DEIM Forum 2011 D4-5

ファイルとWebページの共起頻度に着目した関連性抽出手法の評価

宋　　強† 渡辺　陽介†† 横田　治夫†††

† 東京工業大学工学部開発システム工学科　〒 152-8552 東京都目黒区大岡山 2-12-1
†† 東京工業大学学術国際情報センター　〒 152-8552 東京都目黒区大岡山 2-12-1
††† 東京工業大学大学院情報理工学研究科計算工学専攻　〒 152-8552 東京都目黒区大岡山 2-12-1
E-mail: †{soukyou,watanabe}@de.cs.titech.ac.jp, ††yokota@cs.titech.ac.jp

あらまし　近年、インターネットとストレージ技術が目覚ましいスピードで進歩しており、Web とファイルシステム内に保存されている情報量も多くなってきた。Web を利用しながら、ファイル編集を行うことは日常的になっている。以前の作業で作成したファイル群と閲覧した web ページ群を遡ってもう一度両方を同時に見たいという要望が高まっている。本研究では、同一作業に用いたファイルと Web ページを関連づけて、一つのセットとして、ユーザに提示することを目的とする。このために、アクセスログを利用して、ファイルと Web ページの共起頻度に着目し、両者の関連性を抽出する手法を提案する。さらに実験を通して、提案手法を評価する。
キーワード　ファイル検索、Web ページ、関連性抽出、共起頻度

Evaluation of Relationship Extraction Methods between Web Pages and Files based on Co-Occurrence Frequency

Qiang SONG†, Yousuke WATANABE††, and Haruo YOKOTA†††

† Department of International Development Engineering, Tokyo Institute of Technology　2–12–1 Ookayama, Meguro-ku, Tokyo, 152–8552 JAPAN
†† Global Scientific Information and Computing Center, Tokyo Institute of Technology　2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552, Japan
††† Department of Computer Science, Graduate School of Information Science and Engineering, Tokyo Institute of Technology　2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552, Japan
E-mail: †{soukyou,watanabe}@de.cs.titech.ac.jp, ††yokota@cs.titech.ac.jp

Abstract　Due to the progress of Internet and storage technologies, there are lots of information on both the Web and file systems. Therefore, it has become common for users to edit files while searching information on the Web. Sometimes, users want to find both the files and web pages which were accessed while editing those files. In this researsh, we tackle to make groups which consist of the related files and web pages used in the same task. We analyze both the web and file access logs, and extract relationship between files and pages by considering co-occurrence frequencies of them. We present the proposed methods and the result of our experiments on real data.
Key words　file searching, web page, relationship between web pages and files, co-occurrence frequency

1. はじめに

近年, インターネットの発展によって、Web 上に保存されている情報量が大幅に増加している。同様に、個人のファイルシステム内に保存されているファイルの数も増えている。Web を利用しながら、ファイル編集を行うことが日常的になってきている。例えば、Web 上からプログラミング情報を探しながら、プログラムを書いた場合を考える。後でプログラムを再利用しようとする際に、プログラムファイルだけではなく、当時見た Web ページももう一度見たいという要望が考えられる。ファイル単体または Web ページ単体であれば既存技術により個別に検索可能であるが、しかし、この両者の間の関連性が分からないと、双方を関連づけて提示することはできない。つまり、ファイルと Web ページの間の関連性を抽出することが、検索結果の改善において非常に重要である。

ファイルシステム内のファイルを対象とした、デスクトップ検索 [1] [2] [3] が、既に普及してきている。しかし、あるファイルを編集したときによく見ていた Web ページを提示するとい

うような両者の関連付けは行えていない。それぞれを個別に検索できるだけである。

本研究の目的は、ファイル群と Web ページ群の間の関連性を抽出し、同一作業の両者を関連づけて、一つの仮想ディレクトリの形式で、ユーザに提示することである。ユーザが仮想ディレクトリを見ることによって、ファイルシステム内に存在する同一作業に属するファイル群と Web ページの URL 群を発見することができる。ここでの「作業」というのは、一つの仕事単位を意味する。例えば、卒業論文執筆作業では、卒業論文ファイルを作成するために作成した各種の画像、表ファイルなどと情報収集するために、閲覧した Web ページを含むべきである。

本稿では、ファイルアクセスログと Web アクセスログの 2 種類のログデータからファイル群と Web ページ群の関連性を抽出する二つの手法 **Pre-Merge 法**と **Post-Merge 法**を提案し、さらに実データを使用した評価実験を通して、提案手法の性能を比較評価する。

本論文の構成は以下の通りである。2. 節では、今回提案する二つの手法 Pre-Merge 法と Post-Merge 法について解説する。3. 節では実データを用いた評価実験を行う。4. 節では関連論文の紹介を行う。5. 節では、まとめと今後の課題について述べる。

2. 提 案 手 法

本研究ではアクセスログからファイル群と Web ページ群の関連を抽出する二つの手法、Pre-Merge 法と Post-Merge 法を提案する。本節で具体的な処理過程について説明する。

2.1 アプローチ

本研究の目的は、同じ作業に属するファイル群と同時に参照した Web ページ群を関連づけることである。ここでは、ファイル編集と Web ページ参照が頻繁に共起する組合せほど関係が強いと考える。

このような関係を抽出するために、本研究では、アクセスログの中で頻出する項目の組合せを探索する、頻出アイテム集合抽出 [4] を行う。しかしながら、本研究が対象とするログはファイルアクセスログと Web アクセスログ (プロキシログ、ブラウザログ) の 2 種類があるため、アプローチとして次の 2 つが考えられる。

