# 履歴情報を用いた検索結果のリランキング手法

システム情報工学研究科
古瀬一隆 陳漢雄
大保信夫 西原清一

## 概 要

文章のデジタル化が進む中、ユーザに必要な文章を探しやすくすることが重要となってきている。しかし従来の検索システムでは、個々のユーザの目的に則してランキングを作成するものはほとんど存在せず、ユーザは情報の絞り込み作業に大きな労力を費し、負担となっている。また従来のシステムではランキングを作成しにくい環境が存在する。このような問題に対し、本稿ではユーザの検索履歴を用いたリランキング手法を提案する。これはページ閲覧履歴を用いて、ユーザの嗜好情報をプロファイルに蓄積し、それに基づいてランキングを修正する手法である。これにより、ユーザごとの嗜好に合わせたランキングを結果として返すことが可能となる。また、本手法では従来の検索システムで収集している情報を用いることで、従来手法では精度の良いランキングを作成することが困難であった環境でも、精度を落とさずにランキングを作成することが可能となる。

## 1 はじめに

近年、インターネットの普及やブロードバンド化の促進、情報検索技術の向上にともない、各媒体のデジタル化が進み Web 上に存在するデータや文章の数が膨大になっている。その量は検索エンジン Google[1]の検索対象ページだけでも 2005 年 1 月現在 80 億ページを越えており、このことからも Web 上に存在する文献の多さがわかる。

このような環境の下でユーザが各自の目的にあった情報を収集することは非常に困難である。ユーザは自分が欲しい情報を収集するために情報の絞り込み作業を行わなければならない。その代表例が検索システムを用いたキーワードによる絞り込みである。ユーザはキーワード又はキーワード集合を検索クエリとしてシステムに与え、システムは与えられた文字列が含まれるページ集合を結果として返す。

しかしこの手法は一度で十分な絞り込みが行えないケースが多々あり、ユーザは必要な情報が見つかるまでこの手順を繰り返して絞り込み作業を行う必要が生じる。検索に慣れたユーザならば、効果的な絞り込みが行えるであろうキーワード集合を指定することも可能であるが、検索に不慣れなユーザにとっては、この作業は大きな負担となる。

一般的な検索システムでは、適合文書として絞り込んだそれぞれの情報をランキングとしてユーザに提供している。これはページの重要度を推測し、その重要度順に並べるというものである。ユーザにとって必要な情報が上位にランクしていれば、ユーザは短時間で目的の情報を探すことができる。よってこのランキングの精度が重要となる。

ランキングの精度といっても一概にその善し悪

[1]http://www.google.com/

しを評価することは難しい。ユーザそれぞれによって趣味、嗜好が異なるため、同じ検索キーワードを入力した場合でも必要な情報が異なるからである。現在の検索システムは、どのユーザに対しても各ページの重要度は同じ値を返している。しかしユーザごとに必要な情報は異なるため、より精度の高いランキングを実現するためには、ユーザの嗜好に合わせて重要な情報を推定し、提供する必要がある。

検索を行う環境によってはランキングが作りづらい場合もある。例えば前述の Google では Web のリンク構造を用いてランキングを作成している。しかしサイト内検索の場合、リンク構造の特徴が通常の Web 検索とは異なる。「重要なページに多数リンクを張っているページは重要なページである」という従来手法の概念で上位にランクされるページが、ユーザにとって必要なページとならないことがあるためである。このような環境では、リンク構造から精度の良いランキングを作成することが困難である。

また、電子図書館などのシステムでは、そもそも Web のリンク構造自体をデータとして内包していないため、リンク構造を用いたランキング手法を使うことができない。また、検索対象となる文献の情報も十分に得ることができないことも多い。したがって、索引語頻度のような情報検索の代表的な手法でのランキングも困難である。しかし、このような環境でもランキングを作成することは、ユーザの情報収集の能率を上げるためには必要不可欠である。

このような問題を解決するため、本稿ではユーザの検索履歴を用いたランキングの手法を提案する。これは従来の検索システムのほとんどが収集しているセッションごとのページ閲覧履歴を用いて、ユーザの嗜好情報をプロファイルに蓄積し、そのプロファイルに基づいてランキングを修正するという手法である。プロファイルには閲覧ページのキーワードやその頻度を蓄積してユーザの嗜好情報を保存する。

この手法により、同じ検索クエリに対する結果でも、ユーザごとの嗜好に合わせたランキングを結果として返すことが可能になる。さらに本手法では従来の検索システムで収集している情報のみを用いることで、ページ間のリンク構造が使いづらい環境でも精度を落とさずランキングを作成し、ユーザの情報収集の負担を軽減することが可能になると期待できる。

本稿の構成は以下のとおりである。まず、第 2 節では情報検索におけるランキングの現状と、検索履歴を用いたランキングの作成手法について述べる。第 3 節ではユーザの嗜好や、類似するユーザの情報を用いる協調フィルタリングという手法について論じる。第 4 節では提案手法である検索履歴を用いたリランキング手法について述べる。第 5 節で実験、第 6 節で考察を行い、本手法の有用性を示す。最後に第 7 節でまとめを行う。

# 2 情報検索とランキング

## 2.1 情報検索

情報検索システムでは、求める情報を検索式の形で表現し、その検索式の解釈に基づいて検索を行なう。手法によっては、検索式の修正と検索結果に対するフィードバックを繰り返し、ユーザが求めるデータを特定していくものもある。現在の情報検索システムでは、ユーザが指定したクエリが出現するページそれぞれについて、そのページの重要度を推定してランキングにして返すものが主流となっている。

この情報検索において問題となるのが、検索過程がセッションごとに独立であるということ、すなわち、ユーザ同士の情報共有ができないということである。例えば、同じ事項に興味があるユーザが複数いたとして、同じキーワードで検索をしたとする。このとき、最初のユーザが情報収集のために行った作業は他のユーザには全く活かされず、他のユーザは前に検索を行ったユーザと全く

同じ情報が欲しかったとしても、一からの情報収集作業が必要とされる。

このことは何も異なるユーザ間に限ったことではない。同じユーザが過去に閲覧したページを再び検索するという作業の場合でも、過去の情報収集の作業過程を覚えていない限り、一からの情報収集作業が必要となる。これは作業能率が悪く、ユーザにとっては大きな負担となる。

本稿の提案手法では、過去に検索を行ったユーザの情報収集過程や閲覧ページに関する情報、特に検索をしようとしているユーザと興味や嗜好が似ているユーザの情報を何らかの形で情報検索に活用する。これにより、ユーザの情報収集作業が能率的になるよう支援する。

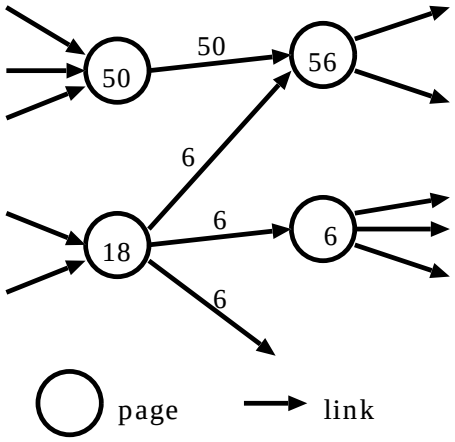


図 1: PageRank の例

## 2.2 ランキング

### 2.2.1 情報検索におけるランキング

前節でも述べた通り、現在の情報検索システムのほとんどはその結果をランキングという形でユーザに返す。これは、検索システム独自の計算手法でページの重要度 (スコア) を推測し、そのスコアの大きいものから順に並べるというものである。ユーザにとって必要な情報が上位にランクしていれば、ユーザは短時間で目的の情報を探すことができる。現在ではこのランキング表示が検索システムとしては最も一般的で、様々な検索システムで用いられている。

しかし、このランキングは精度が良くなければ実用的とは言えない。ユーザの手間を考えても、目的の情報が下位にいけばいく程ユーザの情報収集作業は増える。検索結果として返される情報が多ければ、下位にランクした情報にユーザがたどりつかないうちに諦めてしまうことも考えられる。したがって、ユーザに必要な情報が上位にランクされるようなランキングが求められる。

ランキングを用いている検索システムで現在最も成功しているもののひとつが Google[1] である。Google では PageRank という手法を用いてそれぞれの Web ページのスコアを計算している [2][4]。スコアの算出には Web のリンク構造を用いる。図 1 に例を示す。ページスコアが 50 のページからリンクを張られているページは 1 つだけ存在するので、そのページにはスコア 50 が与えられる (図の 56 のノード)。このページはスコア 18 のページからもリンクを張られている。しかしスコア 18 のページは 3 つのページにリンクを張っているため、それぞれにスコア 6 が配分される。その結果、図の右上のページのスコアは 56 となる。このような計算を全てのページ間のリンクについて行い、収束した値をそれぞれのページのスコアとする。

このような算出法により、PageRank は、多くの重要なページからリンクを張られているページ程、大きなスコアをつける。このスコアの値によって重要なページを推定し、ランキングを決定する。

この PageRank はページに対するスコアであるため、検索クエリによって変化しない。また、ユーザが異なっても同じランキングが返される。

もう一つの代表的なランキング手法として HITS[3] が挙げられる。HITS ではリンク構造に hub と authority という概念を用いて Web ページのスコアを計算する。

この手法では、各ページについて hub 度と authority 度を計算する。hub 度はそのページがリンク

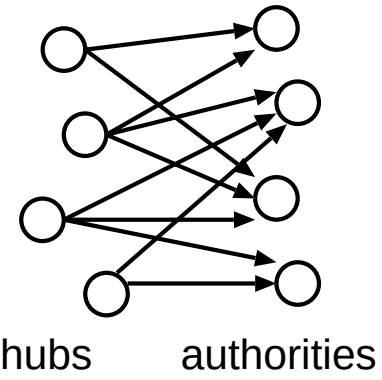


図 2: hub と authority の関係

を張っているページの authority 度の合計、authority 度はそのページにリンクを張っているページの hub 度の合計で算出される (図 2。したがって、重要なページに多数のリンクを張っているページは hub 度が高くなり、重要なページに多数リンクを張っているページからリンクを張られているページは authority 度が高くなる。この計算を値が収束するまで繰り返し、ページの重要度を推測する。この手法も PageRank と同様、ユーザが異なっても同じランキングが返される。

また、HITS にユーザの重要度も定義して、重要なユーザが閲覧したページを重要なページ、重要なページを多数閲覧したユーザを重要なユーザとしてページのスコア算出に用いている手法もある [5]。

このようなランキング手法は近年良く用いられている手法であるが、特徴として Web のリンク構造を用いていることが挙げられる。したがってこの手法を用いるには十分なリンク構造が構成されていることが必要である。また、どのユーザが検索しても、検索クエリが同じならば、返される結果も同じなので、同じ綴りのキーワードで複数の意味を持つ場合などユーザの嗜好の違いによって必要な情報が異なる場合、本当に有用な情報が上位にランクされるとは限らないという問題もある。

#### 2.2.2 検索履歴を用いたリランキング

リンク構造ではなく、各ユーザが検索結果に対して起こした行動をページの重要度の指標とし、ランキングを作成 (修正) する手法もある [6]。この手法はクリックスルーというデータを用いる。

