論文 / 著書情報
Article / Book Information

論題	ファイルレコメンデーションのためのファイル利用履歴に基づくタスク間ワークフロー抽出手法
Title	An Extraction Method for Work ow Patterns of Tasks from File Access Logs toward File Recommendation
著者	宋強, 川端貴幸, 伊藤史朗, 渡辺陽介, 横田治夫
Author	Qiang Song, Takayuki Kawabata, Fumiaki Itoh, Yousuke WATANABE, Haruo YOKOTA
掲載誌/書名	第4回データ工学と情報マネジメントに関するフォーラム(DEIM2012), , ,
Journal/Book name	The 4th Forum on Data Engineering and Information Management(DEIM2012), , ,
発行日 / Issue date	2012, 3

ファイルレコメンデーションのための ファイル利用履歴に基づくタスク間ワークフロー抽出手法

宋　強[†]　川端　貴幸[††]　伊藤　史朗[††]　渡辺　陽介[†††]　横田　治夫[†]

† 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
†† キヤノン（株）ソフトウェア応用技術開発センター　〒 146-8501 東京都大田区下丸子 3-30-2
††† 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
E-mail: †{soukyou,watanabe}@de.cs.titech.ac.jp, ††{kawabata.takayuki,ito.fumiaki}@canon.co.jp,
†††yokota@cs.titech.ac.jp

あらまし　近年、情報爆発のためファイルシステムにおけるファイル数が劇的に増えている。このため、必要な文書を探すノウハウを習得するために大きな労力が必要となっている。そのような時間や労力を削減し、必要な情報の収集や文書の作成など価値ある知的創造性の作業に専念させることが重要になる。そこで本研究では、レコメンデーションによるファイル探索支援を実現する。そのためのアプローチとして、ユーザのファイル利用履歴から抽象タスクを求め、更に抽象タスク間の頻出抽象ワークフローのパターンを抽出する手法を提案する。抽出したパターンからユーザの直近のファイル利用に合致した頻出抽象ワークフローを推定し、推定したワークフローに基づきユーザが次に利用するファイルや、ファイルに対する操作のレコメンデーションを行う。

キーワード　[ファイルレコメンデーション、抽象タスク、頻出抽象ワークフロー]

An Extraction Method for Workflow Patterns of Tasks from File Access Logs toward File Recommendation

Qiang SONG[†], Takayuki KAWABATA[††], Fumiaki ITO[††], Yousuke WATANABE[†††], and Haruo YOKOTA[†]

† 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
†† Canon Inc. Applied Software Technology Development Center　3-30-2 Simomaruko, Ota-ku, Tokyo 146-8501, Japan
††† Global Scientific Information and Computing Center, Tokyo Institute of Technology　2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552, Japan
E-mail: †{soukyou,watanabe}@de.cs.titech.ac.jp, ††{kawabata.takayuki,ito.fumiaki}@canon.co.jp,
†††yokota@cs.titech.ac.jp

1. はじめに

近年、情報爆発のためファイルシステムにおける文書数が劇的に増えている。このため、必要な文書を探すために大きな労力が必要となっている [1]。そのような時間や労力を削減し、必要な情報の収集や文書の作成など価値ある知的創造性の作業に専念させることが重要になる。

また、オフィスにおいて探す対象は社内文書や Web 上の文書など情報源としてのファイルだけではない。ある仕事を達成するための手順であったり、効率的に仕事を進める為の方法であったり、何かしらのノウハウとしての情報も探している。

我々は、通常人手で経験的に整理するワークフローを、システムがユーザの操作履歴から価値あるワークフローを自動的にマイニングし、それを基にユーザに操作のお薦めをすることで、ユーザをより知的創造性の作業に専念させられると考えている。

