった情報も必要である．一般的な構造抽出のアプローチでは，ユーザはまず目的のコンテンツのスキーマ（商品名，価格，メーカ名など）を決めておくことが多い．しかし，ランキング結果を再ランキングすることを考えた場合，あらかじめ目的のスキーマを決めておくことが困難な場合も多い．たとえば，**図 3** に示すように，あるユーザがウェブ検索エンジンを利用する場合を考える．あるときは “digital camera zoom” というクエリで検索結果を閲覧し，検索結果の中から “26x optical zoom” という文字列に興味を持ち，ズームの倍率に応じて検索結果をソートしたいと考えるかもしれない．またあるときは，“healthy pork recipes” というクエリで検索結果を閲覧し，“373 kcal” という文字列を発見し，カロリの少ない順に検索結果を閲覧したいと考えるかもしれない．このように，ユーザが興味を持つ属性値はその時その時の検索で変わるため，検索サービスのスキーマをあらかじめ決めておくことは困難である．

そこで，本研究ではこのような並べ替えを実現するために，ランキング結果中の属性値を，ユーザの再ランキング操作時に動的に抽出する．たとえば，ユーザが検索結果をカロリ順で選択したいと考え，“373 kcal” という文字列を選択すると，システムは自動的に他の検索結果から目的の属性値（この場合はカロリ）を抽出し，抽出された値に基づいて検索結果の並べ替えを行う．この手法により，ユーザはさまざまな属性値でランキング結果を並べ替えることが可能となると考えられる．また，パーサ作成時にランキング結果中の属性情報をあらかじめ指定しておく必要がなくなり，より手軽にパーサを作成できる．

3. 実　　装

我々は提案するシステムをウェブブラウザの 1 種である Firefox の拡張機能として実装した[*1]．システムをウェブブラウザの拡張機能として実装することで，ユーザは普段から使い慣れているウェブブラウザ上で本システムの機能を利用できるという大きな利点がある．

本章では，まずシステムの概要を述べた後，それぞれの機能の詳細について説明する．

3.1　システムの概要

図 4 にユーザが再ランキングを行うまでの流れを示す．ユーザがブラウザ上で新しいページにアクセスすると，システムはページの URL 情報を基に，システムに蓄えられたパーサ DB からそのページを解析可能なパーサを取得する．解析可能なパーサを見つけると，システムはパーサに登録されている XPath を用いて，ページ中からランキング結果を認識し，ランキング結果に対するユーザのマウス操作を監視する．同時に，システムはタームクラウドを生成し，ページにタームクラウドをオーバレイ表示する．

ユーザが検索結果から文字列を選択，もしくはタームクラウド中の単語をクリックすると，システムは選択された単語の周辺に再ランキング用のボタンをユーザに提示する．ユーザがそのボタンをクリックすることで，システムは検索結果を再ランキングし，再ランキング結果をユーザに提示する．

また，ランキング結果ページの構造を認識し終えると同時に，システムはページのリンクの先読みを行い，先読みをしたページからランキング結果を抽出し，現在閲覧中のページに自動的に追加する．リンクの先読みには，“次へ” や “Next” といったアンカーテキストの文字列情報を利用している．システムは，“次へ*”，“次の*”，“次を*”，“next”，“older” といった 5 種類の正規表現を用いて，先読みのためのリンクを抽出する．この手法はシンプルな経験則ではあるが，4.2 節の実験で述べる，システムが正しくパーサ作成可能な 33 件の検索サービスのうち，約 7 割の検索サービスに対して正しくリンクを先読みできる．この

⋆1 http://rerank.jp/everything/

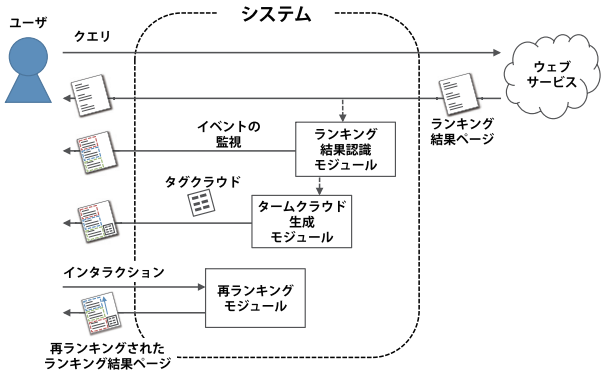

図 4　システム概要
Fig. 4　Overview of our system.

ようにして再ランキングの対象とするランキング結果数を増やすことで，ユーザが今閲覧しているページ中のランキング結果だけでなく，より多くのランキング結果を再ランキングの対象として情報を探すことができる．

3.2　ランキング結果の再ランキング

再ランキングの流れは以下のとおりである．

(1)　ユーザがランキング結果中の単語 t を選択すると，システムは単語 t が数値を含んでいるかどうかを 3.5 節の手法で確認する．単語 t が数値を含んでいる場合は，他の検索結果から数値を抽出する．

(2)　システムはユーザにランキング結果を再ランキングするためのボタンを提示する．単語 t が数値を含んでいない場合，システムは，上昇・下降という 2 つのボタンを提示する．単語 t が数値を含んでいる場合，ランキング結果をソートするためのボタン（昇順・降順）も一緒に提示する．

(a)　ユーザが上昇（下降）ボタンをクリックした場合，システムは単語 t を含むランキング結果を上位（下位）に再ランキングする．このとき，単語 t を含んでいるランキング結果のうち，再ランキング前の順位が上位のランキング結果ほど上位に再ランキングされる．また，このときランキング結果中に出現する単語 t の頻度は考慮せず，出現の有無のみで再ランキングを行う．再ランキングに出現頻度を用いない理由は，システムの動作をユーザが理解しやすくするためである．出現頻度で再ランキングを行うと，たとえば，あるユーザがランキング結果 r 中の単語 t に対して上昇操作を行うと，そのランキング結果 r より下位にある検索結果が，検索結果 r よりも上位に再ランキングされてしまうことがある．一方，出現の有無のみで再ランキングを行うとこのようなケースは起こらず，一般的なフィルタリングと同様の挙動となり，ユーザにとって理解しやすいと考えられる．

(b)　ユーザが昇順（降順）ボタンをクリックした場合，システムは抽出した数値に基づきランキング結果をソートする．このとき，数値が抽出されなかったランキング結果は最も下位に再ランキングされる．

(3)　システムは再ランキング結果をユーザに提示する．

図 5 は，ユーザが Google Scholar [⋆1] の論文検索結果ページを再ランキングする様子である．まず，ユーザは “pseudo relevance feedback” というクエリを入力し，検索結果ページを閲覧する．ユーザが検索結果を閲覧するなかで，著者が “Wei-Ying Ma” である論文に興味を持ったとする．その場合，ユーザは検索結果中の “WY Ma” を選択し，上昇ボタンをクリックすることで，“WY Ma” を含んだ検索結果を上位に再ランキングできる．その後，ユーザが論文の被引用数の多い論文を閲覧したいと思ったとする．ユーザは検索結果中の “Cited by 181” を選択し，システムが提示した降順ボタンをクリックすることで，検索結果を被引用数に基づいてソートすることができ，論文の被引用数順に検索結果を閲覧することができる．

このように，ユーザは自らの興味にあわせてランキング結果ページ中の文字列を選択しながらインタラクティブにランキング結果を再ランキングしていくことができる．図 5 の例では，Google Scholar は検索結果中に論文の被引用数を表示しているが，被引用数に基づく検索結果のソート機能を提供していない．本システムを利用することで，このような，サービス側がソート機能を用意していない属性に関しても再ランキングが可能となる．

3.3　タームクラウドの生成

タームクラウドの生成手法は以下のとおりである．まず，ランキング結果から全文字列を抽出し，正規表現を用いて文字列を，連続するアルファベット，数値，漢字，カタカナなどに分解し単語集合を得る．その後，ストップワードを除去し，それぞれの単語についてランキング結果中の出現頻度数を計算する．そして，各単語のうち，出現頻度が $n/2$ 以下であ

⋆1 Google Scholar. http://scholar.google.com

図 5　論文検索エンジンの検索結果を再ランキングする様子．ユーザは単語ベースのフィードバックおよび数値ベースのフィードバックをシステムに伝達できる．ユーザは興味のある箇所を選択し，適切なボタンをクリックすることでランキング結果の再ランキングができる

Fig. 5　Reranking publication results with both term-based and numerical feedback in a publication search. Users can give both term-based and numerical feedback to the system and rerank the search results by simply selecting an area of interest and clicking the appropriate button.