クリックスルーとは、ユーザに提出された検索結果のランキングに対し、ユーザがどのページへのリンクをクリックしたかという情報である (図 3。このデータの特徴として、ユーザやシステムに大きな負荷を与えることなく、大量のデータを集めることができるということが挙げられる。また、情報とはとかく古くなりがちだが、新しいデータを集め続けることでタイムリーな話題についての情報も得ることができるという特徴もある。

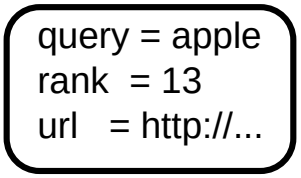


図 3: クリックスルーの例

[6] ではこのクリックスルーを用いてランキングのそれぞれのページの重要度を相対的に比較している。最近の検索システムがランキングに各ページへのリンク先のページに関する情報を載せていることを利用する。例えば検索結果のランキングのうち、あるユーザが 1 位と 3 位と 7 位をクリックしていた場合、このユーザは最低でもランキングの 7 位までのリンクの情報を見ていた可能性が高い。その中でこの 3 つのページへのリンクをクリックしたということは、3 位のページは 2 位のページよりもこのユーザにとって重要だと判断したことになり、7 位のページは 2,4,5,6 位のページよりも重要だと判断したということが推測できる。これから 3 位のページは 2 位のページよりも上位にランクすべきという制約条件と 7 位のページは 2,4,5,6 位のページよりも上位にランクすべきという制約条件が取得できる。

この制約条件を収集したデータから大量に取得する。各ページを高次元空間上にマップしてからそれぞれの制約条件を可能な限り違反しないようなベクトルを決定し、そのベクトルに各ページに射影することでランキングを決定するというものである。

この手法では過去に検索をしたユーザが良さそうだという判断をした情報が統計的に活かされるが、やはり前述の手法と同様にクエリが同じならばユーザがだれであろうと結果のランキングは同じものが提供される。また、[6] では実験の際、各ページを高次元空間にマップするために、それらのページが Google などの検索エンジンで上位に入っているかどうかというような情報や各ページの URL とクエリとの類似度などを用いている。従って各ページの URL が内容を多少でも表していたり、各検索エンジンに検索されるようなページでなければこの手法を用いることが困難である。しかしながら低コストで大量に取得できるクリックスルーに関してはランキングに応用する方法が他にもあると考えられる。そこでこのクリックスルーを用いてユーザの嗜好に合わせたランキングを作成する手法を考える必要がある。

### 2.2.3 ランキングが作りにくい環境

ここまで情報検索におけるランキング手法について、現在主流となっている検索システムをいくつか述べてきた。これらのシステムはユーザに用いられている頻度を考えると、精度の善し悪しはともかく、ある程度ユーザを満足させているということも推測できる。しかしこれらのシステムは必ずしも万能という訳ではない。なぜならこれらのシステムを用いることの不可能、あるいは困難な環境も存在するからである。

一般的に成功している検索システムを用いることが困難な例として、まずサイト内検索が挙げられる。サイト内検索とはある一つの Web サイトを検索することである。サイト内検索に対して、複数の Web サイトや Web 全体から検索することをグローバル検索と呼ぶことにする。

グローバル検索で成功しているシステムの特徴はページ間のリンク構造を使っているということであった。しかしサイト内検索ではこのリンク構造をそのまま利用しようとしても精度の良いランキングを作成することができないことが多い。

その理由として、一つの Web サイト内のリンクはページスコアを反映するほど強い繋がりを持たない。リンクの数そのものが少ない上に、リンクされているページ間の関連もサイト内のリンクについては弱い。そもそもサイト内ではグローバル検索の概念である「多数のページに参照されているページが重要なページである」ということが必ずしも成り立たない。index ページや、help ページには高いスコアがつくが、これらはユーザにとって本当に必要なページとは限らない。したがってリンク構造を用いる検索システムをサイト内検索に適用しようとしても、リンクによる結び付きが弱いためスコアが付けづらいという問題と、スコアがついたページについてもその精度が悪いという問題が生ずる。

ランキングを作成する検索システムとしてクリックスルーを用いる手法も紹介した。これはリンク構造は用いていない。しかしこれをサイト内検索に適用するときにも問題は生ずる。まず、そのサイト内検索を用いるユーザ数が少ないため、よく使われる極めて一般的なクエリでもなければ十分なログが集まらず、そういったクエリのランキングにしか手法を適用できないということである。

[6] は各検索システムの順位なども用いてランキングを決定しているが、検索システムがサイト内全てのページを網羅していないという問題もある。ロボット型検索システムはリンクを辿ってページを探すため、リンクが張られていないページの情報までは得られない。しかしそのようなページにもユーザが必要としている情報が存在する可能性は低くない。

このような問題に対し、[7] ではサイト内検索の

精度を向上させるために、Web サイトのログから相関ルールを抽出、集約し、それをランキング作成に用いている。しかしこの手法も個人の嗜好の違いに対応しておらず、クエリが同じならば結果のランキングも同じものが返ってくる。また階層化されたディレクトリ構造でなければルールの抽出ができないという問題もある

近年電子図書館など文献情報のデジタル化が進んでいる。文献は大量に存在するため、ユーザが能率良く欲しい文献を絞り込むには精度の良い検索システムが必要となる。電子図書館はサイト内検索と言えるが、さらに特化した特徴を持っている。サイト内のリンク構造がほとんど存在しないこと、サイト外からサイト内へのリンクが張られないため、グローバルな検索システムでスコアを算出できないこと、各文献が階層的に管理されていないことである。さらに各文献の情報がタイトルや著者などといった簡略化された形式で載せられているため、同一文献にキーワードが複数回出現することが少ない。従って情報検索で用いられる TF/IDF の類の手法も用いることが困難となる。

このようなことからサイト内検索は未だに単純な全文検索のみで行われ、特に電子図書館においては 50 音順や出版年でしかランキングを作成できないのが現状である。しかしユーザの負担を考えるとグローバル検索と同等の精度を持つようなランキングを作成することが必要である。むしろ、あるテーマについてのページが集められたサイト内からユーザ個人が必要な情報を絞り込むという意味では、サイト内検索の方が高い精度のランキングを返すことが望まれる。したがって、従来の検索システムやリンク構造に依存しないランキング手法で、ユーザ個々が必要としている情報、文献をランキングの上位に位置させるような検索システムが必要である。

# 3　協調フィルタリング

2.1 節で情報検索の問題点として検索過程が独立しているため、情報収集のために行った作業過程や結果が他のユーザに活かされないという問題を述べた。ここでは過去の自分の情報や他のユーザの情報を情報収集に活かす手法である協調フィルタリングという技術について述べる。

## 3.1　協調フィルタリングの概念

協調フィルタリングとはユーザが興味、関心を持った事柄に対して収集した情報を、他に同様の興味、関心を持っているユーザに対して提供することで、ユーザの情報収集活動を支援するという仕組みである。この仕組みによって自分と似たようなユーザが過去に収集した情報のなかでも、有用とされた情報を得ることで、情報収集の能率を上げることが期待できる。この協調フィルタリングは「同じユーザならまた同じ情報を好むであろう」ということと、「似たユーザならば同じような情報を好むであろう」という仮定の元に、それぞれのユーザにとって必要な情報を推定し、それを推薦するというものである (図 4)。

協調フィルタリングでは各ユーザの嗜好を表すために、過去に情報収集のために起こした行動に関する情報や、収集した情報に対する評価などをプロファイルと呼ばれるものに蓄積する。ここで情報を収集しようとしているユーザをアクティブユーザ、アクティブユーザと嗜好が似ているユーザを近隣と呼ぶことにする。プロファイルを蓄積することにより、アクティブユーザのプロファイルと他のユーザのプロファイルを比較することで、似たような嗜好を持つユーザを近隣として特定する。近隣が過去に収集した情報の中からアクティブユーザに興味がありそうなものを推定して、アクティブユーザに推薦する。

協調フィルタリングでは人間が持つ興味、関心などは個人個人によって異なるものではあるが、完全にそれらが一致することはなくとも、部分的

にならば似たような興味や関心を持つ人がいるということを前提にしている。

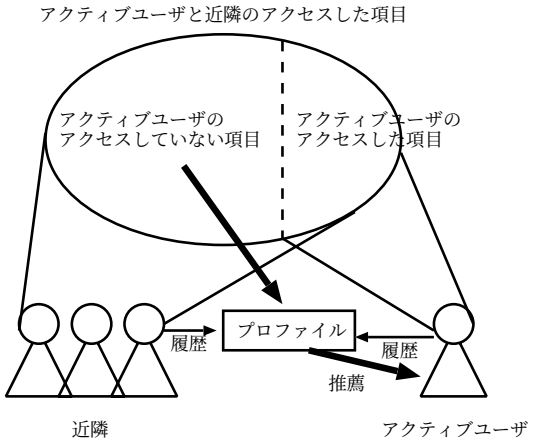


図 4: 協調フィルタリングの概念

こういった協調フィルタリングの手法は大規模なものでは Amazon[2]や、NetNews においては GroupLens[8] などで用いられており、主に商品や記事などにおいてユーザがまだ知らないものでも興味を持ちそうなものを薦めるという目的で用いられている。

協調フィルタリングには様々なバリエーションが存在する。大まかに分けると次の二つになる。ユーザの行動や実際に項目につけた評点などを蓄積しておき、それを元に評点予測や情報推薦を行う memory-based collaborative filtering と、確率的な観点からユーザの評点の値を予測し、それを元に情報推薦を行う model-based collaborative filtering である。これら二つの手法の中にも様々なバリエーションが存在するが、ここでは省略する。

## 3.2 プロファイル

前述の通り、協調フィルタリングでは情報推薦のためにユーザの嗜好に関する情報を収集し、プロファイルという形で記録する。当然推薦はこのプロファイルに基づいて行われるため、いかにユーザの嗜好をプロファイルとして正確に表現、蓄積できるかが重要となる。ユーザの嗜好を表現できなければ精度の良い推薦は行えない。

しかし精度が良ければ良いという単純な話でもない。システム全体を考えると、ユーザそのものや項目に対するユーザのデータは大量に得られることが必要である。しかし収集する際にユーザに大きな負担を課すようなデータを大量に収集することは難しい。また大量に収集することを考え、システムに負荷を与えるようなデータを収集することも望ましくない。したがって、ユーザやシステムに負担を与えずに、できるかぎりユーザの嗜好を表せるようなデータをプロファイルに蓄積することが求められる。

協調フィルタリングに関する研究では、プロファイルに蓄積する情報としてユーザの項目に対する評価や評点を収集するものがある。ユーザの評価は明示的なものと暗示的なものの二つに分けることができる [9]。明示的な評価とはユーザが項目に対しての実際に点数をつけた評点などをいう。明示的評価の特徴としてはユーザの意図を正確に表していることや、評価が変わりにくいこと、そしてユーザに負担を与えてしまうことなどが挙げられる。一方暗示的な評価とはユーザの潜在的に持っている興味を示唆する可能性があるものとして、ユーザのマウスの動きや、閲覧時間など、それだけでは明確な評価にはならないが、ある程度興味の有無を推測できるものを指す。暗示的評価は、ユーザに負担を与えずに収集できること、複数の評価を組み合わせて精度を上げることができること、明示的評価よりも精度が落ちることなどが特徴として挙げられる。