- **Pre-Merge 法**: ログデータの段階で、ファイルアクセスログと Web アクセスログを統合し、統合されたアクセスログに対して頻出アイテム集合抽出を適用する。
- **Post-Merge 法**: ファイルアクセスログと Web アクセスログそれぞれに対して、個別に頻出アイテム集合抽出を適用し、最後に得られた頻出アイテム集合同士を時間情報に基づいて関連付ける。

以降、2.3 節では Pre-Merge 法を解説し、2.4 節では Post-Merge 法を解説する。

なお、両手法の内部処理において、アクセスログをトランザクションに変換し、頻出アイテム集合を抽出する処理として、我々の研究グループが以前に提案した、ファイルアクセスログからの関連抽出手法 FI 法 [5] をベースに用いている。ただし、FI 法はファイルアクセスログのみが対象であるが、本研究では Web アクセスログをも対象としている点が異なる。

2.2 アクセスログ

本研究で用いるアクセスログは、ファイルの読み書き等を記録したファイルアクセスログと、Web ページのアクセスログの２種類である。

ユーザのファイルの使用履歴情報はファイルサーバのアクセスログに記録されている。本研究で使用したファイルアクセスログはファイルサーバ側で記録ツール FaccLog [6] を利用して記録した。ログ中に記述されている情報として、ユーザの接続時刻、アクションネーム、ユーザ名、サーバ IP、クライアント IP、ファイルパスなどがある。

Web ページのアクセスログの取得方法はさらに 2 種類ある。http proxy (squid [7]) サーバから取ったプロキシログと、各クライアント PC 側にインストールされたブラウザ Firefox [8] においてアドオン Boomtango [9] から取ったブラウザログである。両者の違いは下の通りである。

プロキシログの利点

- 特定のブラウザに依存しない。プロキシ経由のすべてのマシンの活動を記録する

プロキシログの欠点

- ユーザの意図しない記録が多く存在する。例えば、一つのページをロードする時に、そのページに含まれるすべての画像ファイルがそれぞれ別々の記録としてログに残っている。このような記録も含むことから、プロキシログはログサイズが大きくなる傾向がある
- クライアントマシンのブラウザの設定および Web サーバの設定によって、ブラウザのキャッシュにヒットするとブラウザがサーバにデータ更新の問い合わせをしないため、プロキシログに記録を残さないことがある

ブラウザログの利点

- ユーザが実際にクリックした Web ページだけを記録する
- 余計な記録がないため、ログファイルサイズが比較的に小さい

ブラウザログの欠点

- ブラウザごとに、記録内容が違う。例えば、firefox では、あるページのアクセス時刻の履歴を全て保存するのに対して、chrome [10] では、各ページの最終アクセス時刻だけを保存する

3. 節の評価実験においては、プロキシログ、ブラウザログの両方を用いている。

2.2.1 ログフィルタリング

ログデータにはユーザの意図しない記録が大量に存在するため、ファイルアクセスログと Web アクセスログをそれぞれ別々にログフィルタリングをかけ、不要なログ情報を除去する。

ファイルの方では、予めに特定拡張子リストを作成しておいて、リスト上にある拡張子を持つファイル以外の操作記録を削除する。ログにはフォルダへのアクセス記録も大量に存在するが、この処理によって、これらの記録を除くことができる。次に、一秒間の短い期間に大量のファイルアクセスが記録されて

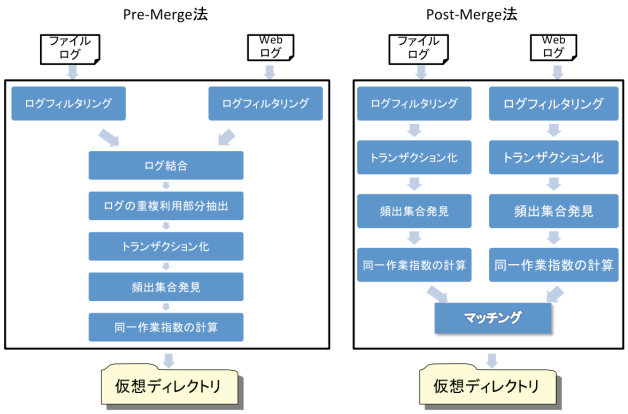

図 1　処理手順 (左)Pre-Merge,(右)Post-Merge

いた場合、これらは利用者の操作によるアクセスではなく、プログラムによる自動化された処理だと考えられる。そこで、一秒間に N(実験では N=4) 回以上のアクセス記録を持つ条目を削除する。これによって、ウィルス対策ソフトのファイルスキャンによるの大量のファイルアクセスなどを除くことができる。

一方、Web アクセスログの方では、ユーザの利用習慣から考えると、ユーザが目的のページに到着するまでに、ホームページからいくつかのページリンクを経由することが想像できる。この過程において、途中の不要なページを見る時間は比較的短く、クリックしてから、すぐに次のページをクリックすることが考えられる。よって、Web アクセスログに対して、あるアクセスから、3 秒以内に他のページのアクセス記録があったら、今のページを目的ページを辿る途中に経由したページだと判断して、削除する。次に、除外サイトリストを用いて、特定サイトへのアクセス記録を除く。例えば、Gmail などの Web Mail のユーザによる自分メールボックスへのアクセスなどは、作業とは無関係に頻繁に起こりやすく、またこれらは個別の検索機能も充実しているため、今回の解析対象からは除外している。プロキシログの中には、ページ内の各要素ごとに、記録がログファイルに残ってしまうため、画像 (.jpg/.JPG/.png/.PNG/.jpeg/.JPEG) ファイルの記録を除く。また、HTTP 応答のステータスコードが 200 以外の記録も削除する。