り，かつ出現頻度が多い単語を順に 30 個抽出する（n はランキング結果ページ中のランキング結果の数を表す）．抽出した単語について，出現頻度に基づいて単語の大きさを決定し，単語を 50 音順にソートすることでタームクラウドを生成し，ランキング結果ページにオーバレイ表示する．単語の抽出の際に，単純な出現頻度順に単語を選択してしまうと，“キャッシュ” や “関連ページ” のようにほぼすべてのランキング結果に含まれ，再ランキングに不要な単語を抽出してしまう．そのため，本研究では中頻度の単語を優先的に選択している．

図 6 はクエリ “pork green pepper recipes” のウェブ検索結果 100 件から生成されたタームクラウドである．この例では，“pork green pepper recipes” というクエリに対して，“garlic”，“onion”，“chinese”，“easy” といった，料理に関する単語がタームクラウドとして提示されていることが分かる．ユーザはこのような単語でランキング結果を再ランキングすることによって，気になったランキング結果を手軽に上位に再ランキングすることができる．

3.4　パーサ作成インタフェース

本研究では，目的のページからランキング結果を認識，抽出するために，記法が分かりやすくプログラムからも扱いやすい XPath [⋆1] を利用する．パーサ作成インタフェースの目的は，対象のページからランキング結果を認識するような XPath を生成することである．

本研究で提案するパーサ作成インタフェースのアイデアは，パーサ作成の際にユーザがマウスを移動させている最中に，現在システムが認識しているノードを可視化し，視覚的なフィードバックをユーザにつねに与えることである．**図 7** は Google のウェブ検索結果ページを認識するパーサを作成する様子である．まず，ユーザはパーサ作成インタフェースを呼び出し，ウェブページ中のランキング結果をどの箇所でもよいのでクリックする．すると，システムは HTML ソースの `body` ノードから，クリックされたノードまでの DOM ツリーを図 7 左下部のように表示する．たとえば，`div.s` と表示されている箇所は，“クラス名が

⋆1 XML Path Language Viewsion 1.0. http://www.w3c.org/TR/xpath/

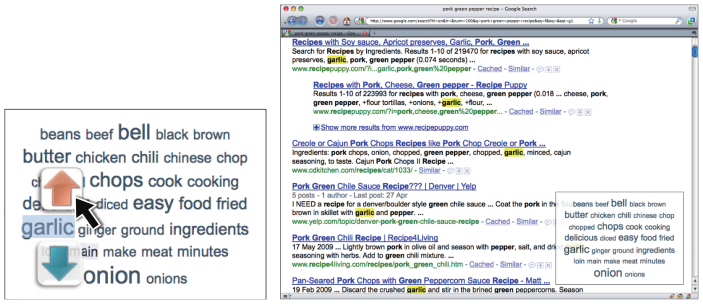

図 6　クエリ “pork green pepper recipes” のウェブ検索結果から生成されたタームクラウドの例
Fig. 6　Example TermCloud for Web search results of query “pork green pepper recipes”.

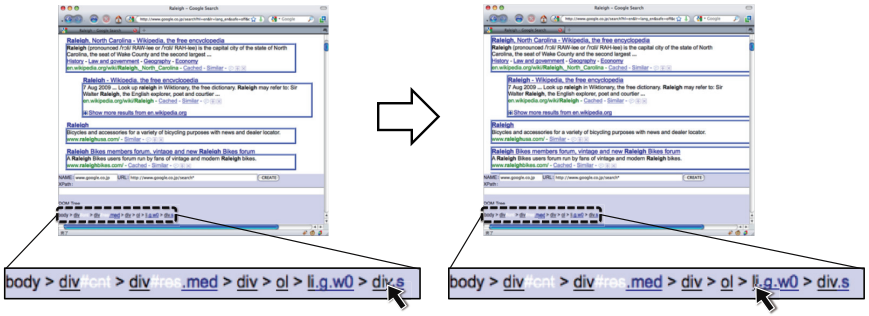

図 7　パーサ作成の様子
Fig. 7　Wrapper Generation Interface.

s であるような div 要素” を表している．その後，ユーザはシステムが表示したツリー上でマウスカーソルを移動する．すると，システムはマウスカーソルの位置に応じて，そのDOM 上で認識可能な要素集合をウェブページ上に表示する．たとえば，図 7 左部の例では，マウスカーソルは div.s 上にあるが，システムの提示を見ると，システムは検索結果のタイトルの部分まで囲めておらず，ランキング結果の単位を正しく解析できていないことが分かる．そこで，ユーザは図 7 右部のように li.g ノード上にマウスカーソルを移動する．すると，システムはランキング結果を正しく認識できていることが分かる．この箇所でパーサ作成ボタンを押すことで，システムはランキング結果を解析する XPath を自動的に作成する．同時に，システムはパーサが動作する URL や，パーサの名前などを，現在閲覧中の

表 1　www.google.co.jp のパーサ例
Table 1　Example wrapper for www.google.co.jp.

項目	出力例
パーサ名	www.google.co.jp
XPath	/html/body/div/div[@class=”med”]/div/ol/li
対象 URL	http://www.google.co.jp/search*

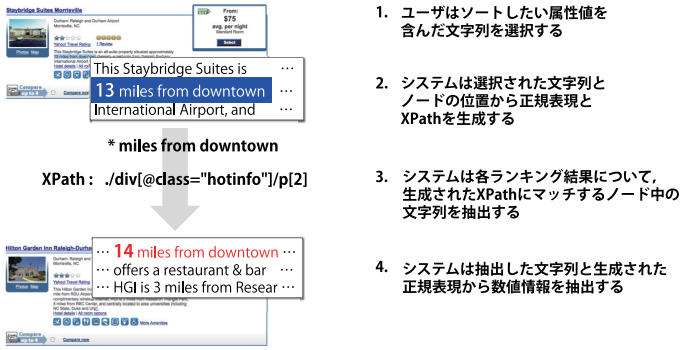

図 8　属性値の動的抽出手法
Fig. 8　The method to extract numerical attributes dynamically.