[8] では、ユーザが読んだ記事に対する評点を蓄積し、推薦を行っている。これは明示的な評価であるため、ユーザにある程度の負担を与える。一方 [9] ではページの閲覧時間やスクロール時間など暗示的な評価を収集し、分析を行っている。また [6] ではプロファイルという形ではないが、クリックスルーをログとして収集している。このデータ

[2]http://www.amazon.com/

はプロファイルという形で収集しても、ユーザに負担を与えずに大量のデータを収集できる。本研究ではこのクリックスルーをプロファイルに蓄積する。詳しくは次章で述べる。

### 3.3 評点予測

プロファイルにユーザの嗜好を表す情報を蓄積したら、それを元にアクティブユーザが未評価の各項目 (アイテム) に対する評点予測を行う。ここでは本研究でも用いている user-based collaborative filtering の評点予測について説明する。因みにこの評点予測法は前述の memory-based collaborative filtering の中の一種である。

各ユーザのプロファイルがベクトルで表されているとする。アクティブユーザを $a$, ユーザ $i$ のアイテム $j$ に対する評点を $v_{i,j}$ とすると、アクティブユーザ $a$ のアイテム $j$ に対する評点 $p_{a,j}$ は次式で予測される。

$$p_{a,j} = \bar{v_a} + \alpha \sum_{i \in N} w(a,i)(v_{i,j} - \bar{v_i}) \qquad (1)$$

ここで、$\bar{v_a}, \bar{v_i}$ はユーザそれぞれの評点の平均値、$\alpha$ は正規化のための値、$N$ は近隣である。$w(a,i)$ にはアクティブユーザとその他の各ユーザとの距離、相関関係や類似度などを用いる。この値は手法によって算出法が異なる。代表的なものは次の二つである。

**相関関係**

ユーザ $a$ とユーザ $i$ の相関関係は以下の式で算出される。

$$w(a,i) = \frac{\sum_j (v_{a,j} - \bar{v_a})(v_{i,j} - \bar{v_i})}{\sqrt{\sum_j (v_{a,j} - \bar{v_a})^2 \sum_j (v_{i,j} - \bar{v_i})^2}} \qquad (2)$$

$j$ はユーザ $a$ とユーザ $i$ の両方が評価をした全てのアイテムである。この算出法ではそれぞれのユーザが共に評点をつけた項目に対してどの程度評価が似ているかを数値化する。ユーザが同じアイテムに対して、共に高い評価をすればするほど、また共に低い評価をすればするほどこの値は 1 に近い高い値をつける。また、両方のユーザに関連性が見られない場合は 0 に近い値となり、嗜好が逆の場合は-1 に近い値となる。この手法はユーザが評価時に段階をつけることができるような場合、すなわち数値の評点をプロファイルとして蓄積した場合などで有効となる。

**類似度**

ユーザ $a$ とユーザ $i$ の類似度は以下の式で算出される。

$$w(a,i) = \sum_j \frac{v_{a,j}}{\sqrt{\sum_{k \in I_a} v_{a,k}^2}} \frac{v_{i,j}}{\sqrt{\sum_{k \in I_i} v_{i,k}^2}} \qquad (3)$$

$j$ は全てのアイテムである。$I_a, I_i$ はそれぞれのユーザが評価をしたアイテムの集合である。この算出方ではプロファイルに蓄積してあるユーザベクトルのコサインの値を算出している。こちらも相関関係同様ユーザが同じアイテムに対して共に高い評価をすればするほど値は大きくなる。しかしこの類似度では各ユーザの平均評点値を用いていない。したがってユーザから段階をつけた評価が直接得られない場合、すなわち評点ではなく、「あり」か「なし」かのような 2 値的な評価しか得られない場合でも用いることができる。

相関関係の場合でも類似度の場合でも算出された $w(a,i)$ の値が高くなるようなユーザ $i$ が近隣となる。この近隣が $N$ となって (1) 式に用いられる。

この user-based collaborative filtering は [8] などで用いられているが、Amazon では item-based collaborative filtering という手法が用いられている [10]。これは近隣として似ているユーザを探す変わりに、全アイテム同士の相関関係や類似度を求める。その値を用いてアクティブユーザが過去に購入、参照したアイテムに強く類似するアイテムを推薦するという手法である。類似度がユーザに依存しないため、あらかじめ計算しておくことが可能であり、レスポンスタイムが早いという特徴がある。

協調フィルタリングではこのような流れで評点予測を行い、情報推薦を行う。

### 3.4 リランキングに用いる際の問題点

情報検索システムの問題点である「検索過程が独立」という問題に対して、これまで述べてきた協調フィルタリングという手法を用いることは、過去にユーザが情報収集に費した作業を活かすことが可能となる。また、ユーザ個々の嗜好に合わせたランキングを作成することで、ユーザの情報収集作業を軽減することにも繋がることが考えられる。しかしその際に問題点がいくつか生じる。

まずどのような情報をプロファイルに蓄積するかという問題である。特に検索システムではユーザの評点の類のものは大量に集めることが難しい。ユーザやシステムに負担を課す情報は集められない。またサイト内検索のような条件の厳しい環境でも大量に集められるような情報でなくてはならない。

次にプロファイルに蓄積した情報からどのように近隣を特定するかという問題がある。この近隣にアクティブユーザと嗜好の近いユーザを選定することができなければ、ユーザにとって有用なランキングを作成することは困難である。

最後に近隣が特定できた場合、どのようなページをランキングの上位に位置させるかという問題がある。一般的に有用とされるページよりも、アクティブユーザにとって有用なページをランキングの上位に位置させることが極めて重要である。

## 4 検索履歴を用いたリランキング手法

本章では本研究で提案する検索履歴を用いたリランキング手法について述べる。本手法では情報検索システムの問題点である、検索過程が独立であることと、検索結果が個人個人の嗜好に合わせられていないということに対して、プロファイルという形でユーザの嗜好を表し、それに合わせて検索結果を修正する。また協調フィルタリングの手法を適用することで他のユーザの情報を検索結果の修正に用いることも行う。これにより過去になされた情報収集作業を活かし、ユーザ毎に異なる検索結果を提示することが可能となる。

また、ユーザの嗜好情報を表すために各ユーザの検索履歴であるクリックスルーを用いる。この情報を用いることで既存の手法では十分に効果を発揮できないような環境でもランキングの精度を上げることが可能となる。以上のような手法により、ユーザは能率的な情報収集を行うことが可能となる。

はじめに、提案するリランキング手法のモデルについて述べる。そのあと各処理でどのような処理を行うかについて具体的に述べる。

### 4.1 リランキングモデル

リランキングモデルは次のようになる。あらかじめ各セッションごとにプロファイルを作成しておく。プロファイルについては次節で述べる。

アクティブセッション a がクエリ Q で検索する場合:

1. a がシステムにクエリ Q を入力
2. サーチエンジンを用いて Q の検索結果 R を取得
3. a の近隣 N を特定
4. N のプロファイルを用いて a のプロファイルを拡張
5. 拡張したプロファイルを持ちいて検索結果 R を修正
6. 修正したランキング R' を a に提示

図 5 はこのモデルを図示したものである。なお、アクティブセッションも近隣の一人と見なすこともできる。

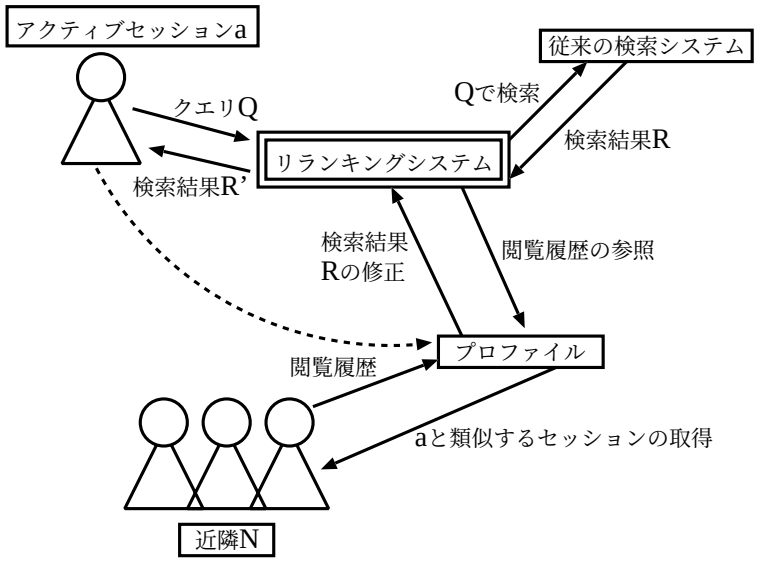


図 5: リランキングモデル

前章まで推薦を受ける対象をアクティブユーザとしてきた。しかし本研究ではユーザという単位ではなく、セッションという単位を対象に推薦を行う。その理由については次節で述べる。ここからは推薦を受ける対象をアクティブセッションと呼ぶことにする。クエリ Q はキーワードの集合である。サーチエンジンを用いるとあるが、これは特定のサーチエンジンではなく、任意のものである。ドキュメント集合全体からクエリ Q が出現するドキュメントに絞り込むために用いている。なお、他のセッションの情報を用いない場合、3,4 の処理は行わない。

次節からそれぞれの処理で具体的にどのような手法を用いるかについて説明する。

## 4.2 プロファイル

プロファイルはアクティブセッションの近隣を特定することと、検索結果を修正する際、アクティブセッションにとって各ページがどの程度有用であるかどうかを推定することに用いる。近隣にはアクティブセッションと嗜好や興味が似ているセッションを特定しなければならない。したがってプロファイルには各セッションで作業をしたユーザの嗜好を表すような情報を蓄積することが必要となる。

しかしながら 3.2 節で述べた通り、ただ精度が良い情報を用いれば良いという訳ではない。人間の嗜好は多種多様であることから、部分的にでも嗜好が類似するユーザを特定するためには大量の情報が必要となる。そのためにはシステムを使用するユーザに余計な負担を与えてはならない。余計な作業を課すことによりユーザが減ることも考えられるし、そもそもユーザの情報収集作業量を軽減するという本研究の目的に矛盾する。また、システムに負荷を与えるような情報も収集するべきではない。システムに負荷がかかれば、それがユーザの負担に繋がるからである。このようなことを考慮した上で精度の良い情報を収集することが必要となる。

協調フィルタリングを用いる recommender system ではユーザの評点を用いることが多い。これはユーザの嗜好を段階のついた数値で直接表すため、良い精度が期待できる。しかし評点を入力すると

いう作業を負担に感じるユーザも多い。このような問題点を解決する方法として、ユーザの検索履歴、具体的には検索システムに入力した検索クエリやその結果の中から実際にクリックしたページへのリンク情報、すなわちクリックスルーをプロファイルとして収集することも有効な手段であると考えられる。