2.3　Pre-Merge 法

Pre-Merge 法では、ログデータの段階で、ファイルアクセスログと Web アクセスログを統合し、統合されたアクセスログに対して頻出アイテム集合抽出を適用する。処理手順を図 1(左) に示す。以下で各ステップごとに述べる。ログフィルタリングについては 2.2.1 節で述べた方法で行なっている。

2.3.1　ログ結合

ログの結合処理のステップでは、まず、両ログファイルのフォーマットを統一する。次に、両ログファイルの各アクセスをアクセス時間順にソートし、時間順にマージしていく。

2.3.2　ログの重複利用部分抽出

ユーザがファイル編集作業を行っていないときに、単に Web 閲覧をする場合も考えられる。ファイルと Web ページの関連

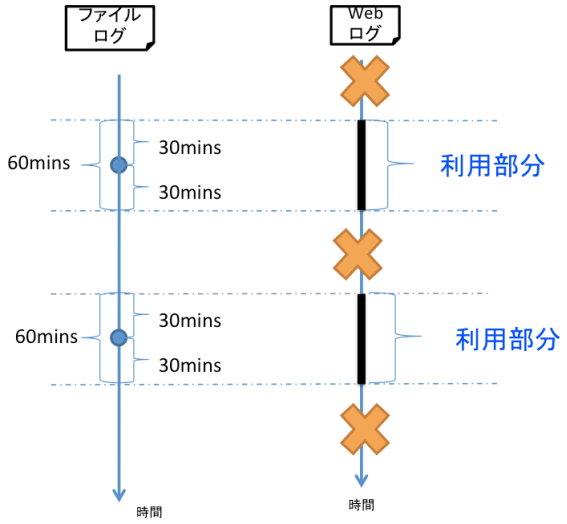

図 2　ログ重複利用部分抽出

を抽出することが本研究の目的であるため、ここでは、ファイルアクセスを行った時点の前後 30 分、計 60 分の間以外の Web 記録を除く (図 2)。これには解析対象とするログのデータサイズを削減し、処理時間を短縮するという効果がある。

2.3.3　トランザクション化

アクセスログに対して頻出アイテム集合抽出を適応するために、アクセスログを一定期間 (パラメータ TransactionTime[s]) ごとに分割し、その各区間で使われたファイル群を 1 つのトランザクションとして変換する。

例えば、「1:02:04　w.docx」「1:05:28　www.example.com」「1:40:39　y.txt」「2:04:06　z.pptx」のようなアクセスログが与えられた場合に、パラメータ TransactionTime を 3600[s] とすると、変換結果としてトランザクション 1 =「2.docx, www.example.com, y.txt」、トランザクション 2 =「z.pptx」が得られる。

2.3.4　頻出集合発見

トランザクション群から、アプリオリアルゴリズム [4] を用いて、頻出集合を抽出する。アプリオリアルゴリズムで指定された一定回数 (パラメータ MinSupport) 以上に出現した組合せを頻出アイテム集合として発見する。これによって、偶然に共起しただけで本来は無関係なファイルと Web ページの組合せを排除することができる。

2.3.5　同一集合指数の計算

この時点の結果として、ある集合が他の集合の部分集合になっている場合がある。このままでは利用者にとって扱いづらい。そこで、本研究では更に、頻出集合たちを極大化する。

例えば、「w,x」,「x,y」,「w,x,y」のような結果が存在する場合、極大集合を求める事によって、小さい集合を除き、集合「w,x,y」が得られる。

「x,y,z,a」と「x,y,z,b」が頻出すると分かった場合、「a,b」は直接に組み合わせとしては頻出しないが、「x,y,z」と同じ作業に属していると考えられる。よって、頻出アイテム集合抽出によって得られた集合同士の関連度によって、関連度が高い集合

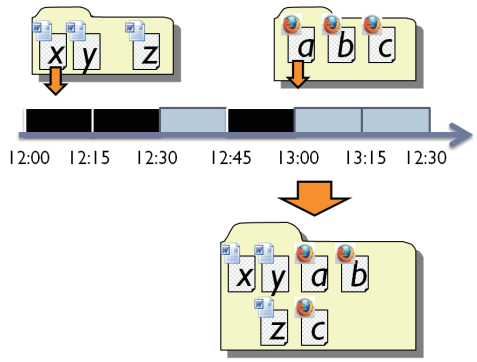

図 3　Post-Merge 法におけるマッチング

同士を結合する。類似の度合いの計算には Dice 係数を用いる。集合 A と B に対する、Dice 係数を用いた類似集合の結合条件は以下の通りである。

$$\frac{2|A \cap B|}{|A| + |B|} \geqq threshold$$

Dice 係数がパラメータ threshold の値以上の集合のみを結合する。

2.4　Post-Merge 法

Post-Merge 法では、ファイルアクセスログと Web アクセスログそれぞれに対して、個別に頻出アイテム集合抽出を適用し、最後に得られた頻出アイテム集合同士を時間情報に基づいて関連付ける。処理手順を図 1(右) に示す。Post-Merge 法では、ログフィルタリング、トランザクション化、頻出集合発見と同一作業指数のステップについて、Pre-Merge 法と共通であるからここでは省略し、最後のステップであるマッチングについてのみ説明する。