ページの URL から自動的に生成しパーサを作成する（**表 1**）．また，必要があれば，ユーザはキーボード操作を用いてパーサの動作 URL や XPath を直接書き換えることもできる．

このように，ユーザはランキング結果をクリックし，表示されたツリー上でマウスを動かし，システムからのフィードバックを閲覧しながら適切なノード上でクリックを押すというシンプルな操作だけでパーサを作成することができる．こうすることで，ユーザは XPath や DOM といった知識がなくても，視覚的なフィードバックにより適切なパーサを作成可能となると考えられる．

3.5　属性値の動的抽出

図 8 はユーザの再ランキング操作に応じて動的に属性値を抽出する手法を説明したものである．ユーザがランキング結果中から文字列を選択すると，まず，システムはユーザが選択した文字列が数値を含んでるかどうかを判定し，数値を含んでいる場合はその数値の種類の判定を行う．本研究では，あらかじめ用意した正規表現を利用することで，“15 件”，“1,000 円” のような整数値，“2010 年 5 月 3 日”，“2010/05/03” のような日付，“12:56” の

ような時間といった数値の種類を判別している．数値の種類を判別した後，システムは選択文字列中の数値部分を置換することで正規表現を生成する．同時に，ランキング結果要素のルートノードから，ユーザが選択した文字列を含むノードまでの XPath も生成する．この正規表現と XPath を用いて，システムは他のランキング結果から数値情報の抽出を試みる．

この手法により，ユーザはさまざまな属性値でランキング結果をソートすることが可能となる．図 8 のランキング結果は宿泊料金，レビューの数といったさまざまな属性値を含んでいる．こうした属性値は `<div>$75</div>`，`<span>13 reviews</span>` のように，比較的よく構造化された HTML ソースで表示されている．サービス側は，このような，よく構造化されている属性値に関してはソート機能を提供していることも多い．一方で，図 8 の例で示しているような，繁華街までの距離のような構造化されてない箇所に含まれる値に関してはソート機能を提供することはほとんどない．本システムを利用することで，ランキング結果がその文字列中に数値情報を含んでさえいれば，サービス側がその属性値による再ランキング機能を提供しているかどうかにかかわらず，ユーザはランキング結果をソートすることが可能となる．

4. 評　　価

本システムの有効性を評価するため，本稿ではシステムの動作速度，パーサ作成インタフェースの使いやすさ，本システムを用いたユーザ実験の 3 種類の実験を行った．

4.1 システムの実行速度

まず，システムの実行速度に関する実験を行った．実行速度を調査するため，ランキング結果ページをパースし，パースされた要素に対してマウスイベントを付与するまでの時間，タームクラウドを生成するためにかかる時間，文字列による再ランキングにかかる時間，および数値による再ランキングにかかる時間という 4 つの項目の実行時間を計測した．実験は Yahoo！のウェブ検索結果ページ[⋆1]を対象とし，ランキング結果ページの検索結果件数を 10 件，40 件，100 件，500 件，1,000 件と変更しながらそれぞれの実行時間を計測した．なお，実験には MacBook Pro（2.8 GHz Intel Core 2 Duo，4 GB Memory，1,400 × 900 resolution display），および，Firefox 3.6.8 を用いた．

表 2 はそれぞれの実行時間を 10 回ずつ計測した平均である．表にあるように，ランキング結果数が 100 件でも 100 ms 以内でタグクラウドの生成や再ランキングが終了することが分かる．また，ランキング結果件数が 1,000 件の場合も 1 秒以内でシステムが動作することが分かる．したがって，一般的なランキング結果ページにおける再ランキングでは，ユーザはストレスをほとんど感じることなく再ランキング機能を利用できると考えられる．

表 2　システムの実行時間
Table 2　Execution time analysis.

	ランキング結果の要素数				
	10 件	**40 件**	**100 件**	**500 件**	**1,000 件**
ランキングページのパース (ms)	8.7	16.2	31.3	232.4	461.2
タームクラウドの生成 (ms)	6.7	11.5	20.1	76.1	144.2
文字列による再ランキング (ms)	10.1	28.2	80.1	266.6	679.9
数値による再ランキング (ms)	11.3	38.3	101.9	459.8	922.8

4.2 パーサ作成インタフェース

次に，本システムのパーサ作成機能の適用範囲を調査するため，どの程度の割合のウェブサービスで正しくパーサが作成できるかの調査を行った．実験にあたり，ウェブページを検索対象としたウェブ検索エンジンから 20 件，商品や論文といったウェブページ以外を検索対象とした検索サービスから 20 件の計 40 件の検索サービスを対象に，それぞれの検索サービスのランキング結果ページに対して正しくパーサを作成できるかを調査した．ウェブ検索エンジンとして，Wikipedia の記事「検索エンジン」[⋆2]中の「主な検索エンジンサイト」に記載されている検索エンジンから 20 件選択した．また，それ以外の検索サービスは，商品検索，ホテル検索，レストラン検索，論文検索の 4 つのカテゴリから，それぞれ著者が有名と考える検索サービスを 5 件ずつ選択した．実験に用いた 40 件の検索サービスを**表 3** に示す．

表 4 はパーサ作成インタフェースが 40 件の検索サービスについてどの程度正しく動作したかを示した表である．全体として，40 件中 33 件の検索サービスのランキング結果ページに対して正しくパーサが作成可能であった．また，一般的なウェブ検索エンジンでは 90%のサービスについて，それ以外の検索サービスでは 75%のサービスに対してパーサが正しく作成可能であり，多くの検索サービスについて本システムのパーサ作成機能が適用可能であることが分かる．一部作成可能だった検索サービスには，ランキング結果以外のコンテンツも一緒に抽出してしまうもの，ランキング結果が 2 段組で表示されておりランキング結果

⋆1 http://x.search.yahoo.co.jp

⋆2 http://ja.wikipedia.org/wiki/%E6%A4%9C%E7%B4%A2%E3%82%A8%E3%83%B3%E3%82%B8%E3%83%B3

表 3　パーサ作成実験に用いた検索サービス
Table 3　Search services used in the experiment.

ウェブ検索エンジン	それ以外の検索サービス	
Ask.jp	**商品検索サービス**	Amazon.co.jp
BIGLOBE		楽天市場
Clusty.jp		価格.com
Cuil		EC ナビ
Excite		ビッダーズ
フレッシュアイ	**ホテル検索サービス**	じゃらん
goo		楽天トラベル
Google		Yahoo！トラベル
Infoseek		ゆこゆこ
ライブドア		一休
Lycos	**レストラン検索サービス**	ホットペッパー
MARSFLAG		食べログ
Mooter		ぐるなび
Bing		グルメぴあ
NAVER		NAVITIME
@nifty	**論文検索サービス**	Google Scholar
Powerset		CiNii
So-net		PubMed
Yahoo!		CiteSeer
百度		Web of Science

表 4　パーサ適用可能性
Table 4　Applicability of our wrapper generation interface.

	ウェブ検索エンジン	それ以外の検索サービス	計
作成可能	18 件	15 件	33 件
一部作成可能	1 件	2 件	3 件
作成不可能	1 件	3 件	4 件

の一部しか抽出できなかったものがあった．作成不可能であった 4 件の検索サービスのうち 3 件は，複数の兄弟ノードで 1 つのランキング結果要素を表していたため，本システムでは正しく認識できなかった．本研究では，パーサ作成インタフェースをシンプルにするため，ランキング結果の各要素は 1 つのタグで囲われており，ランキング結果どうしが兄弟ノードとして表示されていると仮定をしている．そのため，本システムではこの 3 件の検索サービスを正しく認識できなかった．この 3 件の検索サービスような構造のランキング結果ページを正しく認識するためには，ユーザにランキング結果要素を直接範囲選択してもらい，複数の兄弟要素が 1 つのランキング結果要素であることをシステムに伝達可能にするような仕組みが必要となると考えられる．

次に，パーサ作成インタフェースの使いやすさを評価するため，ユーザによる実験を行った．10 名の被験者に，5 つの検索サービスについて実際に本システムを利用してパーサを作成してもらった．パーサを作成するサービスとして，先述の実験でシステムが正しくパーサを作成できたサービスから 5 つ（Yahoo!，百度，Bing，FreshEye，Goo）を選択し実験の対象とした．被験者は全員本システムの利用は初めてであり，5 名は京都大学の大学院生，3 名は京都大学の学部生であり，2 名は会社員である．また，男性は 8 名，女性は 2 名であった．10 名の被験者のうち，HTML を記述した経験のある被験者（以下，HTML 経験者）は 5 名，HTML を記述した経験のない被験者（以下，HTML 未経験者）は 5 名であった．また，HTML 経験者のうち，1 名は XPath を記述した経験があった．