オフィスには業務の目的に合った様々なワークフローが多数存在する。例えば次のようなものがある。

- 見積書作成のワークフロー
- 月報作成のワークフロー
- 購買依頼のワークフローなど

このようなワークフローの典型的なパターンの一つは、**必要な文書の収集 (情報収集) →文書の作成 (情報作成) →承認作業→文書の提出 (情報伝達)** であり、文書に対する閲覧・編集・受け渡し等の操作がその中心的役割を担っている。この点から、文書データを含むファイル全般に対する操作履歴を解析することで、それまで明文化されていなかったワークフローを機械的に抽出できると考えられる。

本研究では、ファイル操作履歴に対する解析からワークフロー抽出を行い、その知識に基づいて、ユーザが現在遂行している業務がどのワークフローに当てはまるのかを推測することで、ユーザが次に必要とするファイルのレコメンデーションを行う。

関連文書のレコメンドのアルゴリズムとして、協調フィルタリング [2] が知られている。協調フィルタリングとは、ユーザの今後の行動を、過去の行動や嗜好が似ている他のユーザの情報をもとに予測する方法である。文書レコメンドシステムでも協調フィルタリングが使える。しかし、協調フィルタリングだけではオフィス内のワークフローを十分に発見することができないため、本研究は協調フィルタリングとは違い、ログからワークフローを抽出し、抽出したワークフローを用いてレコメンデーションを行う新しい手法を提案する。

本論文の構成は以下の通りである。2. 節では、今回提案する手法のアウトラインを述べる。3. 節では、頻出抽象ワークフロー抽出部について解説する。4. 節では、レコメンド部について解説する。5. 節では評価実験について述べる。6. 節では関連論文の紹介を行う。7. 節では、まとめと今後の課題について述べる。

2. 提案手法のアウトライン

本研究の目的はユーザを「探す」という能動的行為から解放し、必要な情報を自動的に検索し提示することによって、ユーザの知的創造性の発揮を支援することである。

システムの全体図を図 1 で示す。ユーザがファイルサーバ上のファイルにアクセスすると、ファイルアクセス履歴がログとして残る。ログにはユーザが、いつ、どのファイルに対して、どのような操作（Open, Close, Create など) をしたかの情報などが記録される。我々はこのログをシステムの入力とし、ワークフローなどの情報を抽出してデータベースに保存する。また、ユーザのファイル操作を監視し、システム内部のデータベースにあるワークフローなどを照合しながら、ユーザの直前の操作によって次にアクセスする可能性が高いファイル及び操作のレコメンデーションを行う。

2.1 ワークフローのモデルの定義

本研究において、ワークフローのモデルをどのように定義するかが重要となる。単純には、ファイル操作をシーケンシャル

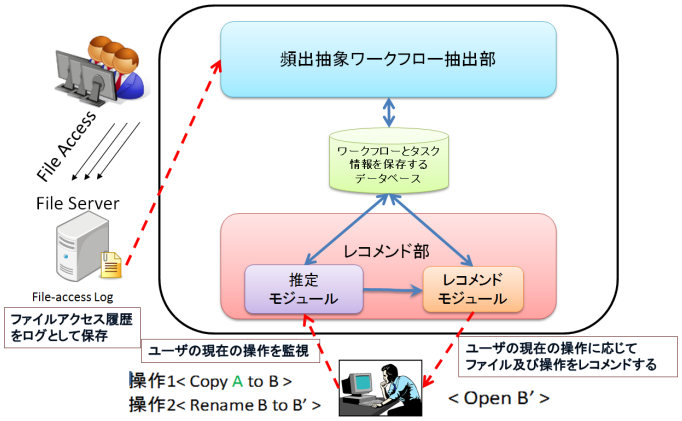

図 1　システム全体図

に並べたものとし、その頻出パターンを抽出すべきワークフローと考えることが出来る。しかし、ワークフローを単純にシーケンシャルな操作パターンとしてみなした場合、同じ順序の完全一致するアクセスパターンが頻繁に出現することは少ないため、見つけることが困難になる。、たとえ、発見したとしても、そのワークフローは汎用的でない可能性が高い。なぜなら、一般的に、オフィスでの仕事のパターンは大まかな流れは似ていても、細かい操作単位で見ると操作の順番が順不同であったり、操作対象のファイルが異なったりすることが多いためである。