2.4.1　マッチング

先の段階で得られたファイルアクセスと Web アクセスのそれぞれの頻出アイテム集合抽出の結果集合をマッチングする方法について解説する。ここでは、関連があるファイル群と Web ページ群がほぼ同じ時間帯に利用されることに着目し、両方の集合の各要素のアクセス時間のオーバーラップ度合いに基づいて、マッチングを行う。

例えば、図 3 では、ファイルの頻出集合「x,y,z」と Web ページの頻出集合「a,b,c」が求められたとする。TransactionTime は 15 分とする。ファイル x が　12:00-12:15 ,12:15-12:30 ,12:45-13:00 の三つの時間帯 (図 3 の黒色の部分) に使われており、同様に、Web ページ a も黒色の三つの時間帯においてアクセスされていた場合、集合 F と集合 W を関連の強い集合同士と判断し、マッチングする。

ここで、ファイル頻出集合 F と Web ページ頻出集合 W のマッチング条件は以下の通りである。集合 F の要素 f_p の出現するトランザクション ID を $TranIDs(f_p)$ と表記し、集合 W の要素 w_q の出現するトランザクション ID を $TranIDs(w_q)$ と表記する。

$$|TranIDs(f_p) \cap TranIDs(w_q)| \geqq overlap$$

2 つの集合の各要素の任意のペアにおいて、同じトランザクションに共起した回数が一定回数 (パラメータ overlap) 以上であった場合、2 つの集合を同一作業とみなして一つの集合に結合する。

表 1　実験データ

ユーザ	期間	ログ種類	サイズ (byte)	ファイル・ページ数
ユーザ 1	2010/09/24〜	ファイル	23,624,947	472
	2010/11/21	ブラウザ	2,036,315	1731
		プロキシ	52,332,311	3939
ユーザ 2	2010/10/1〜	ファイル	42,450,684	506
	2010/10/31	プロキシ	68,546,676	3036

3.　評 価 実 験

3.1　実験の目的

実験の目的は、Pre-Merge 法と Post-Merge 法の最適パラメータを決定することと、両手法を精度と実行時間の二つの面から比較・評価することである。

3.2　実 験 環 境

提案手法を適用する実験環境を以下に示す。

- OS Linux CentOS 4.3
- CPU Dual Core AMD Opteron(tm) Processor 280 *4
- Memory 16GB (4GB*4)

3.3　実験データ

実験で使ったログデータは、我々の研究グループで用いているファイルサーバ上で FaccLog を用いて記録したファイルアクセスログと、グループ内で運用している http proxy サーバで記録したプロキシログ、各ユーザごとに記録した firefox のブラウザログの 3 つである。

今回の実験では、2 ユーザのアクセスログを用いた。実験データの詳細を表 1 で示す。

更に、ファイルアクセスログと Web アクセスログの組み合わせとして、下の三つのデータセットを定義した。

- データセット 1: ユーザ 1 のファイルアクセスログとブラウザログ
- データセット 2: ユーザ 1 のファイルアクセスログとプロキシログ
- データセット 3: ユーザ 2 のファイルアクセスログとプロキシログ

この三つのデータセットを定義した理由として、データセット 1 とデータセット 2 は同じユーザのファイルログと違う種類の Web アクセスログの組み合わせであり、ブラウザとプロキシを用いてどう違うかが分かる。一方、データセット 2 とデータセット 3 のデータの種類は同じであるが、ユーザが違うため、違うユーザ同士にどんな違いが存在するかを調べることができる。

今回は正解セットとして、人手によりファイルとページを含む 4 つの集合を作成した。一つの正解セットが一つの作業に対応する。正解セットの詳細を表 2 で示す。

表 2　正解セット

ユーザ	セット	内容	代表的拡張子	ファイル数	Web ページ数	合計
ユーザ 1	S_1	コア部分の実装	java, pptx	24	1	25
	S_2	前処理部分の実装	pl, txt	8	14	22
	S_3	輪講資料作成	pdf, pptx	2	7	9
ユーザ 2	S_4	学会運営関連	bmp, pdf, xlsx doc, docx, pptx	14	4	18

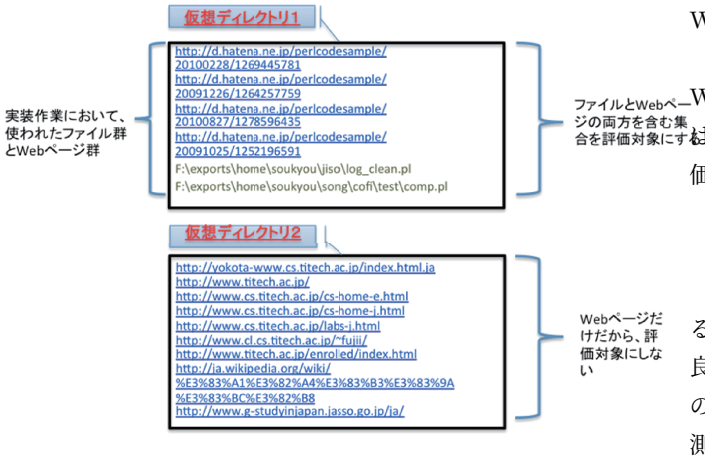

図 4　出　力　例

3.4　出　力　例

提案手法により発見された、ファイルと Web ページからなる仮想ディレクトリの出力例を示す。図 4 のように、一つの仮想ディレクトリが一つの作業を意味する。図 4 では、仮想ディレクトリが二つ生成されている。