実験には，4.1 節で使用したノート PC を使用した．実験では，まず，著者がそれぞれの被験者に対して，Google のウェブ検索結果ページに対してパーサ作成インタフェースを利用してパーサを作成する例を見せながら，本インタフェースの使い方に関する説明を約 2 分程度行った．その後，被験者に実際に 5 つのサービスについてパーサを作成してもらった．実験では，パーサ作成インタフェースを呼び出すボタンを被験者がクリックしてから，パーサ作成が終わった際の確認ボタンを押すまでの時間を計測した．なお，順序効果を避けるため，パーサを作成するサービスの順番は被験者ごとに変更した．

実験の結果，4 名の HTML 経験者，3 名の HTML 未経験者の計 7 名の被験者は 5 つのサービスすべてについて正しくパーサを作成することができた．この結果は，HTML の記述経験のある，通常のユーザと比べるとウェブに関する知識が豊富であるユーザだけではなく，一般的なユーザに対しても，視覚的なフィードバックによるパーサ作成インタフェースが有効に働いていることを示している．誤ったパーサを作成していた 3 名の被験者は，ある 1 つのサービスについて，通常の検索結果だけでなく，スポンサ広告もランキング結果として認識してしまうようなパーサを作成していた．これは，通常の検索結果とスポンサ広告の表示形式が類似しており，また，スポンサ広告が通常の検索結果のすぐ上部に，リスト形式で表示されていたためであると考えられる．このような誤りを防ぐには，単にシステムが認識している構造を単純な視覚的フィードバックで提示するだけでなく，ノードのタグやクラス情報に応じて色を変化させるといったことも必要になると考えられる．また，より精度良く，高速にパーサを作成するためには，既存の構造認識手法とパーサ作成インタフェースを

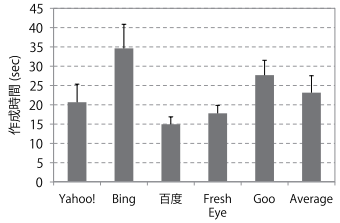

図 9　被験者がパーサ作成に要した平均時間（サービスごと）
Fig. 9　Average completion times to generate wrappers using our system for each service.

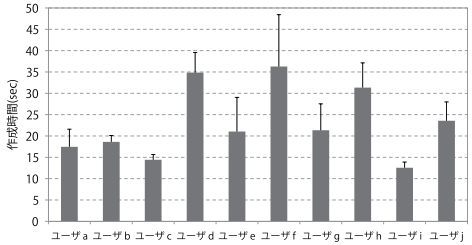

図 10　被験者がパーサ作成に要した平均時間（ユーザごと）
Fig. 10　Average completion times to generate wrappers using our system for each user.

組み合わせることも必要であろう．また，HTML 未経験者のうち 1 名は，あるサービスのランキング結果ページに，各検索結果のタイトル文字列だけを認識してしまうパーサを作成してしまっていた．実験後にこのようなパーサを作成した理由をインタビューしたところ，このサービスの検索結果ページは各タイトルだけが検索結果だと考え，要約文などはあまり重要ではないと判断したため，このようなパーサを作成したと述べていた．このように，どの部分が検索結果の単位であるかを認識する際に，他のユーザとは異なった認識をするユーザもいることが実験より分かった．今後は，一般のユーザもより簡単に，精度良くパーサを作成可能な仕組みを実現していく必要がある．

図 9，**図 10** は，被験者がパーサ作成に要した時間の平均をサービスごと，被験者ごとに表示したグラフである．図中のエラーバーは標準誤差を表している．また，ユーザ a〜ユーザ e は HTML 経験者を，ユーザ f〜ユーザ j は HTML 未経験者を表している．実験の結果，平均 23.1 秒でパーサを作成することが可能であることが分かった．また，HTML 経験者は平均 21.2 秒，HTML 未経験者は平均 25.2 秒であった．これは，HTML 経験者はシステムが提示した `div` や `li` といったタグを見ることで，ある程度どこがランキング結果を表しているのか検討がつくのに対して，HTML 未経験者の多くはそれができないためであると考えられる．実際に，パーサ作成に時間のかかった HTML 未経験者はすべてのタグに対してマウスを移動することが多く，これが作成時間の差につながったものと考えられる．しかし，一方で，ユーザ i のように，HTML に関する知識がないにもかかわらず，短い時間でパーサを作成する被験者もおり，提案システムのように視覚的にパーサを作成することの有用性を示していると考えられる．

ユーザのインタラクションを用いて対象ページのパーサを作成する類似のシステムとしては，Irmak らのシステム[7] があるが，彼らのシステムはユーザが目的のデータのスキーマを指定する必要があるため，パーサの作成に 2 分程度の時間がかかると述べられている．構造抽出の目的が異なるため，彼らのシステムと我々のシステムの比較を直接することはできないものの，平均 23.1 秒という結果は，現実的な時間でパーサを作成することが十分可能であることを示していると考えられる．

4.3　ユーザ実験

次に，本システムを使用してユーザ実験を行った．ユーザ実験の目的は，さまざまなサービスのランキング結果ページ上で，ユーザがどのようにシステムとインタラクションを行うのかを調査し，本システムの利用がユーザの情報探索行為にどのような影響を与えるのかを評価することである．

実 験 方 法

実験では，7 名の被験者に本システムを 7 日間にわたり使用してもらった．7 名のうち，3 名は 4.2 節の実験にも参加しており，残りの 4 名のうち 3 名は京都大学の大学院生，1 名は京都大学の教員である．また，被験者は全員男性であり，通常のウェブブラウジングに Firefox を使用していた．

実験では，まず，被験者に本システムの使い方を説明するため，システムに関する動画[⋆1]（約 3 分）を視聴してもらい，その後，被験者自身の PC に本システムをインストールしてもらった．システムにはあらかじめ有名な検索サービス（Google，Yahoo!，Bing，Amazon など）のパーサをインストールしておいた．実験にあたり特定の検索タスクは用意せず，日常のブラウジングの中で自由にシステムを使用してもらい，その期間のユーザの行動履歴を収集した．収集した情報は，ユーザが訪れたページの URL，再ランキングに用いられた文

⋆1 http://rerank.jp/everything/index_ja.html

表 5　再ランキング操作の分布
Table 5　Distributions of reranking operations.

	再ランキング操作の種類				
	単語ベース		数値ベース		
再ランキング箇所	上昇	下降	昇順	降順	計
ランキング結果	138	18	15	92	263
タームクラウド	131	14	-	-	145
計	269	32	15	92	408

表 6　単語ベースのフィードバックの検索サービス間による比較
Table 6　Use of term-based feedback for search services.