そこで我々は、ファイル操作単位での微妙な差異を吸収するために、複数のファイル操作から成る意味のあるまとまりを**抽象タスク**として定義し、抽象タスクをシーケンシャルに並べたものを**抽象ワークフロー**としたモデルを提案する。例えば、

- 記録 1: 12:00 Open FileA
- 記録 2: 12:01 Open FileB
- 記録 3: 12:30 Open FileC

の三つの操作履歴から、

- Task1: Open FileA, Open FileB
- Task2: Open FileC

の二つのタスクを抽出し、更に抽象タスクのシーケンスとして抽象ワークフロー

- Workflow1: Task1–> Task2

を抽出する。抽象タスクはファイル操作の集合としているので、この抽象タスク内でのファイル操作の順番は無視される。こうすることによって、操作の順番或いはファイルが違っても、同じ作業だと解釈できる。

2.2 処 理 手 順

システムの全体図は図 1 で示している通り、内部では頻出抽象ワークフロー抽出部とレコメンド部の 2 つの部分から構成され、更にレコメンド部は推定モジュールとレコメンドモジュールの二つのモジュールを持つ。以下、それぞれについて説明する。

- **頻出抽象ワークフロー抽出部**：ファイルアクセスログを

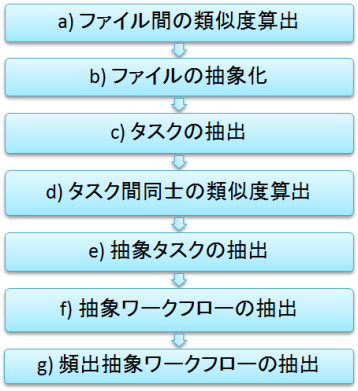

図 2　提案手法の頻出抽象ワークフロー抽出部の処理手順

読み込み、タスク、抽象タスク、抽象ワークフローと頻出抽象ワークフローを発見し、データベースに保存する。タスク、抽象ワークフローについては 3. 節で述べる。

- **レコメンド部 (推定モジュール)**：現在ユーザが行なっている操作情報をフェッチし、データベースから類似度が高い抽象タスクと頻出抽象ワークフローの推定を行う。推定出来た頻出抽象ワークフロー情報を次のレコメンドモジュールに入力する。
- **レコメンド部 (レコメンドモジュール)**：推定できた頻出抽象ワークフローから、ユーザにファイル及び操作のレコメンデーションを行う。

以下の 3. 節と 4. 節でそれぞれの部分について解説する。

3.　頻出抽象ワークフロー抽出部

ここでの目的はログから抽象タスクや頻出抽象ワークフローなどを抽出することである。最初に、タスク、抽象ワークフローについて説明したのち，処理手順を説明する。

まずタスクについて述べる。ユーザがある時間以上ファイル操作をしていないということは、一つのタスクの終了と、次の新しいタスクの開始であると考えられる。そこで、ユーザごとに分けたログをある時間間隔で区切り、得られたファイル操作履歴のサブシーケンスを**タスク**とする。これはユーザの小さい作業単位だと考える。タスクをそのまま出力することも可能であるが、しかし、2.1 節で述べたように、同じ目的の作業であっても、操作の順番が順不同であったり、操作対象ファイルが異なったりする場合もあるため、タスクを抽象化して扱う必要がある。そこで、タスクをクラスタリングし、類似度が高いタスク群を 1 つにまとめたものをここでは抽象タスクとして扱う。つまり、抽象タスクとはファイルのクラスタの集合である。

次に抽象ワークフローについて述べる。最終目的である頻出抽象ワークフローを求める際に，その暫定候補として抽象ワークフローをまず抽出する。**抽象ワークフロー**とは、シーケンス順に並べたタスクの列のことである。隣り合って出現する抽象タスク同士がある時間間隔（ここの時間間隔はタスク生成時の時間間隔とは別なもの）以上空いたら、一つの抽象ワークフローが終了し，新たな抽象ワークフローが開始されたとみなす。