この情報はユーザやシステムに余計な負担を与えることはない。実際、検索システムではサーバーログとしてこのような情報を記録しているシステムが大半である。また、ほとんどのユーザはこのような情報が収集されていることを知らずにシステムを使用しているということも事実だ。そうでないユーザも、そのようなことを知っていてもそれが当り前のこととしてシステムを使用している。

このようなことからクリックスルーは簡単に収集できるものである。しかし、ユーザが過去に入力したクエリや、閲覧したページの集合のみからでは、近隣を特定する情報としては不十分である。なぜなら過去に同一のクエリで検索したユーザや、同一のページを閲覧していたユーザが重なる可能性が低いからである。同一項目に対するユーザの評価を得られない限り、近隣を特定するための指標となる「類似度」などの値を算出することは困難となる。

そこで本研究では、ユーザが過去に閲覧したページに出現するキーワードと、その出現頻度を重みとしてそれぞれプロファイルに蓄積する手法を用いる。ここでいうキーワードとは Yahoo[3]のような検索サイトに存在するカテゴリ名のようなもので、それぞれのページのテーマを表すようなキーワードとする。具体的にはニュース、天気、スポーツ、ファイナンスなどが挙げられる。

このキーワードをプロファイルに蓄積することで、各ユーザがどのような話題のページに興味があるのかということを推定することが可能となる。また、これらのキーワードの出現頻度も収集しておくことで、それぞれの話題についてどの程度の興味があるかという度合を示す値も取得することが可能となり、特に強く興味を持っているのはどういう話題かを特定することが可能となる。

プロファイルに蓄積するキーワードはページがテーマごとに分類されている検索システムであれば、その分類のテーマをそのままキーワードとして用いれば良い。分類されていない場合、各文献における単語の出現頻度を調べ、TF/IDF による重み付けを行い、その重みの大きいものを選択するという手法や、ページのクラスタリングを行い、各クラスタに頻出するキーワードをそれぞれ調べ、それをキーワードとする手法などが考えられる。

この情報をプロファイルに蓄積しておくことで、ユーザ同士で過去に共通して閲覧したページが存在しなかったとしても同じようなテーマのページを見ているかどうかで、彼らの嗜好が近いか遠いかを比較することが可能となり、その内で特に嗜好が近いユーザを近隣として特定することが可能となる。

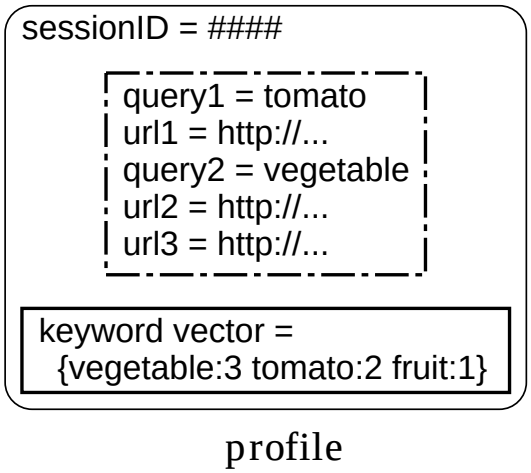


図 6: プロファイルの例

クリックスルーと閲覧ページに出現したキーワード及びその頻度を組み合わせてプロファイルに蓄積することで、ユーザやシステムに負担を与えず、的確な情報をプロファイルとして表すことが可能となる。本研究ではこれらの情報をセッション単

[3]http://www.yahoo.com

位で管理する。ユーザ単位ではなく、セッション単位で管理する理由の一つは人間の嗜好が移り変わり易いという特徴があるからである。大抵の検索システムはクリックスルーを含むユーザのログをセッション単位で管理している。ここでのセッションの単位は各検索システムに依存するが、ブラウザの開閉や一定時間で区切られているものが一般的である。それにより同一ユーザでも複数のセッションに区切られることもありうる。

しかし人間の嗜好、特にブラウジングにおけるそれは多様であり且つ時々刻々と移り気である。それならば長い時間の履歴をユーザ単位で管理するよりも、短いセッションという単位で管理する方が、一つの話題に対するセッション単位でのユーザの嗜好情報を正確に把握することができると考えられる。

また、ユーザ単位で管理することを考えた場合、複数セッションから同一ユーザのセッションであるかどうかを特定する作業が生ずる。ここにはプライバシーの問題が絡んで来る。複数セッションが同一ユーザのものであるかどうかを見分けるためには IP アドレスなどの情報が必要となる。これにより万が一情報が流出するような事態になれば閲覧情報がどのユーザのものであるか特定できてしまう可能性も生じてくる。IP アドレスなどの情報がない、セッション単位の管理であれば、万が一情報が流出してもそこから個人を特定することは難しい。このような理由からもユーザ単位のプロファイルの管理よりもセッション単位のプロファイルの管理の方が安全かつ有用であると言える。

本研究ではプロファイルにユーザの閲覧ページの履歴 (URL)、過去の閲覧ページに出現するキーワードとその出現頻度を組み合わせて蓄積する手法を用いる。これらの情報は大量に収集できる上に、オフラインで処理できるという利点もある。つまりユーザがあまり使用しない時間帯を選んでプロファイルを作成する作業を行えるということである。このような情報を蓄積することで簡単に精度の良いユーザの嗜好をプロファイルとして表現することが可能となる。

### 4.3 近隣

検索結果を修正する際、協調フィルタリングの手法を用いるにはアクティブセッションと嗜好が似ているセッションを特定し、近隣を特定する必要がある。それぞれのセッションが似ているかどうかを判断するために前節で述べたプロファイルをセッションごとに作成し比較する。

プロファイルに記述した情報のうち、キーワードとそれに対する重みをセッションごとに比較する。各セッションのプロファイルはベクトルの形で表現できる。このベクトルの類似度を次式から算出し、アクティブセッションと高い類似度となったセッションを近隣に選定する。アクティブセッション $a$ とセッション $i$ の類似度は

$$sim(a,i) = \frac{\sum_{c \in C} f_{a,c} \cdot f_{i,c}}{\sqrt{\sum_{c \in C} f_{a,c}^2} \sqrt{\sum_{c \in C} f_{i,c}^2}} \qquad (4)$$

で求められる。ここで $C$ はキーワード集合、$f_{a,c}, f_{i,c}$ はそれぞれ各セッションにおけるキーワード $c$ に対する重みである。

プロファイルに存在するキーワードは各セッションでそのセッションが割り当てられたユーザが閲覧したページから抽出されている。キーワードの重みは閲覧したページに出現した頻度であるから、その頻度が大きいキーワードについてはそのセッションが割り当てられたユーザが興味があった事柄であると考えられる。

したがって、同じキーワードに大きな重みをつけているプロファイルをもつセッション同士は、それぞれが同じある特定の分野のものに強い興味があるユーザ同士のセッションであると言える。そのようなユーザは嗜好が大いに似ていると考えられる。したがって (4) 式のコサイン距離でセッション間の類似度を算出することで、同じ分野に共通に強い興味があるユーザを近隣と特定することが可能となる。

協調フィルタリングを用いる研究では (2) 式の

ような相関関数を用いて近隣を特定するための指標としているものもある。しかしこの式では「同じ項目に良い評価を与えている」場合だけではなく、「同じ項目に悪い評価を与えている」場合でも強い関連があるとされる。この手法は評点のようなユーザのポジティブな反応とネガティブな反応の両方を評価として得られるような場合に有効ではある。しかし、本研究の場合、ネガティブな反応を評価として得ることは難しい。それはある分野のキーワードに対する重みが 0 であったとしても、その分野について興味がないとは断定できないからである。たまたまその分野のページを見なかった可能性は十分に考えられる。また、同じ分野に興味がないユーザならば同じ分野に興味があるということが成り立たないのは明白である。

以上から本手法では (4) 式により、アクティブセッションと同じ分野に強い興味をもつセッションを近隣として特定する。

### 4.4 スコア算出

ここではページのスコア計算について述べる。ページのスコアはアクティブセッションのユーザに有用と推定したページ程大きな値をつける。アクティブセッション $a$ に対するページ $p$ のスコア $S_p$ は次式で算出する。

$$S_p = \alpha\ rank(p) + \beta\ sim(a,p) \tag{5}$$

$\alpha$ と $\beta$ は定数である。もし元の検索システムのランキング精度が悪ければ $\alpha$ を 0 にする。またここでは $sim(a,p)$ でアクティブセッションとページとの類似度を算出している。まずページ $p$ に出現するキーワードとその出現頻度からセッションの場合と同様にページ $p$ をベクトルの形にする。次にセッションの場合と同様にアクティブセッションを構成するページとページ $p$ との類似度を次式で算出する。

$$sim(a,p) = \frac{\sum_{c\in C} f_{a,c}\cdot f_{p,c}}{\sqrt{\sum_{c\in C} f_{a,c}^2}\sqrt{\sum_{c\in C} f_{p,c}^2}} \tag{6}$$

この式によりアクティブセッションの嗜好に合ったページに大きな類似度がつき、スコアの値が大きくなる。この処理をアクティブセッションの入力したクエリの検索結果全てに行い、スコア順に並べることでランキングを修正する。

協調フィルタリングの手法を用いる場合は、あらかじめ近隣のプロファイルを用いてアクティブセッションのプロファイルを拡張しておく必要がある。プロファイルの拡張は次式で行う。$N$ を近隣集合とする。

$$\vec{a'} = \vec{a} + \frac{1}{|N|}\sum_{n_i\in N} sim(\vec{a},\vec{n_i}) * \vec{n_i} \tag{7}$$

各近隣のプロファイルのベクトル表現を、アクティブセッションのプロファイルのベクトル表現に加える。この際、アクティブセッションとの類似度の大小によって重みを付ける。このようにすることで、アクティブセッションのプロファイルが拡張され、アクティブセッションが過去に見たページや、アクティブセッションは過去に見ていないが、嗜好が類似するセッションが過去に見て、有用であると推定されたページに大きなスコアがつくものと考えられる。

本研究では以上のような手法を用いてランキングの修正を行う。この手法によりセッション単位でのユーザの嗜好にあったページをランキングの上位に位置させることが可能になる。既存の検索システムが最初に返す結果はランキングの上位 10〜20 件であり、また大抵のユーザはランキングの上位しか閲覧しないという傾向がある。このようなことから本研究がユーザの作業量を減らすことを支援すると考えられる。

## 5 実験

提案手法を実装し、筑波大学図書館 Tulips のログを用いて実験を行った。本章ではこの実験について述べる。更にどの程度ランキングの精度を向上させたかを評価する。

## 5.1 実験に用いるデータセット

本研究では筑波大学附属図書館で用いられているシステム Tulips(筑波大学電子図書館)[4]のデータを使用して実験を行った。データは本研究とは関係なく、普段から Tulips で収集しているデータである。

### 5.1.1 Tulips の概要

Tulips では筑波大学図書館で管理する文献について、ユーザが入力したクエリがタイトルや編集者、件名に出現する文献を絞り込みリストにして返す。このとき返される文献の順番はタイトルの順番 (アルファベット順、50 音順) となっている。一度結果が返されると編集者順、出版年順、順不同でリランキングすることもできる。また最初に結果として表示される文献数は 20 件で、10 件、50 件に変えることもできる。