仮想ディレクトリ 1 はユーザ 1 の実装作業において、作成された perl ファイル群と参照した perl プログラミング使用例を載せている Web ページ群である。既存のディスクトップ検索では、仮想ディレクトリ 1 の中の 2 つ perl ファイルを検索することは可能であるが、同時に参照した四つの perl プログラミング使用例を持つ Web ページを同時にユーザに提示することはできない。このような仮想ディレクトリを提供できることが本提案手法のメリットである。

仮想ディレクトリ 2 は Web ページ群をしか含まない。このような結果が出力された原因として、ユーザが Web 閲覧する間に、たまにあるファイルを触り、頻出集合を求める段階で、ファイルアクセス記録が除かれたため、頻出する Web ページ群しか求められなかったと推定できる。このような Web ページ群しか含まない結果集合は、ユーザの単なる Web 閲覧の結果である。本研究の目的はファイル群と Web ページ群の関連性の抽出することであるため、このような、ファイル群あるいは Web ページ群しか含まない結果集合を評価対象としないこととする。

3.5　評 価 方 法

適合率 ($Precision$)、再現率 ($Recall$)、F 値 ($F-measure$) を用いて評価する。それぞれの定義を下に示す。

$$Precision = \frac{|Result \cap Examinee|}{|Result|} \quad (1)$$

$$Recall = \frac{|Result \cap Examinee|}{|Examinee|} \quad (2)$$

$$F-measure = \frac{2 * Precision * Recall}{Precision + Recall} \quad (3)$$

正解セットの各要素が複数の集合に分散して得られてしまった場合は、各集合を正解セットと比較し、Recall 値が最大となった集合のみを評価対象とした。また、結果にファイルと Web ページの両方を含む集合だけを評価の対象とした。

図 4 のような複数の結果集合が求められた場合、ファイルと Web ページの両方とも含む集合だけを評価対象にする。図 4 では、ディレクトリ 1 は評価対象であるが、ディレクトリ 2 は評価対象ではない。

3.6　実 験 結 果

3.6.1　実　験　1

実験 1 の目的は両手法の最適パラメータを確定することである。最適パラメータの定義として、結果の F-measure 値が一番良いパラメータを最適パラメータとする。対象パラメータ以外のパラメータを確定した上で、対象パラメータだけを変えながら測定し、F-measure 値が一番高いパラメータを選択する。各手法の最適パラメータの結果は表 3、表 4、表 5 の通りである。各表の中のパラメータ列の表示内容の順番は、Pre-Merge 法では、TransactionTime-MinSupport- Dice 係数の threshold 値の順であり、Post-Merge 法では、TransactionTime-MinSupport-Dice 係数の threshold 値-overlap の順である。

結果から、両手法のそれぞれの最適パラメータが違い、それに、違うユーザでは、普段の利用習慣が違うため、最適パラメータも違うことが分かった。

TransactionTime が短すぎると、関係するファイルが結合しなくなる可能性が高い。逆に、TransactionTime が長すぎると、関係の無いファイルが結果に入ってしまう可能性が高くなる。結論として、ユーザ 1 の Pre-Merge 法では、TransactionTime が 1800[s] 前後が一番良く、Post-Merge 法では、TransactionTime が 3600[s] 前後が最も良いことが分かった。ユーザ 2 の Pre-Merge 法では、TransactionTime が 3600[s] 前後が最も良く、Post-Merge 法では、TransactionTime が 5400[s] 前後が一番良いことが分かった。

Dice 係数の threshold の最適値が、ユーザによってかなり変わってくることが分かった。ユーザ 2 の Post-Merge 法が 0.7 以外では、0.1 から 0.4 までの区間が一番良いことが分かった。あまり高い threshold 値を設定すると、集合が結合しにくくなり、Recall が低くなってしまうためと考えられる。

Post-Merge 法ならではのパラメータ overlap に関して、3 あるいは 4 が一番良いことが分かった。6 まで増やすと、ファイル頻出集合と Web ページ頻出集合が結合しなくなり、両方を含む結果集合が得られない。

3.6.2　実　験　2

実験 2 では、実験 1 で確定した最適パラメータにおいて、両手法を精度の面から比較する。実験の結果は図 5 の通りである。

両手法を Precision, Recall と F-measure の観点から比較すると、Pre-Merge 法は Post-Merge 法よりかなり優れているこ

表 3　データセット 1 の最適パラメータ

手法	パラメータ	正解セット	Precision	Recall	F-measure	実行時間 [s]
Pre-Merge	1800-2-0.1	S_1	0.61	0.76	0.68	
		S_2	1.00	0.86	0.93	
		S_3	1.00	0.67	0.80	
		平均	0.87	0.76	0.80	35.31
Post-Merge	3600-2-0.4-3	S_1	0.56	0.76	0.64	
		S_2	0.21	0.13	0.16	
		S_3	0.00	0.00	0.00	
		平均	0.26	0.30	0.27	18.85