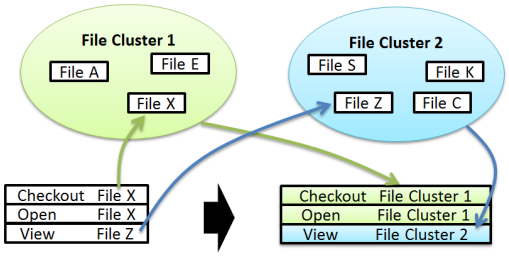

図 3　ファイルの抽象化

つまり、抽象ワークフローは抽象タスクの一段上の概念となり、ユーザの仕事のパターンと言える。しかし、このようなパターンは大量にあるため、不要な情報を除くために、抽象ワークフロー群から頻出する物を抽出し、頻出抽象ワークフローとする。

頻出抽象ワークフロー抽出部の処理手順を図 2 に示す。以下それぞれについて説明する。

3.1　ファイル間の類似度算出

まず、ファイルシステム内にある全ファイル同士の類似度を計算する。ファイル間類似度は，コピー関係類似度とファイル名類似度の重み付線形和によって与えられる。コピー関係による類似度とはコピー元とコピー先の関係を持っている 2 つのファイルの間の類似度のことである。一方、ファイル名とファイル拡張子が似ているほど、ファイル同士の類似度が高いとも考えられる。例えば、「仕様書-version1.docx」と「仕様書-version2.docx」のようなファイル同士に類似度が高いと考えられる。

file A と file B のファイル名による類似度 sim(fileA, fileB) として次にように定義する。

$$sim(fileA, fileB) = \frac{len(LCS(fileA, fileB))}{min(len(fileA), len(fileB))}$$

ここで、len(fileA) は fileA のファイル名の長さを表し、min(len(fileA), len(fileB)) は fileA と fileB でファイル名が短いほうの長さを表す。また LCS(fileA, fileB) は fileA のファイル名と fileB のファイル名の最長共通部分列 (Longest Common Subsequence, LCS) [3] を表す。

3.2　ファイルの抽象化

ファイルを 3.1 節で求めたファイル間類似度によってクラスタリングする。ファイルの抽象化とはファイルをファイルクラスタに置き換えることである。例えば、図 3 ではクラスタリングした結果として、ファイルクラスタ 1 はファイル「X」,「A」,「E」を含み、ファイルクラスタ 2 はファイル「S」,「Z」,「K」,「C」を含むとすると、ファイル抽象化の結果、ユーザがファイルクラスタ 1 をチェックアウト、オープン後にファイルクラスタ 2 を表示したと見なすことができる。

3.3　タスクの抽出

ファイル操作履歴 (ログ) をユーザごとに分け、ユーザごとのログ内で操作と操作の間の空白期間が一定時間（パラメータ:タスクのギャップタイム）以上であれば、1 つのタスクが終了し、新たなタスクが開始されたものとみなす。例えば、図 4 の例で

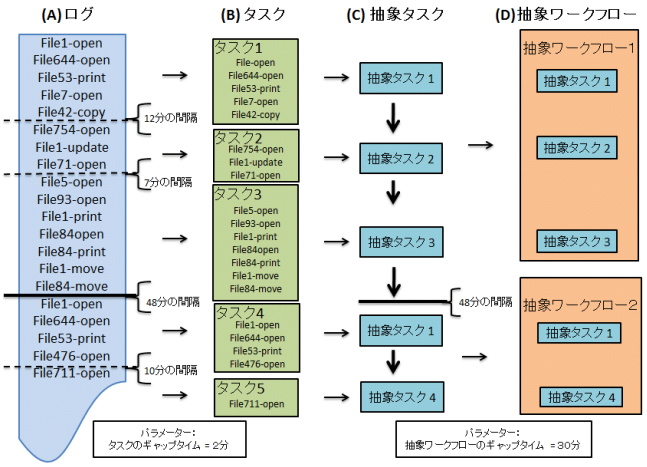

図 4　ユーザごとに分けたログを抽象ワークフローに変換する例

は、タスクのギャップタイムを２分とし、ログをタスクのギャップタイムで区切り、５個のタスクに変換している（図４（A）–>（B））。

3.4　タスク間同士の類似度算出

ファイル間の類似度を利用して、すべてのタスク間同士の類似度を計算する。タスク間の類似度とは、タスクに含まれるファイル操作集合の要素の一致度を表す指標である。ここでは計算方式としてダイス係数を利用する。集合 TaskA と TaskB に対する、Dice 係数を用いた類似度計算式は以下の通りである。