検索結果のリストには各文献の情報ページへリンクが張られている。各文献の情報ページでは書名、著者名、出版、刊年、件名、分類、所蔵、ISBN の番号などが参照できる。

このようなシステムのデータを用いて実験を行った。データは次の二つのログを使用した。

### 5.1.2 Tulips のシステムログ

Tulips のシステムログはセッションごとに区切られている。各セッションのログから次のデータを取得し、実験に用いた。

- 入力クエリ
- クエリを入力した年月日、時刻

ログの一部を例として下に示す。

セッションでのクエリ入力一回が一行に対応する。10 月 1 日のセッション 0001 は 12 時すぎに「JAVA」、「C」、「JAVA Computer」、「Computer Science」の 4 回の検索をしたことになる。なお、複数単語で構成されるクエリに関してのログは (JAVA AND Computer) のように AND が入れられた形になっており、検索式と同じ表記となっている。

[4]http://www.tulips.tsukuba.ac.jp/

### 5.1.3 Tulips の Apache ログ

Tulips の Apache ログからは次のデータを取得し、実験に用いた。

- アクセスした文献 ID
- アクセス時刻
- セッションの ID、列数 (sessionSeq)

ログの一部を例として以下に示す。

一行目から三行目までが一つのログである。ID が 20041001.1003 のセッションが文献 ID136986 の文献を閲覧したことになる。この時の時刻が一行目の括弧の中からわかり、セッション列数が二行目から 9157 であるとわかる。

なお、IP アドレスも取得できたが本研究の実験では使用しなかった。また、クエリの検索結果を実際に Tulips のシステムを用いて取得し (タイトル順)、実験に用いた。

### 5.1.4 ログの分析

前節で述べたログを分析し、クリックスルーを抽出した。Tulips のシステムログと Apache ログとをセッション ID によってマッチングした。この作業によって、セッションごとのクエリリストと閲覧文献リストが得られた。さらに検索時刻や文献情報のアクセス時刻、セッション列数を考慮して、各セッションでどのクエリの結果としてどの文献をクリックしたかを特定した。

### 5.1.5 文献データ

クリックスルーからプロファイルを作成するために、各文献情報ページからキーワードを抽出し

(20041001.0001)
Oct 1 12:24:49 #1 TREE KEYWORD1=JAVA
Oct 1 12:26:12 #2 TREE KEYWORD1=C
Oct 1 12:26:29 #3 TREE KEYWORD1=(JAVA AND Computer)
Oct 1 12:33:04 #4 TREE KEYWORD1=(Computer AND Science)

(20041001.0002)
Oct 1 12:30:21 #1 TREE KEYWORD1=フランス革命
Oct 1 12:35:58 #2 TREE KEYWORD1=西洋美術

(20041001.0003)
Oct 1 12:32:10 #1 TREE KEYWORD1=(国語 AND 教育)
Oct 1 12:32:41 #2 TREE KEYWORD1=教育学
Oct 1 12:36:01 #3 TREE KEYWORD1=漢字

図 7: Tulips のログデータの例

195.251.###.### - - [01/Oct/2004:04:27:34 +0900] ”GET /cgi-bin/limedio/limewwwopac/book?sessionId=20041001.1003;sessionSeq=9157;sessionLang=eng;sessionCode=jis;bibid=136986 HTTP/1.1” 200 5492
222.5.###.### - - [01/Oct/2004:04:27:50 +0900] ”POST /cgi-bin/limedio/limewwwopac/search HTTP/1.1” 200 9530
222.5.###.### - - [01/Oct/2004:04:28:42 +0900] ”GET /cgi-bin/limedio/limewwwopac/magazine?sessionId=20041001.1004;sessionSeq=8427;sessionLang=jpn;sessionCode=utf8;bibid=15920 HTTP/1.1” 200 6978
130.158.###.### - - [01/Oct/2004:04:28:43 +0900] ”GET /cgi-bin/limedio/limewwwopac/book?sessionId=20041001.1001;sessionSeq=10533;sessionLang=jpn;sessionCode=utf8;bibid=2702384 HTTP/1.1” 200 6250
222.5.###.### - - [01/Oct/2004:04:28:56 +0900] ”POST /cgi-bin/limedio/limewwwopac/magazine HTTP/1.1” 200 6989
130.158.###.### - - [01/Oct/2004:04:28:59 +0900] ”GET /cgi-bin/limedio/limewwwopac/book?sessionId=20041001.1007;sessionSeq=10817;sessionLang=jpn;sessionCode=utf8;bibid=139632 HTTP/1.1” 200 5029

図 8: Apache のログデータの例

た。Tulips の文献情報ページには「件名」という項目があり、ここに文献の分野を表す単語 (政治、言語学、歴史、数値計算、精神療法など) がある。しかしこの情報は全ての文献に存在するわけではない。したがって本実験ではこの「件名」のみをキーワードとした場合と、「件名」と「書名」と「著者名」をキーワードとした場合とで実験を行った。なお、キーワードを抽出する際、「is」、「at」、「著」、「編」、特殊文字などのストップワードは除去した。これにより作成したプロファイルの一部を以下に記す。

2001 建築儀礼:2 船舶:2 橋梁:2 民間信仰:2
2002 民俗学:1 日本:1 風俗:1 習慣:1
2003 企業:2 日本:2 経営分析:1 便覧:1
2004 中小企業金融:1 税務会計:1 株式:1
2005 結晶成長:1 人工結晶:2 工業材料:3
2006 社会組織:1 社会学:1
2007 飼料作物:2 牧草:2
2008 アフリカ:2 教育:2 国際協力:1 論文集:1

左の 4 文字はセッション ID、それ以降はキーワードとそのキーワードを含むページを見た頻度である。ID が 2003 のセッションは「企業」、「日本」が出現するページをそれぞれ 2 件ずつ、「経営分析」、「便覧」が出現するページをそれぞれ 1 件ずつ閲覧したことになる。

## 5.2 実験手法

Tulips の 2004 年 10 月 1 日から同年 10 月 31 日までのログを使用し、実験を行った。セッション数は 116737 件でこのうち、件名のみからプロファイルを作成できたセッションは 23209 件、件名と書名と著者名からプロファイルを作成できたセッションは 51818 件であった。

セッションに出現した文献は 96003 件であった。このうち件名が存在するものが 37110 件、件名か書名か著者名のいずれかが存在するものが 95733 件であった。

実験のため、複数のクエリで検索を行ったセッションで、6 件以上の文献を参照した状態でさらにクエリ (以下テストクエリ) で検索を行い、その結果について 6 件以上の文献を参照したセッションをアクティブセッションとし、テストに用いた。アクティブセッション以外のセッションはそれぞれプロファイルを作成し、近隣の候補とした。アクティブセッションについては、テストクエリ以前の閲覧文献のみからプロファイルを作成し、テストクエリで提案手法のランキングを作成した。評価はアクティブセッションがテストクエリの検索結果で閲覧したページ (以下テストデータ) のランキングがどう変化したかで行った。

アクティブセッションは 166 件、テストデータは 1413 件。パラメータとして近隣の上限数を 10,20,50,100 と変えて実験を行った。また、アクティブセッションのみのプロファイルを用いての実験も行った。結果では近隣数 0 として表記した。

## 5.3 評価の手法

評価には次のような指標を用いた。

**順位向上率 (商)**

式：$\frac{\text{(Tulips の元順位)}}{\text{(各手法での順位)}}$

評価で用いるテストデータは各セッションで閲覧された文献であるから、このデータの順位を上位に修正することが最も重要である。したがって、順位向上率 (商) は 1 より大きい値となれば良い値といえる。値が大きくなればなるほど良いランキング手法といえる。例えば元順位が 30 位のものを 10 位にすれば 3 となる。この値をテストデータ全てについて算出し、平均を調べた。順位向上率 (商) はこのシステムを使用したときの効果の目安となる。

**順位向上率 (差)**

式：(Tulips の元順位) – (各手法での順位)

上で述べた順位向上率は順位同士の差で評価したが、差をとった値でも評価を行った。テストデータを上位に修正していればいるほど正の整数で大きな値となる。この値も全てのテストデータで平均を調べた。順位向上率 (差) は各手法を使用した時に、自分が必要としている文献が (現在のシステムを使用した場合と比べ) 何位上昇するか、下降するかの期待値となる。

**平均順位**

式：$\frac{\text{(テストデータの平均順位)}}{\text{(テストクエリの検索結果文献数)}}$

各セッションで閲覧された文献の平均順位がどの位置にあるかを示す。例えばテストクエリの結果が 100 件返ってくるクエリで、テストデータが 3 件あり、それぞれの順位が 25 位、30 位、35 位ならば、各順位の平均値は 30 位であるから、$30/100 = 0.3$ がこの指標の値となる。元のランキングよりも 0 に近くなればなるほど、ユーザが各セッションで閲覧した文献を上位に修正していることになる。また、ランキングをランダムで作成した場合、この値は 0.5 に近くなることが考えられるため、この手法で 0.5 より小さい値にならなければ良いランキング手法とは言えない。

**N 位以内率**

式：$\frac{\text{(N 位以内に入ったテストデータ数)}}{\text{(全テストデータ数)}}$

テストデータが N 位以内に入る確率である。各手法を使用したときに、自分の必要な文献の何%が N 位以内に入るかを示すものである。N が小さいときに値が大きくなることが重要となる。N は 1,10,20,50,100 とした。

以上の指標で従来のランキングと提案手法のランキングの比較を行った。また、文献の閲覧回数のみの情報で作成したランキングも作成し、比較を行った。

## 5.4 実験結果

ここでは実験結果について述べる。各評価手法において、文献情報の件名のみをプロファイル作成に用いたものを「提案手法 (件名のみ)」と表記し、プロファイル作成に著者名と書名を用いたものを「提案手法 (件名＋著者＋書名)」と表記した。また比較対象のクリック数でランキングを作成したものを「クリック数」と表記し、結果を載せた。順位向上率以外は元のランキングでの数値も算出し、「元ランク」と表記して結果を載せた。

### 5.4.1 順位向上率 (商)

順位向上率 (商) の結果は表 1 である。また、グラフを図 9 に示す。グラフの横軸が近隣数であり、縦軸が順位向上率 (商) である。順位向上率は大きくなるほど良い値である。

一番良い結果となったのが、「提案手法 (件名＋著者＋書名)」であり、「提案手法 (件名のみ)」が次に良い結果となった。「クリック数」は一番結果が悪かった。算出手法が全く同様の提案手法ではあるが、件名だけをプロファイルに用いた場合は、近隣数が多くなればなるほど値が良くなった。しかし著者名と書名も用いた場合は近隣数が 50 のときが最も良い値となり、それ以降は近隣数を増やすと値が僅かに悪くなった。

### 5.4.2 順位向上率 (差)

順位向上率 (差) の結果は表 2 である。また、グラフを図 10 に示す。グラフの横軸が近隣数、縦軸が順位向上率 (差) である。順位向上率は大きくなるほど良い値である。