表 4　データセット 2 の最適パラメータ

手法	パラメータ	正解セット	Precision	Recall	F-measure	実行時間 [s]
Pre-Merge	1800-2-0.3	S_1	0.00	0.00	0.00	
		S_2	1.00	0.36	0.53	
		S_3	0.80	0.44	0.57	
		平均	0.60	0.27	0.37	38.79
Post-Merge	3600-2-0.4-3	S_1	0.56	0.76	0.64	
		S_2	0.21	0.13	0.16	
		S_3	0.00	0.00	0.00	
		平均	0.26	0.29	0.27	22.81

表 5　データセット 3 の最適パラメータ

手法	パラメータ	正解セット	Precision	Recall	F-measure	実行時間 [s]
Pre-Merge	3600-2-0.3	S_1	0.80	0.22	0.35	
		平均	0.80	0.22	0.35	67.38
Post-Merge	5400-2-0.7-4	S_1	0.10	0.17	0.13	
		平均	0.10	0.17	0.13	24.38

とが分かった。

原因は二つ考えられる。1 番目の原因とは、性質上から、Web ページだけの頻出集合が少ないことである。例えば、ある作業に属するファイル「A,B,C」と Web ページ「w」があるとすると、ユーザは毎回 A,B,C ファイルのどれかを修正するときに、w ページを参照する場合、Pre-Merge 法では、ファイル群「A,B,C」と Web ページ w の関連性を見つけることができる、しかし、Post-Merge 法では、先にファイルと Web ページのそれぞれの頻出集合を計算するため、「A,B,C」がファイル頻出集合として求められるが、Web ページ w と頻出するページが少ないあるいはないため、マッチングする前の Web ページ頻出集合に含まれない場合も考えられる。つまり、Post-Merge 法の Web ページだけの頻出集合を求める段階で、頻出する Web ページ集合が少ないため、精度が落ちてしまうと考えられる。

2 番目の原因として、Post-Merge 法の最後のマッチングのところで、関係のないアイテムも結合されてしまうため、適合率が低下してしまったと考えられる。これについてはマッチング処理の改善が今後必要であることが分かった。

なお、今回の実験データの期間が 1-2 カ月ぐらいで、これから、もっと長いデータを利用すると、Pre-Merge 法と Post-Merge 法の精度の差がもっと出ると考えられる。

データセット 1 とデータセット 2 の結果を比べると、ブラウザログを用いた場合の Precision、Recall 及び F-measure の全体的にプロキシログを用いた場合より高いことも分かった。

これに関しても、原因が二つあり、一つ目の原因は、プロキシログにはユーザの意図しない広告サイトのような記録とページに埋め込まれたコンテンツのロード記録が大量に存在するため、結果にゴミが入り、Precision が落ちてしまうと考えられる。二つ目の原因として、ユーザのブラウザと Web サーバ側の設定によって、一回アクセスしたサイトの内容がブラウザの

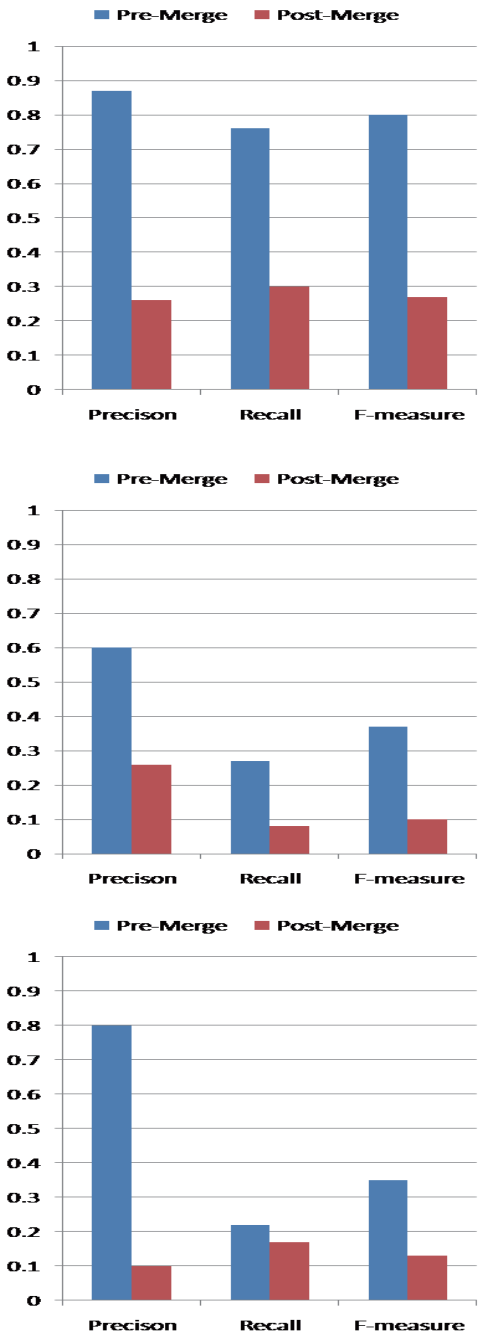

図 5　両手法の精度からの比較 (上) データセット 1, (中) データセット 2, (下) データセット 3

キャッシュに保存され、キャッシュの期限が切れるまでに、再度 Web サーバにアクセスしないため、二回目以降のアクセス記録がブラウザのログにしか記録されず、プロキシログに記録されないこともあるため、Recall が落ちてしまうと考えられる。

3.6.3　実　験　3

実験 3 では、実験 1 で確定した最適パラメータにおいて、両手法を実行時間の面から比較する。目的は、両手法の精度が一番良い時に、どのぐらいの実行時間の差があるかを調べることである。実験の結果は図 6 の通りである。データセット 1 の実験では、Pre-Merge 法は Post-Merge 法の 1.87 倍の実行時間がかかり、データセット 2 では、Pre-Merge 法は Post-Merge 法