$$sim(TaskA, TaskB) = \frac{2|TaskA \cap TaskB|}{|TaskA| + |TaskB|}$$

２つのタスクの要素に共通なファイルが多いほど、類似度が高いとみなされる。

3.5　抽象タスクの抽出

タスク間の類似度を用いて凝集型階層的クラスタリング [4] を適用し、類似度が高いタスク群をクラスタとしてまとめる。１つのクラスタに含まれるタスク数がN個 (パラメータ:タスクを構成する最小タスク数) 以上なら、それを抽象タスクとする。例えば、図４の例では、クラスタリングした結果によって、一連のタスクを抽象タスクに変換した（図４（B）–>（C））。本研究ではNを３とする。こうする理由としては、要素数が少ないクラスタは汎用的なタスクとは言えないからである。

3.6　抽象ワークフローの抽出

再びファイル操作履歴を読み込み、手順 c) で抽出されたタスクの情報に基づいて、ファイル操作の順序列からタスクの出現の順序列に変換する。さらに手順 e) で生成した抽象タスク (タスクのクラスタ) の情報を用いて、タスクの出現の順序列から、抽象タスクの順序列に変換する。タスクに対応する抽象タスクがない場合はそのタスクは変換対象としない。

得られた一連の抽象タスクの順序列を時間的な閾値を用いてサブシーケンスに分割し、抽象ワークフローを生成する。ここでタスクの抽出と同様に、一定時間以上（パラメータ:抽象ワークフローのギャップタイム）が空いたら、１つの抽象ワークフローが終了したと見なす。例えば、図４の例では、抽象ワークフローのギャップタイムを３０分とし、抽象タスク３と抽象タスク１の間に抽象ワークフローのギャップタイム以上空いているため、抽象ワークフローとして分離する。

仮ワークフローid	タスクシーケンス
1	タスク1, タスク2, タスク3
2	タスク1, タスク4, タスク2, タスク3
3	タスク2, タスク4, タスク1
4	タスク1, タスク1, タスク4

最小サポート値=2
最小シーケンス長=2

ワークフローid	タスクシーケンス	サポート値
1	タスク1, タスク2, タスク3	2
2	タスク1, タスク4	2

図 5　抽象ワークフローから頻出抽象ワークフローを抽出

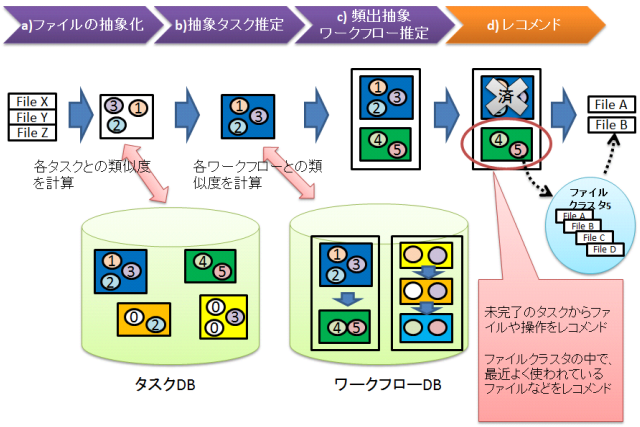

図 6　レコメンデーションを行う例

3.7　頻出抽象ワークフローの抽出

前のステップで作成された抽象ワークフローの集合から、シーケンシャルパターンマイニング [5] を使って、出現回数が一定値以上且つ長さが一定値を超えた頻出タスクシーケンスを頻出抽象ワークフローとして抽出する。例えば、図５では、四つの抽象ワークフローから最小サポート値と最小シーケンス長の両方の条件を満たす二つの頻出抽象ワークフローが得られる（図４（C）–>（D））。