	ウェブ検索エンジン	それ以外のサービス	計
ランキング結果	57	99	156
タームクラウド	94	51	145

字列，再ランキング操作の種類，ユーザがクリックしたランキング結果の順位などである．

実 験 結 果

実験期間中に全被験者が行った再ランキング操作は 408 回であった．**表 5** は被験者が行った再ランキング操作の分布である．表より，約 7 割の再ランキング操作が，単語ベースのフィードバックによるランキング結果の上昇操作であり，多くのユーザが上昇操作を頻繁に利用したことが分かる．上昇操作は選択した単語を含むランキング結果を上位に表示する操作であり，ランキング結果の精度を向上させたり，単に興味のある単語を含んだランキング結果を上位に表示させたりと，さまざまな目的に利用できる．また，4.1 節の結果にもあるように，上昇操作による再ランキングにかかる時間は非常に短い．こうした理由から，多くのユーザが上昇操作をよく利用したのではないかと考えられる．

また，単語ベースのフィードバックによる再ランキングの約半数はタームクラウドを通じて行われていたことが分かった．この結果は，タームクラウドが再ランキング語の推薦としてある程度有効に働いた結果であると考えられる．

タームクラウドがどのような状況で用いられたのかを分析するため，検索サービスごとのタームクラウドの利用頻度の比較を行った．**表 6** はウェブ検索エンジン（Google，Yahoo!，Bing）とそれ以外の検索サービス上のランキング結果ページ上でのユーザの再ランキング操作の分布である．表 6 より，ウェブ検索エンジンとそれ以外のサービスでは，タームクラウドの利用頻度に大きな差があることが分かる．ウェブ検索エンジン以外の検索サービスとしては，Amazon の商品検索や Google Scholar の論文検索といった検索サービスが利用されていた．こうした検索サービス上では，タームクラウドから単語を選択するよりも，ランキング結果ページ中の単語を直接選択することで，特定のメーカの検索結果や，特定の出版社の論文を上位に再ランキングするといった，フィルタリング的な利用をしていることが多かった．論文や商品といった検索結果は，検索結果にさまざまな属性情報を含んでおり，ユーザは検索結果から再ランキングに利用する単語を比較的容易に選択できたのではないかと考えられる．一方で，ウェブ検索結果はタイトル・URL・サマリという比較的構造化されていない情報が表示されるため，その中から直接単語を選択するという行為はユーザにとってより困難であると考えられる．さらに，ウェブ検索を利用する際のユーザの目的は，他の検索サービスを利用する場合よりも欲しい情報が明確に決まっていない場合も多い．このような場合，タームクラウドのような仕組みで明示的にさまざまな単語をユーザに提示することで，ユーザは興味を引く単語や意外な単語をタームクラウドから発見することができる．こうした理由から，ユーザはウェブ検索時にタームクラウドをより積極的に利用したのではないかと考えられる．

数値ベースのフィードバックによる再ランキング操作は商品検索，論文検索，動画共有サイトといったさまざまな検索サービス上で利用されていた．たとえば，動画共有サイトでは閲覧数や動画のアップロードされた日付などが選択された再ランキングが行われており，論文検索では被引用数や論文の出版年が頻繁に再ランキングの対象となっていた．こうした属性による再ランキングの中には，サービス側がソート機能を提供をしていない属性での再ランキングもされており，本システムを利用することで，ユーザ自身の観点からランキング結果を再ランキングできることを示している．また，ウェブ検索結果ページ上での数値ベースのフィードバックによる再ランキングも行われていた．たとえば，ある被験者は，“iPod” というクエリで検索を行い，検索結果から “64 GB” という文字列を選択し，数値ベースの再ランキングを行っており，この例では，“160 GB” という文字列を含んだ検索結果が最も上位に再ランキングされていた．このように，動的に属性値を抽出することにより，ウェブ検索結果においてもさまざまな観点で検索結果をソートすることが可能であった．しかし，ウェブ検索中で被験者が行った数値ベースの再ランキング回数は合計で 22 回と非常に少なかった．この理由としては，ウェブ検索結果をソートするというタスクがユーザにとって不慣れなタスクであるということと，ウェブ検索結果というあまり構造化されていない検索結果の中から興味を持つ属性値情報を発見することが難しいことが考えられる．また，ユーザ実験後，どこまで文字列を選択すればシステムがランキング結果を意図する属性でソートしてくれるのかがよく分からないという意見を被験者からもらった．こうした問題を解決する

表 7　属性抽出の精度
Table 7　Accuracy of attribute extraction method.

正しく抽出可能	75 件
正しく抽出不可能	25 件
判断不能	7 件
計	107 件

ためには，検索結果ページ中で数値ベースで再ランキング可能な箇所をハイライトするような，推薦に近い仕組みが必要であると考えられる．

また，本稿で提案した属性値抽出手法の精度を評価するために，ユーザ実験より得られた 107 件の数値ベースのフィードバックの記録を分析した．分析にあたり，被験者が閲覧していたランキング結果ページの URL，再ランキング時に選択した文字列，選択文字列の位置といった情報をもとに，著者が実際にランキング結果ページにアクセスし，属性値抽出がどの程度ユーザの意図を正しく反映していたのかを調査した．分析結果を**表 7** に示す．表 7 の “判断不能” とは，被験者が閲覧していたページがログインを必要とするパーソナルなページであり，著者からは実際にどのようなページだったのかアクセス不可能であるため，再ランキングの意図とその結果が著者には判断不可能であったものを表している．

表 7 から分かるように，107 件中 75 件の再ランキングは正しく属性値を抽出可能であった．本稿で提案した属性値抽出手法はユーザの選択文字列とその位置を利用したシンプルな手法ではあるが，7 割の再ランキングが正しくユーザの意図どおりに動作していると考えられ，実際にシステムを利用するうえで有用であると考えられる．しかし，属性値が正しく抽出できなかったケースも 25 件存在しており，その多くは属性値抽出手法が表記揺れに対応できていないことや，属性値の数値の種類が正しく判別できなかったことが原因であった．表記揺れが原因となったケースとして，被験者があるランキング結果から “10x zoom” という文字列を選択した際に，他のランキング結果の “12 zoom” という文字列から 12 という数値を抽出すべきところを正しく抽出できていなかったことがあった．また，属性値の数値の種類を正しく判別できなかったケースとして，被験者があるランキング結果から “11ヶ月前” といった文字列を選択した際に，他のランキング結果中の “1 年前” や “30 分前” といった表記の異なる文字列から数値を抽出できていなかったことがあった．今後は，表記揺れや数値の種類の判定の問題を解決した手法を考える必要がある．さらに，表記揺れの問題は単語ベースの再ランキングにおいても解決すべき問題である．今後はより柔軟に，また，よりユーザの意図にそった再ランキングができるように，周辺テキストの類似度や構造の類似度を用いて再ランキングを行う手法が必要になると考えられる．

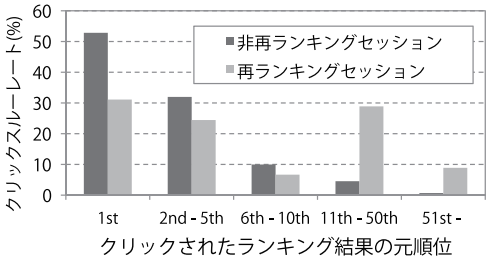

図 11　クリックスルーデータの比較
Fig. 11　Comparison of clickthrough rates.

最後に，本システムを利用することで，実際にユーザが閲覧するランキング結果にどのような影響があるのかを調査するため，ユーザ実験で得られたランキング結果のクリックスルーデータの分析を行った．まず，被験者があるランキング結果ページを訪れ，そのページを閉じるか，別の検索サービスのランキング結果ページを訪れるまでに行われた行動を 1 つの検索セッションと定義し，その検索セッションを “非再ランキングセッション” と “再ランキングセッション” の 2 種類に分類した．“非再ランキングセッション” とは，被験者が再ランキング操作を 1 度も行わなかったセッションである．また，“再ランキングセッション” とは被験者がランキング結果ページ内で 1 回でも再ランキング操作を行ったセッションである．**図 11** は 2 つの検索セッションにおけるクリックスルーデータの分布を比較したものである．図 11 における “元順位” とは，ユーザが再ランキング操作を 1 度も行っていない，元々のランキング結果ページにおけるクリックされた結果の順位を表している．結果を見ると，再ランキングを行った検索セッションでは，ユーザは元々下位にあったランキング結果をクリックする可能性が高いことが分かる．通常の検索では，ユーザが上位の数件しかランキング結果を閲覧しないため，複数のユーザが同じクエリを投入したとしても，同じ検索結果しか閲覧しない．一方で我々の結果は本システムを利用し検索結果を自らの興味で再ランキングすることによって，ユーザごとに異なる多様な検索結果を閲覧する可能性が高い．

このように，本システムを利用することによって，ユーザはさまざまな観点からランキング結果を閲覧し，また，普段は閲覧しないような下位のランキング結果を閲覧する可能性が高くなることが分かった．

表 8　4.4 節の実験に用いたクエリ
Table 8　Queries used in Section 4.4.