一番良い結果となったのが、「提案手法 (件名＋著者＋書名)」であり、次が「クリック数」となった。3 つの手法で一番結果が悪かったのは「提案手法 (件名)」となった。著者名や書名をプロファイルに用いた場合は、近隣の数を増やせば増やす程良い値となった。しかし、プロファイルを件名

| 近隣数 | 0(ユーザ) | 10 | 20 | 50 | 100 |
|---|---|---|---|---|---|
| 提案手法 (件名のみ) | 3.549 | 3.641 | 3.616 | 3.754 | 3.777 |
| 提案手法 (件名＋著者＋書名) | 5.020 | 5.060 | 5.112 | 5.252 | 5.233 |
| クリック数 | 3.476 | | | | |

表 1: 順位向上率 (商)

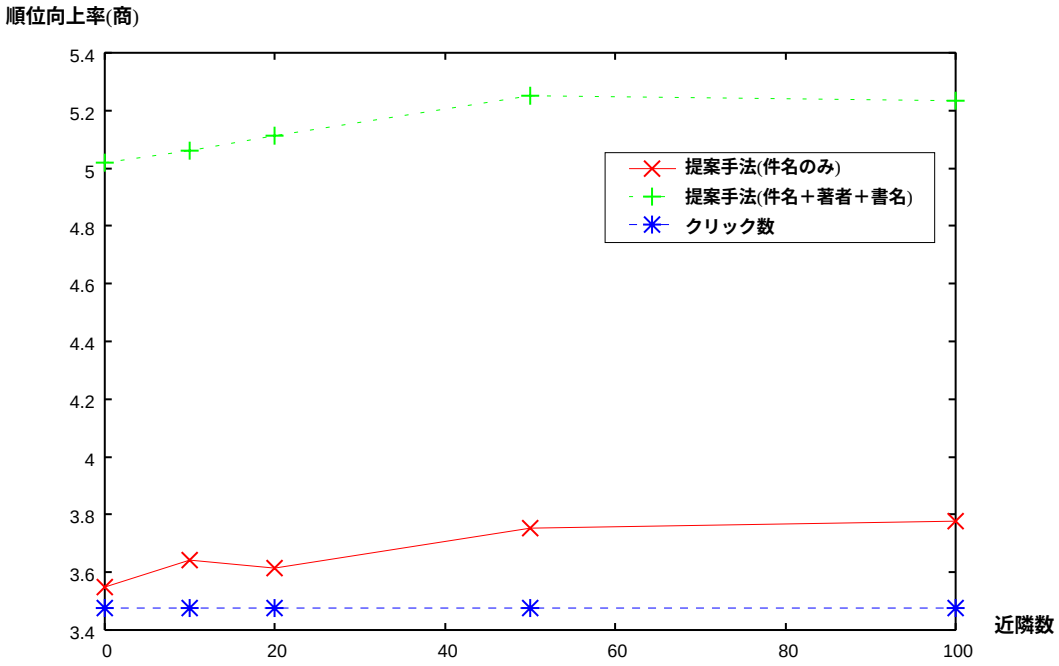


図 9: 順位向上率 (商)

| 近隣数 | 0(ユーザ) | 10 | 20 | 50 | 100 |
|---|---|---|---|---|---|
| 提案手法 (件名のみ) | 5.219 | 5.706 | 5.430 | 5.539 | 5.138 |
| 提案手法 (件名＋著者＋書名) | 14.58 | 15.08 | 15.36 | 15.71 | 16.37 |
| クリック数 | 7.443 | | | | |

表 2: 順位向上率 (差)

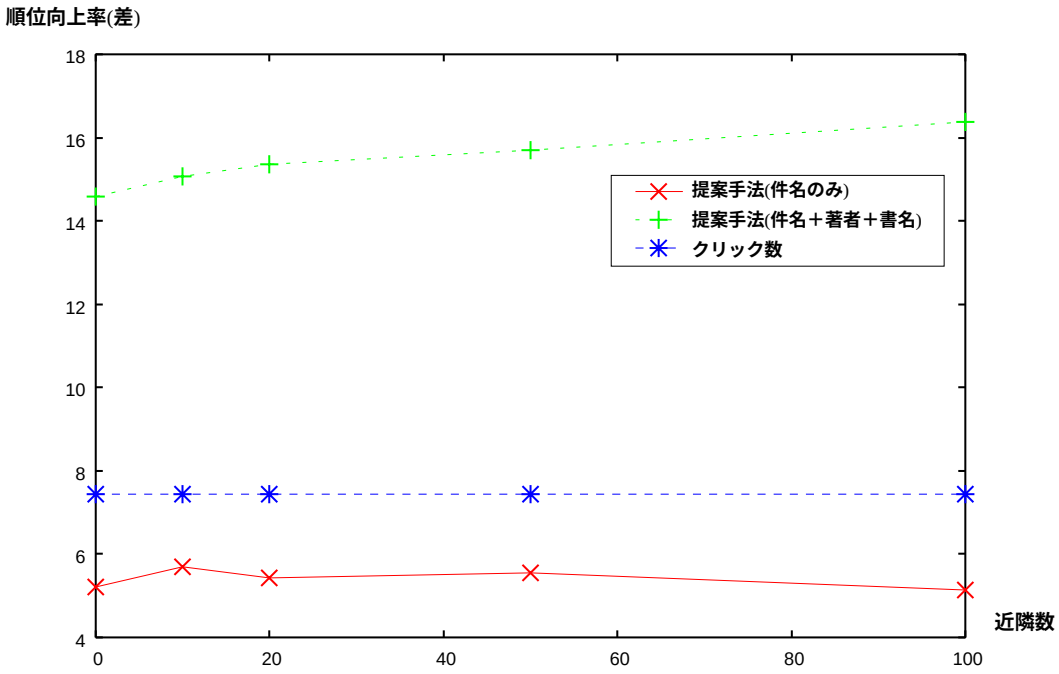


図 10: 順位向上率 (差)

情報のみから作成した場合は近隣数を増やしても良い値にならず、逆に僅かに悪くなった。

### 5.4.3 平均順位

テストデータの平均順位を適合文書全体のどの位置に修正したかが表 3 である。またそのグラフを図 11 に示す。横軸は近隣数、縦軸が平均順位の値となる。平均順位は元のランキングの値が 0.4574 なので、その値よりも小さければ平均的にテストデータのランクを上位に修正していることになる。

一番良い結果となったのが、「提案手法 (件名のみ)」であった。次が「クリック数」、「提案手法 (件名＋著者＋書名)」だが、この二つに大きな差は見られなかった。特に近隣数が 50 の場合はほぼ同じ値である。プロファイルに著者名と書名を用いた場合は近隣数が 50 のときが一番良く、それ以降は近隣数を増やしても値は悪くなった。一方プロファイルに件名のみを用いた場合は、近隣数を 100 までならば増やせば増やすほど良い値となった。また、3 つの手法は全て「元ランク」の平均順位よりも良い値となった。

### 5.4.4 20 位以内率

テストデータが 20 位以内に入る確率が表 4 である。また、グラフを図 12 に示す。グラフの横軸が近隣数であり、縦軸が 20 位以内率である。値は大きいほど良い。

一番 20 位以内率が高かったのが、「提案手法 (件名＋著者＋書名)」である。次に「クリック数」、「提案手法 (件名のみ)」とつづき、「元ランク」が一番低い値となった。著者名と書名を用いてプロファイルを作成した場合は他の指標と同様に近隣数が 50 を超えると値が悪くなりはじめた。この指標でも 3 つの手法は元のランキングよりも良い結果となった。

| 近隣数 | 0(ユーザ) | 10 | 20 | 50 | 100 |
|---|---|---|---|---|---|
| 提案手法 (件名のみ) | 0.3366 | 0.3257 | 0.3194 | 0.3163 | 0.3002 |
| 提案手法 (件名＋著者＋書名) | 0.4050 | 0.3951 | 0.3952 | 0.3882 | 0.3941 |
| クリック数 | 0.3868 | | | | |
| 元ランク | 0.4574 | | | | |

表 3: 平均順位

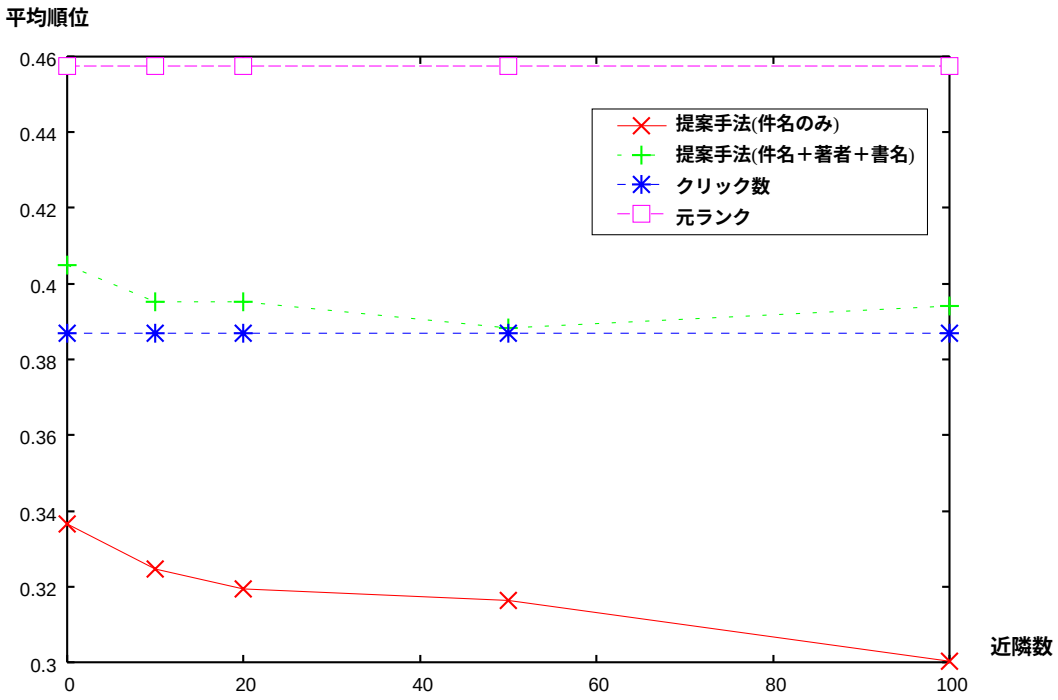


図 11: 平均順位

| 近隣数 | 0(ユーザ) | 10 | 20 | 50 | 100 |
|---|---|---|---|---|---|
| 提案手法 (件名のみ) | 0.4791 | 0.4728 | 0.4742 | 0.4728 | 0.4692 |
| 提案手法 (件名＋著者＋書名) | 0.4883 | 0.4989 | 0.5012 | 0.5067 | 0.4975 |
| クリック数 | 0.4874 | | | | |
| 元ランク | 0.4387 | | | | |