4.　レコメンド部

レコメンド部の処理手順を図７に示す。

4.1　レコメンド部 (推定モジュール)

ここでの目的はユーザの直近のファイル操作情報からユーザが行なっているワークフローの推定である。2.1 節で述べたように、同じ作業のパターンと言っても、扱うファイルが違うと考えられるため、ユーザが現在操作しているファイルを抽象化し、特定のファイルに対する操作としてではなく、同じ特徴で

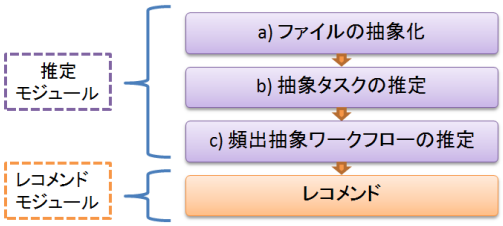

図 7　提案手法のレコメンド部の処理手順

グループ化されたファイルクラスタへの操作と見なす。それから、抽象タスクと頻出抽象ワークフローの推定を行う。

4.1.1　ファイルの抽象化

ファイルを抽象化するとは、ファイルを 3.2 節で行ったファイルクラスタリング結果に応じて、ファイルクラスタに置き換えることである。もし既出のファイルの場合、直接ファイルクラスタに置き換えるが、そうでない場合、次のようにファイル抽象化を行う。例えば、ファイル A が既出でない場合、3.1 節で述べたファイル間類似度によって、最も類似度が高いファイル B を探索し、ファイル B が含まれるファイルクラスタに置き換える。図 6 の例では、入力された三つのファイル「X」,「Y」,「Z」を抽象化して、ファイルクラスタ「1」,「2」,「3」に変換する。

これによって、ユーザが現在操作しているファイルを同じ特徴でグループ化されたファイルクラスタへの操作と見なすことができる。

4.1.2　抽象タスクの推定

ユーザが操作している抽象化したファイル情報によって、今ユーザが行なっている抽象タスクを推定する。3.4 節と同様の類似度を使い、最も類似している抽象タスクを選択する。図 6 の例では、抽象タスク [1, 2, 3] が推定される。

4.1.3　頻出抽象ワークフローの推定

抽象タスクの推定が終わったら、頻出抽象ワークフローの推定を行う。ある抽象タスクを含む頻出抽象ワークフローが複数個存在することが考えられるので、ここで頻出抽象ワークフローを選ぶ基準として次の 3 点に従う。

- 抽象タスクとの一致度
- 頻出抽象ワークフローの出現頻度
- 頻出抽象ワークフローのレコメンド可能要素数

抽象タスクとの一致度が高いほど類似した頻出抽象ワークフローと考えられるためスコアを高くする。頻出抽象ワークフローの出現頻度が高いほど、組織内の多くの人が再利用しているパターンであるためスコアを高くする。頻出抽象ワークフローのレコメンド可能要素数とは、推定した頻出抽象ワークフローにおいて、まだユーザが行っていないファイル操作数を指し、これが多いほどレコメンドする情報多いためスコアを高くする。

以上の三つの基準を加味して、頻出抽象ワークフローのスコア付けを行う。図 6 の例では、推定した抽象タスク [1, 2, 3] を用いて、頻出抽象ワークフローは [1, 2, 3]–> [4, 5] が推定される。

4.2　レコメンド部 (レコメンドモジュール)

この段階の目的は推定した頻出抽象ワークフローに基づき、次にユーザが操作する可能性が高いファイルと操作のレコメンデーションを行うことである。ここでは、ユーザが既に操作したファイルクラスタをレコメンド対象としない。