論文検索タスク	適合フィードバック	ウェブ検索
レシピ検索タスク	豚肉 ピーマン	牛肉 タマネギ

4.4　通常の検索サービスとの比較実験

最後に，通常の検索サービスのみを用いた場合と本システムを用いた場合で，検索の効率性がどのように変化するのかを評価するため，ユーザ実験を行った．

実 験 方 法

6 名の被験者により実験を行った．6 名のうち，2 名は 4.3 節のユーザ実験にも参加した京都大学の大学院生であり，残りの 4 名のうち 3 名は京都大学の学部生，1 名は京都大学の大学院生である．実験は，4.1 節で述べたノート PC 上で行った．実験では，まず，被験者に本システムの使い方を説明するため，4.3 節に示したシステムに関する動画を視聴してもらった．その後，被験者にシステムおよび実験環境に慣れてもらうため，Google と Google Scholar 上でそれぞれ 5 分程度自由にシステムを使用してもらった．そして，以下に示す 2 つの検索タスクを被験者に行ってもらった．

（1）論文検索タスク：Google Scholar を利用して，**表 8** の上部に示したクエリに言及している論文のうち，被引用数が 500 件以上の論文を 10 件収集する．ただし，タスクにかける時間は最大 5 分とする．

（2）レシピ検索タスク：Google を利用して，表 8 の下部に示したクエリを使用したレシピのうち，被験者が興味を引くと判断するレシピを 5 分間収集する．その際，被験者は閲覧中のある検索結果のレシピに対して興味があると判断すると，その検索結果をダブルクリックすることで，興味があることをシステムに伝える．

被験者は，(1)，(2) の検索タスクを，表 8 に示した 2 種類のクエリについて行う．このとき，1 つのクエリは通常の検索サービスのみを用いて，もう 1 つのクエリは本システムも用いて行う．また，6 名の被験者を 2 グループに分割し，提案システムを使用するクエリをそれぞれのグループごとに変更した．さらに，本システムを利用するかどうかにかかわらず，被験者はタスク中にクエリを修正して再検索したり，検索サービスが提供する機能を利用したり，検索結果の URL をたどったりすることを自由に行ってよいものとした．

前者のタスクの目的は，本システムを利用する場合と利用しない場合で，属性値に関連した検索意図に関するタスクの実行速度がどのように変化するのかを評価することである．また，後者のタスクの目的は，本システムを利用する場合と利用しない場合で，被験者が興味を引くと判断する検索結果の数にどの程度差が生じるのかを評価することである．

表 9　論文検索タスクにかかった時間の平均，() 内は標準偏差を表す
Table 9　Average times to complete paper search tasks.

	タスクにかかった時間
本システムあり	127.67 秒 (65.17)
本システムなし	232.00 秒 (58.01)

表 10　レシピ検索タスクにおいて被験者が興味があると判断した検索結果数の平均，() 内は標準偏差を表す
Table 10　Average numbers of search results in which subjects were interested.

	個数	**被験者ごとに正規化**
本システムあり	20.1 (23.44)	0.58 (0.10)
本システムなし	16.5 (21.22)	0.42 (0.10)

実 験 結 果

まず，論文検索タスクの結果について述べる．本システムを利用した場合，6 名の被験者全員が 5 分以内に 10 件以上の論文を検索することができた．一方，Google Scholar の機能のみを用いた場合では，6 名中 2 名の被験者は時間内にタスクを終えることができなかった．本システムでは被引用数を選択することで検索結果を被引用数順に並べ替えることができる．しかし，Google Scholar は被引用数に基づくソート機能を提供していないため，本システムを利用しない場合，被引用数の多い論文の探索に時間がかかるため，時間内に論文を 10 件発見することができなかったと考えられる．**表 9** は，論文検索タスクにかかった時間を，本システムを利用した場合と，Google Scholar の機能のみを用いた場合で比較した結果である．なお，5 分以内にタスクが完了しなかった場合は，タスク完了にかかった時間を 5 分として計算した．表から分かるように，本システムを利用した場合は平均 127 秒であり，通常のインタフェースよりも効率良く論文を収集できたことが分かる．この結果は，検索サービスがソート機能を提供しない属性に関する意図を持った検索を行う場合，本システムを利用することで効率良く検索することができることを示している．

次に，レシピ検索タスクの結果について述べる．**表 10** は，レシピ検索タスクにおいて被験者が興味があると判断した検索結果数を，本システムを使用した場合と，通常の Google のみを用いた場合とで比較した結果である．表 10 より，本システムを利用することで，Google の機能だけを利用するよりも，興味のある検索結果を多く発見することができていることが分かる．被験者が興味があると判断した検索結果の数は，被験者ごとに大きく異なっていた（最大 67 個，最小 2 個）．そのため，被験者間の影響を抑制するために，興味があると

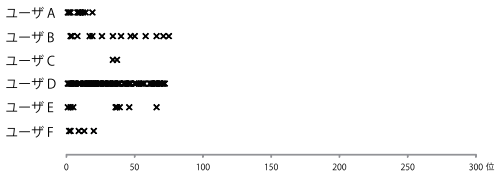

図 12　被験者が興味があると判断したランキング結果の元順位の分布（本システムなし）
Fig. 12　Distributions of ranks for each subject without using out system.

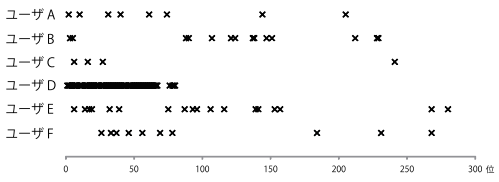

図 13　被験者が興味があると判断したランキング結果の元順位の分布（本システムあり）
Fig. 13　Distributions of ranks for each subject using out system.

判断した検索結果数を被験者ごとに正規化し，その平均を求めたものを表 10 の右部に示している．Google のみを利用する場合，ユーザは検索意図を伝達する手段がクエリを投入するかクエリを修正して再検索することしかない．一方，本システムを利用することでユーザは興味のある単語を選択して再ランキングすることができる．Google のみを利用して検索するタスクでは，ユーザがクエリを修正して再検索を行った回数は平均 0.5 回であった．一方で，本システムを利用した際にユーザが行った再ランキング回数は平均 6.5 回であった．再ランキングによりユーザ自らの興味を何度も伝達することで，興味を引く可能性の高い検索結果を手軽に上位に再ランキングできるため，Google のみを利用するよりもより多くの興味を持つ検索結果を発見できたのではないかと考えられる．

また，**図 12**，**図 13** は，レシピ検索タスクにおいて，被験者が興味があると判断した検索結果の元順位の分布を示したものである．提案システムを利用しない場合は，検索結果を上位から閲覧することしかできないため，図 12 のように被験者が興味があると判断した検索結果の多くが上位に存在することが分かる．一方，提案システムを利用しながらレシピを検索することで，図 13 のように，上位の検索結果だけではなく元々下位に存在する検索結果も多く選択されていることが分かる．こうした結果は，普段のブラウジングでは閲覧しないような検索結果を閲覧する可能性が上がるだけではなく，ユーザごとにそれぞれ異なる，多様な検索結果を閲覧できる可能性も示している．