表 4: 20 位以内率

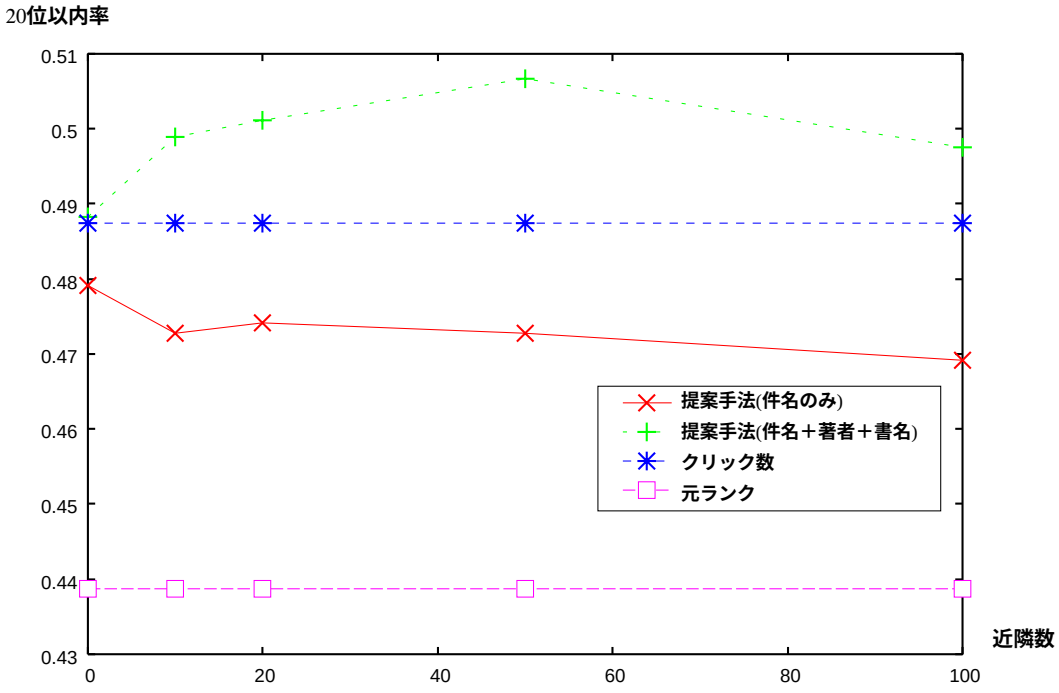


図 12: 20 位以内率

#### 5.4.5 N 位以内率

20 位以内率の 20 を 1,10,20,50,100 に変えて実験した結果が表 5 である。さらにグラフを図 13 に示す。グラフの横軸は N とし、縦軸が N 位以内率となっている。値は大きい程良い。なお、ここでの提案手法はどちらも近隣を 50 としたものである。

N の値にもよるが、良い値が出たのは「提案手法 (件名＋著者＋書名)」となった。特に 20,50,100 位以内率が一番良い値となった。次に良い結果となったのが「クリック数」と「提案手法 (件名のみ)」であった。「クリック数」は N が小さいときには一番高い値となるが、N が大きくなるにつれ「提案手法 (件名のみ)」が逆転した。「元ランク」は全ての N で一番悪い結果となった。また、どの手法でも N が大きくなるにつれ、N 位以内率も大きくなったが、その上昇度合はどんどん小さくなった。

# 6 考察

本章では前章の実験結果についての考察を行う。

## 6.1 各評価指標に関する考察

### 6.1.1 順位向上率

順位向上率 (商) では提案手法がクリック数でランキングしたものよりも良い結果となった。特にプロファイルに著者名と書名を用いた場合では大きな値となった。これは元ランクで順位の高かったテストデータの順位を下げずに、順位の低かったテストデータを多数上位に修正したためであると考えられ、精度の良いランキング修正が行えていると言える。

一方件名のみをプロファイルに用いた場合では著者名と書名を用いた場合程良い結果とはならなかった。その理由としてランキング修正が十分に行えなかったことが挙げられる。件名が存在する

| N | 1 | 10 | 20 | 50 | 100 |
|---|---|---|---|---|---|
| 提案手法 (件名のみ) | 0.0326 | 0.2902 | 0.4728 | 0.6971 | 0.8004 |
| 提案手法 (件名＋著者＋書名) | 0.0375 | 0.3270 | 0.5067 | 0.7233 | 0.8415 |
| クリック数 | 0.0411 | 0.3302 | 0.4874 | 0.6839 | 0.8003 |
| 元ランク | 0.0241 | 0.2519 | 0.4388 | 0.6596 | 0.7757 |

表 5: N 位以内率

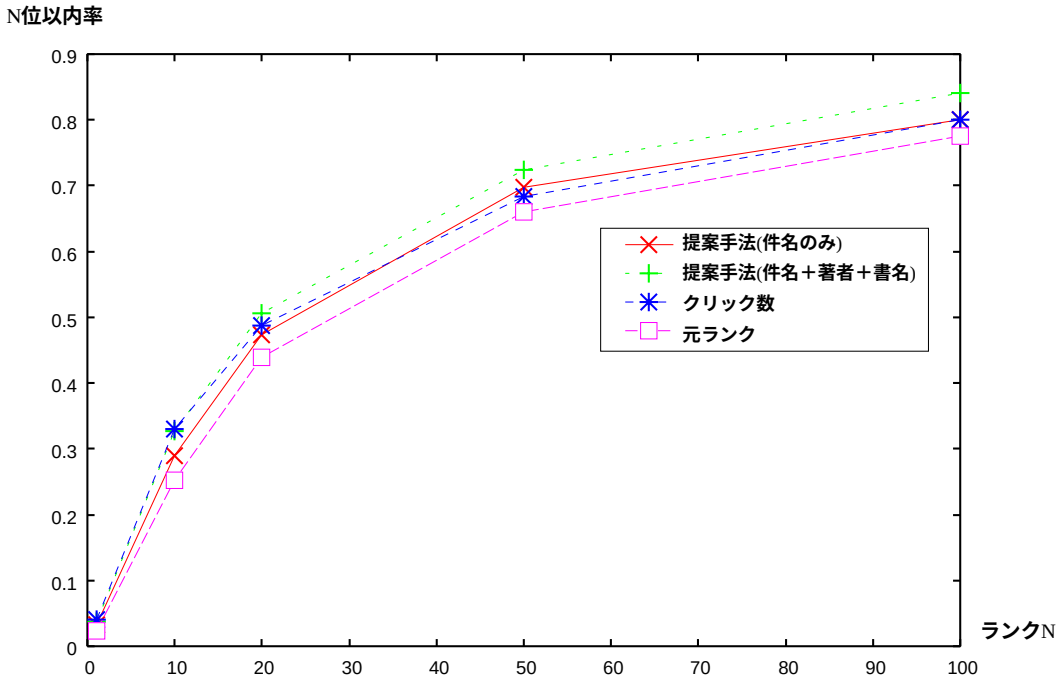


図 13: N 位以内率

文献は 5.2 節で示した数値の通り約 39%である。したがってテストデータの内にも同様の割合で件名が存在しない文献が含まれることになる。

順位向上率 (商) は低い順位のテストデータを高い順位に多数修正することで大きくなる指標である。つまり良い方に修正した回数が値に大きく関わる。したがって文献情報が約 39%しか取得できなかった件名と、99%を超える文献情報を取得できた著者名と書名を使用した場合や順位修正対象が 100%となるクリック数とでは順位を修正する対象の文献数が異なってくる。よってクリック数と値に大きな差が見られない「提案手法 (件名のみ)」ではあるが、順位修正対象の文献数の少なさを考慮すれば、精度の良いランキング修正が行えていると考えることができる。

順位向上率 (差) はテストデータを平均で何位上位に修正したかという値である。これについても順位向上率 (商) と同様のことが言える。ほとんどのテストデータの文献情報が得られる「提案手法 (件名＋著者＋書名)」の値が抜群に良くなった。その一方で「提案手法 (件名のみ)」は値が小さい。これも文献情報が半分弱しか得られなかった影響がでている。順位向上率 (差) もテストデータの低い順位を高い順位に修正する精度と同時に回数が大きく関わるためである。したがって、「提案手法 (件名のみ)」はクリック数よりも値は低いが、提案手法はどちらも精度の良いランキングを作成したと考えられる。

順位向上率と提案手法の近隣数との関係であるが、近隣数が 50 までであれば、近隣数を増やせば増やすほど良くなる傾向にある。「提案手法 (件名のみ)」の近隣数が 10 の場合の値が高いが、傾向としては近隣数は多い方が良い。しかし、近隣数を 100 とすると精度が落ちる場合がある。これは近隣数を増やしすぎるとプロファイルを拡張する際に、アクティブセッションとは関係のないキーワードもプロファイルに記述されるためである。したがって順位向上率の指標からでは近隣数は 50 とするのが良いと考えられる。以上から提案手法は順位向上率の指標からは良い精度のランキング修正を行っていると考えられる。

### 6.1.2 平均順位

平均順位はテストデータの平均値が適合文書数と比べ、どの位置にあるかという指標である。また、この指標では文献情報が取得できなかったテストデータは計算に入れていない。したがって、純粋な手法のランキング精度を表す指標といえる。

「元ランク」の値は 0.4574 であった。つまりテストに使用したセッションで閲覧されたページはランキングにあまり依存することなく、内容で選ばれていると言える。このことは重要である。一般的に検索システムの結果は上位しか見られない傾向にある。そういう意味で本研究で用いたデータは信頼できるデータであると言える。

結果が一番良かったのはプロファイルに件名のみを用いたもので、他の手法の値とは大きな差が見られた。このことはランキングの精度が最も良いということに他ならない。具体的にはもし適合文書が 100 件あるクエリで検索したときに、ユーザが必要とするページが元のシステムでは平均順位 45 位で返されるのに対し、この手法では平均順位が 30 位で返されることになる。大きくランキングの精度を向上させていると言える。

このような結果になった理由としては件名という文献の情報を的確に表したキーワードで各セッションのプロファイルを作成したことにより、それが各セッションでのユーザの嗜好を正確に表すことができたことに他ならない。また、協調フィルタリングの手法も効果を発揮した。ユーザのプロファイルのみでのランキング修正した場合での平均順位の値は 0.34 であるが、近隣数を 100 として協調フィルタリングの手法を用いた場合での平均順位は 0.30 まで向上した。

一方で著者名や書名もプロファイルに使用した場合の値は、元ランクの値よりは向上したものの、クリック数でランキングを作成した場合よりも悪

くなった。この理由としてプロファイルに余分な情報が混じってしまったことが考えられる。今回の実験では文献情報抽出時に 20 語程度の stopword の除去を行っただけで、十分な整形作業を行っていない。例えば著者の名字の片仮名表記であったり、普段の情報検索実験では除去するような stopword もプロファイルに入っている。それらの影響でセッションのプロファイルに余分な情報が入り込み、ランキング修正の精度が落ちてしまったと考えられる。