頻出抽象ワークフローとは抽象タスクの順序列であり、抽象タスクはファイルクラスタに対する操作の集合である。ユーザが現在行っているタスクと、そのタスクの属しているワークフローがわかれば、そこから次の抽象タスクが特定できる。

各ファイルクラスタは複数個のファイルを含むため、レコメンドの際には複数の候補が存在する。例えば、図 6（d）の例のように、ファイルクラスタ 5 に「A」,「B」,「C」,「D」の四つのファイルが存在する。この場合にどのファイルを選んでユーザにレコメンドするべきかという問題がある。我々は以下の 2 つの推薦方法を提案する。

- 最近使ったファイルを優先的にレコメンド (ファイルの最後のアクセス時間順)
- 頻繁に使うファイルを優先的にレコメンド (ファイルのアクセス頻度順)

この二つの推薦方法を組み合わせてレコメンドすることも可能である。例えば、最近一か月に操作されたファイルだけをアクセス頻度順にレコメンドする等の使用方法が考えられる。なお、現在ユーザが操作しているファイルがレコメンド対象のファイルクラスタに含まれる場合に、ユーザが現在操作しているファイルを優先的にレコメンドする。

5.　評 価 実 験

5.1　実験の目的

実験の目的は、本研究の提案手法と比較手法のレコメンド結果を比較・評価することによって、提案手法の有効性を検証することである。

提案手法の比較対象として、本論文で提案する抽象タスクという概念がなく、ファイル操作のシーケンスから直接頻出するファイル操作シーケンスを抽出し、それをワークフローとする方法を用いる。例えば、図 8 で示すように、規定時間以上の間隔が空いたところで、ファイル操作シーケンスを区切り、抽象ワークフローとする。頻出するファイル操作シーケンスを求める方法は、提案手法と同様である。

レコメンド方法は、与えられたファイル操作ともっとも一致度が高いワークフローを求め、そのワークフローの中でまだ操作されていないファイル操作をレコメンドする。

5.2　実験データ

今回の実験は、実際に企業から提供された生のファイルアクセスログを用いて行った。実験データの詳細を表 1 で示す。ログデータには、ユーザ名、アクセス時刻、ファイルパス、操作などの情報が時系列に記録されている。操作としては、「表示」,「更新」,「チェックアウト」,「チェックイン」などがあり、すべての操作の割合を図 9 で示す。

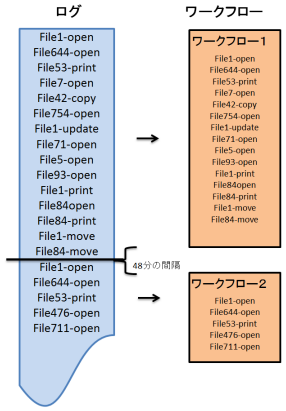

図 8　比較手法：ファイル及びタスクを抽象化せずに抽象ワークフローを抽出する例

表 1　実験データ

ユーザ数	期間	レコード数	ファイル数
22	8ヶ月間	9917	1417

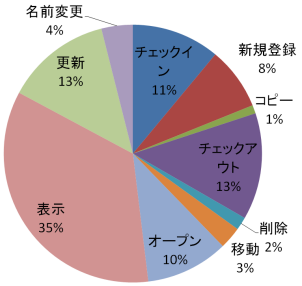

図 9　ログにあるすべての操作の割合

5.3　実験方法

ログを 7 対 3 の割合で分割し、古い 7 割のログを学習用として、タスク、ワークフローの抽出を行い、新しい 3 割のログを評価用とする。評価用のログは、ユーザごとに分け、提案手法と同じ抽象ワークフローのギャップ時間を用いて、ログを分割し、さらに、人手により不適切なログを排除することで、162 個のテストケースを作成した。