5. 考　　察

本システムでは，ランキング結果を再ランキングするために，ユーザ側でパーサを作成する必要がある．4.2 節の実験により，パーサの作成にかかる時間は平均 23.1 秒であったが，本実験のユーザ層はウェブに慣れているユーザであり，通常のユーザにとってパーサを作成して再ランキングを行うことはある程度負担であると考えられる．4.3 節のユーザ実験では，パーサをまったく作成しないユーザや，パーサを積極的に作成するユーザがおり，最も積極的にパーサを作成したユーザは，7 日間の間に 20 以上のサービスに対してパーサを作成していた．このように，積極的にパーサを作成するユーザの行為を活用するために，現在ではユーザが作成したパーサを自動的に他のユーザと共有する仕組みを実現している．これは，ユーザがパーサを作成すると同時に，全システムで共通のリポジトリ[*1]にパーサがアップロードされ，システムが定期的にそのリポジトリにアクセスしてパーサをダウンロードする形として実装されている．現在では，113 のサービスに関するパーサがリポジトリに登録されている．パーサを共有することで，一般のユーザもパーサを作成せずともさまざまな検索サービス上で再ランキング機能を利用できると考えられる．

本システムは，ランキング結果を再ランキングすることを目的に実装したシステムであるが，著者が実際にシステムを使用していくなかで，ランキング結果の再ランキング以外にも有用なケースがあることが分かった．本システムのパーサ作成機能は，コンテンツの単位がシンプルなリスト構造になっていればどのようなコンテンツでも再ランキング対象のコンテンツとして扱える．したがって，掲示板の書き込みや，表データのコンテンツなどについても再ランキングの対象とすることが可能である．たとえば，掲示板の書き込みの中から特定の人物の書き込みやスパムコメントを下位に落とすことで可読性を向上させたり，また，1 日ごとのアクセス解析ログのような膨大な表データに対して，指定した IP を選択し上昇することで，特定のユーザの行動を追ったりといった使い方ができる．

また，本システムの利用により得られるユーザの再ランキング操作ログは，ユーザの嗜好を強く反映したものであり，ランキング結果のパーソナライズに非常に有用であると考えられる．特に，本システムはクライアント側で動作するため，サービス横断的にユーザの再ラ

⋆1 wedata. http://wedata.net/databases/RerankEverything/items

ンキング操作ログを収集することができる．そのため，ある検索サービスのランキング結果をパーソナライズする際に，別の検索サービス上でのユーザの行動履歴を利用できる．たとえば，あるユーザが京都の観光地に関する情報をウェブ検索エンジンを利用して調べ，その後，京都の宿泊場所を探そうとホテル予約サービスのランキング結果を閲覧したとする．このとき，ウェブ検索結果を閲覧していた際にユーザがどんなページをクリックしていたか，どんな単語を選択して検索結果を再ランキングしていたかといった情報を利用することで，ホテルのランキング結果を自動的に再ランキングするといったことが考えられる．さらに，クライアント側で動作することの利点として，個人的なコンテンツに関する検索結果を扱うことができるという点もあげられる．たとえば，あるユーザ自身が登録したお気に入り動画の一覧のページや，デスクトップ検索の検索結果といった，通常の検索エンジンではインデックス化できない Deep Web のようなコンテンツも，コンテンツがウェブブラウザ上でリスト形式で表示されていれば，本システムは扱うことが可能である．通常の検索サービス上で行ったインタラクションの情報を利用することで，こうした個人的なコンテンツを再ランキングするといったことも可能になると考えられる．このように，サービスに依存しない形でユーザの興味を抽出することで，より柔軟なパーソナライズが可能になると考えられる．

6. 関 連 研 究

6.1 ウェブコンテンツに対する機能拡張

WWW 上のコンテンツが多様化するにつれて，さまざまな目的に合わせたシステムが提案されている．Hunger Gatherer[8] や Internet Scrapbook[9] は，ウェブページ中のコンテンツを自由に切り取り，切り取った情報を手軽にまとめることを支援するシステムである．Piggy Bank[10] や Thresher[11] はウェブページから構造化されたデータの抽出を支援するシステムである．WebVCR[12] や Smart Bookmark[13] は，あるウェブサイトにおけるユーザの一連の行動を記録しておくことで，ユーザがそのサイトを再訪問した際に，その行動を再生することで，通常のブックマークでは再現できない動的なコンテンツを復元することが可能である．こうしたシステムを利用することでユーザはウェブページ中の情報の管理が容易になるが，ページ中のコンテンツをユーザ側の尺度で並べ替えながら閲覧するということはできず，本システムと目的が異なる．

我々の提案するシステムと類似のシステムとしては，Sifter[14] がある．Sifter はブラウザの拡張機能として実装されており，検索結果ページの構造を自動的に解析することで，検索結果に対してソートやフィルタリングの機能をユーザに提供する．しかし，Sifter 上でユーザが実際に検索結果の再ランキングを行うには，ブラウザのサイドバーを開いてシステムを起動し，検索結果を解析を行い，サイドバーに提示されたリストボックスから必要な情報を 1 つ 1 つ選択していくといったように，複数回のインタラクションが必要であり，ユーザにとって再ランキングのコストが高かった．それに対して，我々のシステムは，ユーザはページ上に表示されているランキング結果そのものに対してインタラクションを行うことで再ランキングを行うものであり，より直感的で少ないインタラクションで検索結果の再ランキングを行うことが可能である．

6.2 ウェブ検索におけるインタラクション

情報検索の分野では，ユーザの適合性判断を利用して検索結果の精度を向上させる技術として適合フィードバック[15] がある．我々のシステムの一部は，単語ベースの適合フィードバックシステムと見なすことができる．検索結果に対して適合性判断を行うよりも，単語に対して適合性判断を行う方がユーザにとって容易であり，ユーザの興味が強く反映されているため，単語ベースのフィードバックは重要であると考えられている[16]．適合フィードバックはインタラクティブに検索結果を閲覧するための重要な技術であるにもかかわらず，多くの研究がユーザが行ったフィードバックをどうランキングに反映するかを扱ったものであり，ユーザがシステムに対してどのようにフィードバックを与えるのかに関して考えられてはいない．たとえば，Tan らのシステム[16] では，ユーザはシステムが提示した複数のチェックボックスに 1 つ 1 つチェックをしていくことで初めてフィードバックが可能になるなど，フィードバックを伝達するためのコストが大きい．我々は，フィードバックをランキング結果にどのように反映させるかと同様に，フィードバックを与えるためのインタフェースも重要であると考えている．

また，ほかにもさまざまなインタラクティブな検索手法が提案されている．Hearst らはウェブページ，画像，歴史的オブジェクトなどを検索するために，ファセットに基づく検索手法を提案している[17),18)]．Paek らは魚眼レンズを用いることで，検索結果のレイアウトを動的に変化させるシステムを提案している[19)]．また，検索結果をクラスタリングすることで，ユーザのウェブ検索を支援する手法もさまざまなものが提案されている[20),21)]．こうしたシステムは，ある特定の種類の検索結果を対象としたものであり，また，ユーザ側でランキング結果を自由に変更することができない．しかし，本システムをこうしたシステムの検索結果ページ上で動作させることも可能であり，彼らのシステムと本システムの機能を組み合わせることもできると考えられる．

7. ま と め

本稿では，さまざまなランキング結果を再ランキングすることができるシステム RerankEverything を提案し，プロトタイプを実装した．本システムを利用することで，ユーザはシンプルで統一的なインタフェースでランキング結果の再ランキングが可能となり，自らの興味にあわせてランキング結果を自由に閲覧していくことが可能となる．