クリック数によるランキングの結果についても述べておく。元ランクの値よりもかなり良くなっているが、これにも理由がある。Tulips の文献は膨大である。したがって、一つの文献を何度も閲覧されるということは少ない。特に今回の実験に用いたログの期間である一か月ではなおさらのことである。クリック数にはアクティブセッションの閲覧履歴は当然除いてあるが、Tulips のセッションは短い。したがって、同じユーザが時間を置いて、あるいは他の日に、前の作業の続きとして同じ文献の情報を閲覧した場合、クリック数にカウントされてしまう。つまりクリック数の値は本来のその手法で算出されるべき値よりも良い値になりやすい傾向があるということである。テストデータに対するクリック一回がランキングに大きな影響を及ぼすという特殊な環境ゆえ、このクリック数の値の偏りについては考慮すべきである。

平均順位と提案手法の近隣数との関係であるが、アクティブセッションのプロファイルを正確に表している場合のみ近隣数を増やす程良い結果となった。これはプロファイルを正確に表しているならば、本当に嗜好が類似するユーザを近隣と特定できるからである。因みに「提案手法 (件名のみ)」の近隣数を 200 とした場合は 0.2950 となり、近隣数を 100 とした場合よりも良い結果が得られた。一方、プロファイルに余分な情報が入り込んでしまうと、近隣を増やすことで、その余分な情報まで拡張されてしまう。それによって「提案手法 (件名＋著者＋書名)」の近隣数 100 の場合の結果が悪くなったと考えられる。

以上から平均順位では「提案手法 (件名＋著者＋書名)」はクリック数よりも低い値となったが、前述のクリック数の値の偏りを考慮すれば、良い結果と言える。元ランクとの比較ならば歴然である。そして質の良いデータをプロファイル作成に使用できるならば、「提案手法 (件名のみ)」の結果から本手法で精度の良いランキングが作成できたと言える。

### 6.1.3 N 位以内率

N 位以内率では N の値により、一番良い結果となった手法が異なった。元ランクの値よりは他の 3 つの手法の値の方が良い結果となった。

N が小さい場合、クリック数が一番良い値となった。この理由としては平均順位で述べたことがここでも挙げられる。もしアクティブセッションと同じユーザが他のセッションで同じ文献を閲覧した場合、クリック数が増える。文献一件ごとのクリック数が少ない (0 の文献も多い) ため、クリック数が一回増えることにより大幅な順位の上昇が起こる。したがって上位 1 件や上位 10 件では高い値となりやすい。その反面 N を 20 よりも大きくすると値が悪くなるのもその影響である。

N を 20 以上にすると、「提案手法 (件名＋著者＋書名)」が一番良い値となる。ランキングの修正により、各セッションで必要な文献をランキングの上位に修正していることがわかる。クリック数と比較すると、特に 11 位以降から 100 位以内にテストデータの文献を上位に修正していると考えられる。

「提案手法 (件名のみ)」は「提案手法 (件名＋著者＋書名)」に比べると、低い値であるが、N を大きくしていくと、クリック数の値よりも良くなる。この指標においても前述の件名取得率の問題が絡んでくる。手法としては修正するべき文献のランクが、件名情報がないことにより修正できないということである。そのような環境で元ランクより

も常に良い値となっていることから、作成しているランキングの精度は良いと言える。

20 位以内率の指標から、近隣数について言えることは、近隣数は 20 から 50 までとするのが良いということである。やはり近隣数は多すぎると余計な情報が混ざるということが考えられる。「提案手法 (件名のみ)」では近隣数が 0、すなわちセッションのプロファイルのみを用いた場合が一番良い結果となった。セッションの嗜好がプロファイルに正確に表されているとも考えられるが、N が 10 で値が悪くなり、N が 20 で値が良くなっていることから、誤差の範囲とも考えられる。

以上から N が小さいときにはクリック数の方が良い値がでているものの、精度のよいランキングを提案手法で作成していると考えられる。

## 6.2 提案手法の有効性

前節で各指標ごとの考察を行った。その結果ランキングの精度、順位向上率とも提案手法のいずれかが良い結果となった。N 位以内率についてもデータの偏りを考慮すれば、提案手法 (件名＋著者名＋書名) の方が良い結果であったと言える。

本研究での実験ではプロファイルに件名のみを用いた場合と、著者名と書名も用いた場合とで実験を行った。提案手法で述べた通り、プロファイルには文献のテーマを表すようなキーワードを蓄積するのであるから、本来ならば件名のみを用いるべきである。しかし、今回用いたデータには件名が存在しない文献が 60%以上存在した。そこで著者名と書名を用いてほとんど全ての文献についてプロファイルを作成できるようにした次第である。

ランキングの純粋な精度という観点では平均順位の指標からも「提案手法 (件名のみ)」が一番良い。値も他との差は大きかった。これは重複するが、各セッションのユーザの嗜好を正確に表すことができたからである。しかし件名のみを用いた場合、順位向上率や N 位以内率は平均順位ほど良い値にはならなかった。

順位向上率や N 位以内率はテストデータを上位に修正した数が影響する。提案手法 (件名＋著者＋書名) は、書名や著者が存在しない文献が少ないことから、ほぼ全ての文献が順位修正の対象である。事実この手法では元ランクやクリック数といった、今回の実験環境で適用可能な既存の手法のランキングよりも良い結果となった (図書館の文献情報では TF/IDF すら難しい)。プロファイルの精度が平均順位で落ちていることがわかっているにも関わらずである。

以上から本手法は従来の手法よりも有用な手法であるといえる。また、プロファイル作成に使用するデータを文献のテーマを表すという意味で質の良いデータをプロファイルに用いることでランキングの精度を上げることができ、さらにそのようなデータを多くの文献から得ることで、高い順位にユーザにとって必要な文献を多数修正することができると言える。

ランキングの精度と、上位にユーザの必要な文献を多数存在させることを両立するためには、質の良い情報を全ての文献から取得できればよい。例えば今回の実験で言えば、件名が全文献についていればその実現が可能であったと考えられる。

本研究の手法は電子図書館のみではなく、通常の大半のテキスト検索システムに適用することが可能である。一般的な検索システムは本研究のクリックスルーにあたるログを収集している。また検索の効率化のため、各ページのキーワードをインデックス化して管理している。このキーワード情報は全てのページから取得していると考えられ、またこのキーワードはページのテーマを表すものである。したがって、このようなシステムに本研究の手法を適用することで精度の良いランキング修正が行え、ランキングの上位にユーザにとって必要なページを修正できると考えられる。また従来の検索システムではランキングを作成しづらいサイト内検索のような環境でも、本研究の電子図書館における実験と考察で示した通り、精度の良いランキング修正が行えると考えられる。

最後に本研究の実験では差を明白にするため、純粋にプロファイルと文献情報の類似度でランキングを作成した。そして元のランキングに重みをつけてそれも考慮するということは行わなかった。もし順位の修正候補を絞り込む検索システムの返す結果の精度が良い場合 (例えば Google など)、その順位をスコアの算出に含めることも考えられる。これにより PageRank が高く、しかもユーザに合ったページのみを検索結果の上位に修正することが可能になると考えられる。

以上に述べた手法でランキング修正を行うことにより、精度の良いランキングを作成し、ユーザの情報収集のための作業量を減らすことが可能になると考えられる。

## 7 おわりに

本稿では検索履歴を用いたリランキング手法を提案した。

この手法はプロファイルを用いてユーザの嗜好情報を表し、それによってランキングを修正するというものである。プロファイルには従来の検索システムのほとんどが収集している、セッションごとのページ閲覧履歴を蓄積する。これにより従来の検索システムではランキングを作成しづらかった環境においても、精度の良いランキングを作成することが可能となり、またユーザの嗜好に応じたページをランキングの上位に修正することが可能となる。

実験では、提案手法によるランキングの修正を行い、修正されたランキングと修正前のランキングとの比較を行った。実験結果からランキングの改善に関する考察を行うとともに、本研究の手法の有効性を確認した。

今後の課題としてはより精度の向上が見込まれるランキング関数の検討が挙げられる。

ユーザがまだ詳しくない分野について検索を行うことを想定すると、今までのユーザの嗜好に合ったページよりも、PageRank が高いような一般的なページ (その分野の初心者向けページなど) をランキングの上位に修正した方が良いこともある。

このような場合を考慮し、従来の検索システムの PageRank などのスコアと、本手法によって算出されるスコアとの両方を考慮して、ランキングすることにより、更なるランキングの精度の向上が期待できると考える。

## 参考文献


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

[2] L.Page, S.Brin, R.Motwani and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web, Stanford Digital Library Technologies Project, 1998.

[3] J. Kleinberg. Authoritative Sources in a Hyperlinked Environment, Proc. of the 9th ACM-SIAM Symp. on Discrete Algorithms, 1998.

[4] S. Brin and L. Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine, Computer Network and ISDN Systems, Vol. 30, No. 1–7, 1998.

[5] J. Wang, Z. Chen, L. Tao, W.-Y. Ma, and L. Wenyin. Ranking User's Relevance to a Topic through Link Analysis on Web Logs, Proc. of the 4th Int'l Workshop on Web Info. and Data Management, Nov. 2002.

[6] T. Joachims. Optimizing Search Engines using Clickthrough Data, Proc. of the ACM Conf. on Knowledge Discovery and Data Mining, 2002.

[7] G.-R. Xue, H.-J. Zeng, Z. Chen, W.-Y. Ma, and C.-J. Lu. Log Mining to Improve the Performance of Site Search, Proc. of the 3rd Int'l. Conf. on Web Information Systems Engineering, Dec. 2002.

[8] P. Resnick, N. Iacovou, M. Sucker, P. Bergstrom, and J. Riedl. GroupLens: An Open Architecture, Proc. of the 1994 ACM conf. on Computer Supported Cooperative Work, 1994.

[9] M. Claypool, P. Le, M. Wased, and D. Brown. Implicit Interest Indicators, Proc. of Int'l. Conf. on Intelligent User Interfaces, 2001.

[10] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based Collaborative Filtering Recommendation Algorithms, Proc. of the 10th Int'l. World Wide Web Conf., May 2001.

[11] J. S. Breese, D. Heckerman, C. Kadie. Empirical Analysis of Predictive Algorithms for Collaborative Filtering, Proc. of 14th Annual Conf. on Uncertainty in Artificial Intelligence, July 1998.

[12] S. M. McNee, I. Albert, D. Cosley, P. Coparlkrishnan, S. K. Lam, A. M. Rashid, J. A. Konstan, and J. Riedl. On the Recommending of Citations for Research Papers, Proc. of the 2002 ACM Conf. on Computer Supported Cooperative Work, 2002.

[13] J. Han, J. Pei, and Y. Yin. Mining Frequent Patterns without Candidate Generation, Proc. of 2000 ACM SIGMOD Int'l. Conf. on Management of Data, 2000.

[14] L. Terveen, J. McMackin, B. Amento, and W. Hill. Specifying Preferences Based On User History, Proc. of CHI 2002, Apr. 2002.

[15] R. Kumar, P. Raghavan, S. Rajegopalan, and A. Tomkins. Recommendation Systems: a Probabilistic Analysis, Proc. of IEEE Symp. on Foundations of Computer Science, 1998.

[16] J. B. Schafer, J. A. Konstan, and J. Riedl. Meta-recommendation Systems: User-controlled Integration of Diverse Recommendations. Proc. of the Conf. on Information and Knowledge Management, Nov. 2002.