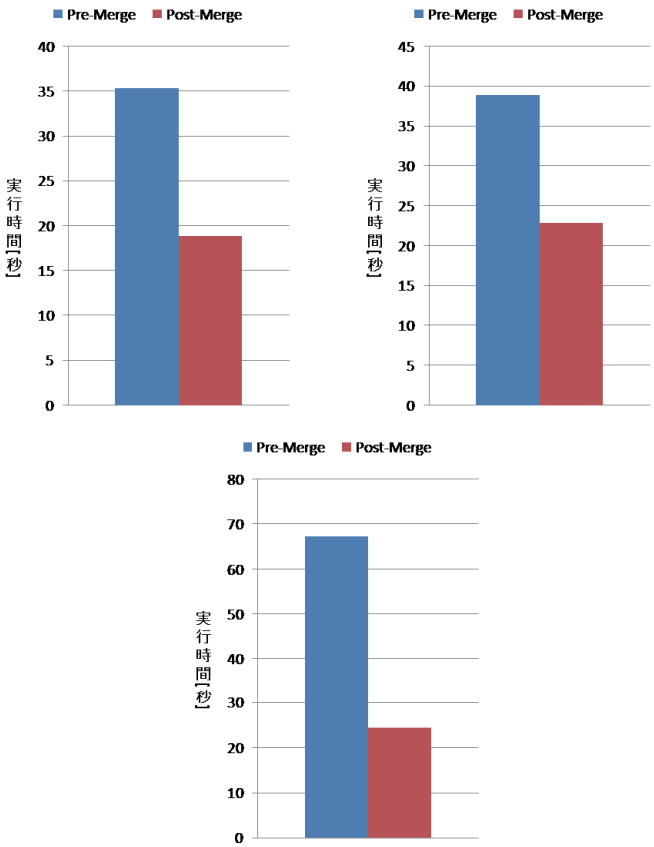

図 6　両手法の実行時間からの比較 (左上) データセット 1, (右上) データセット 2, (下) データセット 3

の 1.70 倍の実行時間がかかり、そして、データセット 3 では、Pre-Merge 法は Post-Merge 法の 2.76 倍の実行時間がかかる事が分かった。

原因が二つ考えられる。1 番目の原因として、各処理ステップにおいて一番時間がかかる頻出集合発見で、Pre-Merge 法では、先に両方のログを結合してから、頻出集合を求めるために、対象のログデータのサイズが大きくなってしまい、Post-Merge 法より多くの時間がかかったと考えられる。2 番目の原因として、Pre-Merge 法のログ結合するところで、二つのログを時間順にソート作業を行う必要があるが、この作業にも無視できない時間がかかっていることが挙げられる。

これから、もっと記録期間の長いデータを利用すると、両手法の間の処理時間の差が更に大きくなると考えられる。そのため、提案手法、特に Pre-Merge 法の処理効率の改善は今後の課題となった。

4. 関 連 研 究

ファイルを検索するシステムといえば、デスクトップ検索システムが普及している。例えば、Windows デスクトップサーチ [1]、Google デスクトップ [2]、Mac OS X の spotlight [3] がある。しかし、デスクトップ検索システムはファイルの中のテキストを用いてキーワード検索を行うため、画像のようなテキストを含まないファイルを検索できない問題がある。ユーザがある作業の全体のファイルを検索するときに、適切なキーワードを選ぶ必要があるが、同じ作業に属するファイルと言っても、必ず、同様なキーワードを持っているとは限らない。特に、作業の規模が大きくなると、キーワード検索を用いて、ある作業に属するファイル群を漏れなく検索することは困難になっていくと考えられる。

渡部らによる FRIDAL [11] [12] [13] は、デスクトップ検索システムの検索キーワードを含まないファイルが検索できない問題を改良する研究である。FRIDAL では、先にキーワード検索を使って、検索語を含むファイル群を一回発見してから、更に、アクセスログから抽出したファイル間関連度を用いてファイル群を探す。これによって、検索キーワードを含まないファイルが検索できない問題が解決される。直接に検索キーワードを含まないファイルに対して、キーワード検索で発見したファイル群との関連性を利用して、検索することが可能となった。例えば、ある文書に含まれるキーワードを入力すると、キーワードを含む文書ファイルと同時に使用されたキーワード非含有ファイルも発見できる。

小田切らによる CO 法 [14] [15]、FI 法 [5]、COFI [16] 法が、FRIDAL と同じく、ファイルを対象にした研究である。ファイルシステム内の複数のディレクトリに分散している同一作業に関連するファイル群を発見し、仮想ディレクトリを生成する手法である。3 手法とも、ファイル群を作業ごとに分類することが目的である。

CO 法 (Clustering using Overlap of file-use) は、使用時間が重複するファイルを発見し階層的クラスタリングを用いて使用時間が長く重複していたファイル群を一つの仮想ディレクトリにまとめる手法である。CO 法はファイルシステム内に分散したファイルを発見することができる、しかし、問題点として、同じ作業に属するファイル群としても、重複しない場合もあるので、このような利用時間が重複しないファイルを発見出来ないことである。

FI 法 (Frequent Itemsets discovery of file-use) は CO 法の改良版として、一定期間内にアクセスしたファイル群を一つのトランザクションに入れて、頻出するファイルの組み合わせを発見することによって、仮想的ディレクトリを生成する手法である。