ユーザ実験では，本システムを利用することでユーザはさまざまな観点でランキング結果を再ランキングし，元々下位にあったランキング結果を閲覧する可能性が高くなるといった結果が得られた．また，タームクラウドを利用しさまざまな単語によりランキング結果を再ランキングしながらランキング結果を閲覧することが分かった．このような検索行為を行うことによって，ユーザはより積極的にランキング結果を閲覧できるようになると考えられる．パーサ作成インタフェースに関しても，実験よりユーザは平均 23 秒程度で 1 つのパーサを作成できることが分かり，有効に働くことが分かった．今後は，システムを広く公開することでユーザの操作ログを収集し，大規模なログ分析を行いたいと考えている．

また，本システムはモバイル環境にも適していると考えられる．モバイルのデバイスは文字入力環境が制限されている場合が多く，通常の PC よりも，クエリを再修正することが困難であると考えられる．本システムではユーザはキーボードを利用することなくマウス操作のみでランキング結果を再ランキングすることが可能である．そのため，たとえばタッチスクリーンを備えたデバイスでは，ディスプレイに直接タッチすることでランキング結果を再ランキングするということが考えられる．今後は，そのようなモバイルデバイス上でシステムを実装し，ユーザ実験を行っていきたい．

謝辞　本研究の一部は，科学研究費補助金特別研究員奨励費「ユーザインタラクションに基づく情報検索」（研究代表者：山本岳洋，課題番号 09J55302），グローバル COE 拠点形成プログラム「知識循環社会のための情報学教育研究拠点」，文部科学省科学研究費補助金特定領域研究「情報爆発時代に向けた新しい IT 基盤技術の研究」，計画研究「情報爆発に対応するコンテンツ融合と操作環境融合に関する研究」（研究代表者：田中克己，課題番号 1809041），独立行政法人情報処理推進機構「IPA」未踏 IT 人材発掘・育成事業 2008 年度上期未踏ユースによるものです．また，本研究を進めるにあたり，慶應義塾大学安村通晃教授からさまざまなご指導ご助言をいただきました．ここに記して感謝の意を表します．

参 考 文 献

1) Cui, H., Wen, J., Nie, J. and Ma, W.: Probabilistic query expansion using query logs, *Proc. 11th International Conference on World Wide Web*, pp.325–332 (2002).
2) Wen, J.-R., Nie, J.-Y. and Zhang, H.-J.: Clustering user queries of a search engine, *Proc. 10th International Conference on World Wide Web*, pp.162–168 (2001).
3) 山本岳洋，中村聡史，田中克己：Rerank-By-Example：編集操作の意図伝播によるウェブ検索結果のリランキング，情報処理学会論文誌（トランザクション）データベース，Vol.49, No.SIG7(TOD37), pp.16–28 (2008).
4) White, R. and Morris, D.: Investigating the querying and browsing behavior of advanced search engine users, *Proc. 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.255–262 (2007).
5) Käki, M. and Aula, A.: Controlling the complexity in comparing search user interfaces via user studies, *Information Processing & Management*, Vol.44, No.1, pp.82–91 (2008).
6) Sinclair, J. and Cardew-Hall, M.: The folksonomy tag cloud: when is it useful?, *J. Inf. Sci.*, Vol.34, No.1, pp.15–29 (2008).
7) Irmak, U. and Suel, T.: Interactive wrapper generation with minimal user effort, *Proc. 15th international conference on World Wide Web*, ACM New York, NY, USA, pp.553–563 (2006).
8) Zhu, Y., Modjeska, D., Wigdor, D. and Zhao, S.: Hunter gatherer: interaction support for the creation and management of within-web-page collections, *Proc. 11th international conference on World Wide Web*, pp.172–181 (2002).
9) Sugiura, A. and Koseki, Y.: Internet scrapbook: automating Web browsing tasks by demonstration, *Proc. 11th annual ACM symposium on User interface software and technology*, pp.9–18 (1998).
10) Huynh, D., Mazzocchi, S. and Karger, D.: Piggy bank: Experience the semantic web inside your web browser, *Web Semantics: Science, Services and Agents on the World Wide Web*, Vol.5, No.1, pp.16–27 (2007).
11) Hogue, A. and Karger, D.: Thresher: automating the unwrapping of semantic content from the World Wide Web, *Proc. 14th international conference on World Wide Web*, ACM, p.95 (2005).
12) Anupam, V., Freire, J., Kumar, B. and Lieuwen, D.: Automating Web navigation with the WebVCR, *Computer Networks*, Vol.33, No.1-6, pp.503–517 (2000).
13) Hupp, D. and Miller, R.C.: Smart bookmarks: automatic retroactive macro recording on the web, *UIST '07: Proc. 20th annual ACM symposium on User interface software and technology*, New York, NY, USA, ACM, pp.81–90 (2007).
14) Huynh, D., Miller, R. and Karger, D.: Enabling web browsers to augment web

sites' filtering and sorting functionalities, *Proc. 19th annual ACM symposium on User interface software and technology*, pp.125–134 (2006).
15) Salton, G.: *The SMART Retrieval System Experiments in Automatic Document Processing*, pp.312–323 (1971).
16) Tan, B., Velivelli, A., Fang, H. and Zhai, C.: Term feedback for information retrieval with language models, *Proc. 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.263–270 (2007).
17) Hearst, M.: Design recommendations for hierarchical faceted search interfaces, *ACM SIGIR Workshop on Faceted Search* (2006).
18) Yee, K., Swearingen, K., Li, K. and Hearst, M.: Faceted metadata for image search and browsing, *Proc. SIGCHI Conference on Human Factors in Computing Systems*, ACM, p.408 (2003).
19) Paek, T., Dumais, S. and Logan, R.: WaveLens: a new view onto Internet search results, *CHI '04: Proc. SIGCHI conference on Human factors in computing systems*, New York, NY, USA, ACM, pp.727–734 (2004).
20) Hearst, M.A. and Pedersen, J.O.: Reexamining the cluster hypothesis: scatter/gather on retrieval results, *SIGIR '96: Proc. 19th annual international ACM SIGIR conference on Research and development in information retrieval*, New York, NY, USA, ACM, pp.76–84 (1996).
21) Zamir, O. and Etzioni, O.: Grouper: a dynamic clustering interface to Web search results, *Computer Networks-the International Journal of Computer and Telecommunications Networkin*, Vol.31, No.11, pp.1361–1374 (1999).

(平成 22 年 6 月 19 日受付)
(平成 22 年 10 月 14 日採録)

(担当編集委員　渡辺 知恵美)

山本　岳洋（学生会員）
京都大学大学院情報学研究科博士後期課程在学中．日本学術振興会特別研究員（DC1）．2007 年京都大学工学部情報学科卒業．2008 年京都大学大学院情報学研究科社会情報学専攻修士課程修了．ウェブ検索，ユーザインタラクションに関する研究に従事．日本データベース学会学生会員．

中村　聡史（正会員）
京都大学大学院情報学研究科附属情報教育推進センター特定准教授．2004 年大阪大学大学院情報学研究科博士後期課程修了．博士（工学）．主にヒューマンコンピュータインタラクション，ウェブ検索の研究に従事．日本データベース学会会員．

田中　克己（正会員）
京都大学大学院情報学研究科社会情報学専攻教授．1976 年京都大学大学院博士修士課程修了．博士（工学）．主にデータベース，マルチメディアコンテンツ処理，ウェブ検索の研究に従事．IEEE Computer Society，ACM，人工知能学会，日本ソフトウェア科学会，日本データベース学会各会員．