更に、COFI 法 (Clustering using Overlap of file-use time for Frequent Itemsets) は、CO 法と FI 法を統合して、FI 法でのトランザクションを CO 法のように、利用時間の重複によって、トランザクション同士の関連度を計算し、クラスタリングを行う。

上の研究の共通問題としては検索対象がファイルに限っており、過去に参照した Web ページを検索する事ができないことである。本研究は、検索対象をファイルだけに限らず、過去に同時に参照した Web ページも一緒に検索することを目的とする。

ファイルと過去に見た Web ページの両方とも検索できるシステムとして、大澤らの俺デスク [17] がある。各アプリケーションにプラグインをインストールする事によって、各アプリケーションの動作を記録する。ユーザーの活動履歴をタイムラインで表示でき、ユーザーの「Word で企画書を作成するときに参

照したウェブページをもう一度見たい」という要望に一応対応できる。タイムライン表示で過去のある時点で同時に使用したファイルと見た Web ページを知る事ができるが、しかし、過去のどの時点で一緒にそのファイルと Web ページをアクセスしたかを特定できないような場合に、ユーザは結局過去の履歴を全部調べなければならないことになる。本研究では、頻出アイテム集合を抽出することによって、関連の強いファイル群と Web ページ群を発見して、ユーザに提示することを目的としている。同じ作業に使われたファイル群と一緒に参照した Web ページ群を自動分類する点では、俺デスクとは異なっている。

5. まとめと今後の課題

本研究では、ファイル群と Web ページ群の間の関連性を抽出し、同一作業に属する両者を関連づけて、一つの仮想ディレクトリとしてユーザに提示することを目的とする。このために、アクセスログから、ファイルと Web ページの共起頻度に着目した二つの手法 Pre-Merge 法と Post-Merge 法を提案した。

評価実験では、両手法の最適パラメータを確定した上で、最適パラメータにおける Precision、Recall、F-measure と実行時間を用いて、両手法を比較した。

Pre-Merge 法が Precidion,Recall と F-measure で Post-Merge 法より良い、しかし、処理時間は Post-Merge 法の 1.70〜2.76 倍もかかることが分かった。

今後の課題として、精度の向上、実行時間の削減、ユーザの使用習慣によってのパラメータの自動決定方法と評価対象のユーザ数を増やすことなどが挙げられる。

謝辞 本研究の一部は, 文部科学省科学研究費補助金特定領域研究 (#21013017),

日本学術振興会科学研究費補助金基盤研究 (A)(#22240005) の助成により行われた。

文　献

[1] Windows デスクトップサーチ. http://www.microsoft.com/japan/windows/desktopsearch/default.mspx.
[2] Google デスクトップ. http://desktop.google.co.jp/.
[3] Mac os x spotlight. http://www.apple.com/jp/macosx/what-is-macosx/spotlight.html.
[4] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. *Proc. ACM SIGMOD*, pp. 207–216, 1993.
[5] 小田切健一, 渡辺陽介, 横田治夫. ユーザ作業を反映する仮想ディレクトリ生成のためのアクセス履歴解析手法. 第 148 回 データベースシステム・第 95 回 情報学基礎 合同研究発表会, 2009.
[6] Facclog. http://www2s.biglobe.ne.jp/˜masa-nak/fal_down.htm.
[7] Squid cache. http://www.squid-cache.org/.
[8] Firefox. http://mozilla.jp/firefox/.
[9] Boomtango. http://www.boomtango.com/.
[10] Chrome. http://www.google.com/chrome/.
[11] 渡部徹太郎, 小林隆志, 横田治夫. ファイル検索におけるアクセスログから抽出した関連度の利用 (情報抽出, 夏のデータベースワークショップ 2007(データ工学, 一般)). 電子情報通信学会技術研究報告. DE, データ工学, Vol. 107, No. 131, pp. 503–508, 2007.
[12] 渡部徹太郎, 小林隆志, 横田治夫. キーワード非含有ファイルを検索可能とするファイル間関連度を用いた検索手法の評価. 第 19 回データ工学ワークショップ (DEWS2008), 2008.
[13] Tetsutaro Watanabe, Takashi Kobayashi, and Haruo Yokota. A method for searching keyword-lacking files based on interfile relationships. In *OTM '08: Proceedings of the OTM Confederated International Workshops and Posters on On the Move to Meaningful Internet Systems*, pp. 14–15, Berlin, Heidelberg, 2008. Springer-Verlag.
[14] 小田切健一, 渡辺陽介, 横田治夫. アクセス履歴に基づくファイル間関連度を用いたデスクトップ情報管理ツールの開発 (poster presentation). 信学技報, 2008.
[15] 小田切健一, 渡辺陽介, 横田治夫. アクセス履歴を用いたユーザの作業に対応する仮想ディレクトリの生成. 第 1 回データ工学と情報マネジメントに関するフォーラム (DEIM2009), 2009.
[16] 小田切健一, 渡辺陽介, 横田治夫. 頻出ファイル集合のアクセス時間を考慮した仮想ディレクトリ生成手法. 第 2 回データ工学と情報マネジメントに関するフォーラム（DEIM2010), 2010.
[17] 大澤亮, 高汐一紀, 徳田英幸. 俺デスク:ユーザ操作履歴に基づく情報想起支援ツール. 情報処理学会第 47 回プログラミング・シンポジウム, 2005.