評価方法について図 10 を用いて説明する。テストケース 1 つにつき、後述する精度 (Precision)、再現率 (Recall)、F 値 (F-Measure) を求め、全テストケースでのそれぞれの平均を提案手法と比較手法とで比較する。テストケースのファイル操作列のうち、先頭から n 操作をレコメンドシステムへの入力とし、レコメンド結果集合 (Results) を得る。テストケースのファイル操作列の先頭から n 操作を除いた残りのファイル操作列を正解集合 (Examinees) とする。Examinees と Results をマッチングすることで、以下の式で定義される Precision、Recall、F-Measure を求める。

$$Precision = \frac{|Results \cap Examinees|}{|Results|} \quad (1)$$

$$Recall = \frac{|Results \cap Examinees|}{|Examinees|} \quad (2)$$

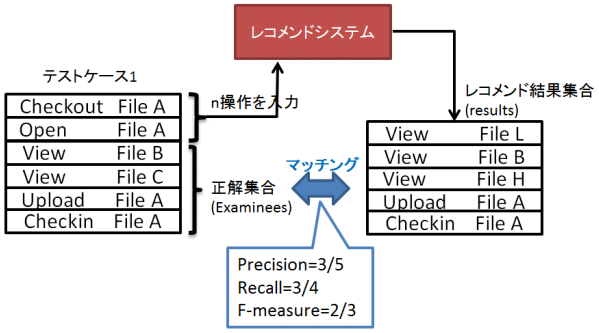

図 10　評価方法の例

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

5.4　実験結果と考察

はじめに、実験に用いた提案手法のパラメータについて述べる。仮タスク間のギャップタイムを 120 秒とし、抽象ワークフロー間のギャップタイムを 30 分とした。タスクを構成する最小タスク数を 3 以上とし、ワークフロー抽出の最少頻出回数と最短シーケンス長はどちらも 2 とした。これらのパラメータはログの特徴を観察し、主観で決めたものである。

上記パラメータを用いて学習用のログから抽出した、タスクやワークフローの情報を表 2 に示す。提案手法と比較手法で抽出されたワークフローを比較すると、提案手法はワークフローの数は減っているが、平均のシーケンス長は長くなっている。提案手法では類似したワークフローがまとまるため、このような違いが表れる。さらに、図 11 を用いて、それぞれのワークフローの違いについて述べる。このバブルチャートは、横軸がワークフローの出現回数（サポート値）を表し、縦軸はワークフローが持つファイル操作の数（シーケンス長）を表し、バブルの大きさはワークフローの数を表している。赤いバブルが提案手法であり、青いバブルが比較手法である。比較手法により抽出されたワークフローは、サポート値、シーケンス長も短いものが大部分である。サポート値が少ない（例えば 2 回の）ワークフローは、偶発的な可能性も高く、再利用性の高いワークフローとは言えない。また、シーケンス長が短いワークフローもレコメンド出来る要素が減り、好ましくない。提案手法では、比較手法に比べ、よりサポートが高く、シーケンス長が長い（右上の）方向へシフトしていることが見て取れる。このことから、より再利用性が高く、レコメンドに適したワークフローが抽出できていると言える。

次に評価用のログから作成したテストケース数について述べる。テストケース数は 162 であり、レコメンドシステムへの入力操作数 n は全テストケース共通で 2 とした。このとき、全テストケースでの正解集合の操作数の平均は 5.2 であった。

シーケンス No	ファイル	操作	レコメンド結果
1	管理/経費実績/2011/JSKA201106278503551.XLS	新規登録	
2	管理/経費実績/2011/JSKA201106278503551.XLS	チェックアウト	入力
3	管理/経費実績/2011/JSKA201106278503551.XLS	更新	O
4	管理/経費実績/2011/JSKA201106278503551.XLS	名前変更	O
5	管理/経費実績/2011/2011_06.XLS	チェックイン	O
6	管理/経費実績/2011/2011_06.XLS	表示	X

表 4　レコメンド例

表 2　抽出したワークフローの情報

	提案手法	比較手法
ログ数	7309	
ファイル数	1358	
ファイルクラスタ数	666	-
ファイルクラスタの平均サイズ	2.039039	-
タスク数	171	-
タスクの平均シーケンス長	2.1	-
ワークフロー数	346	1365
ワークフローの平均シーケンス長	3.54	2.86

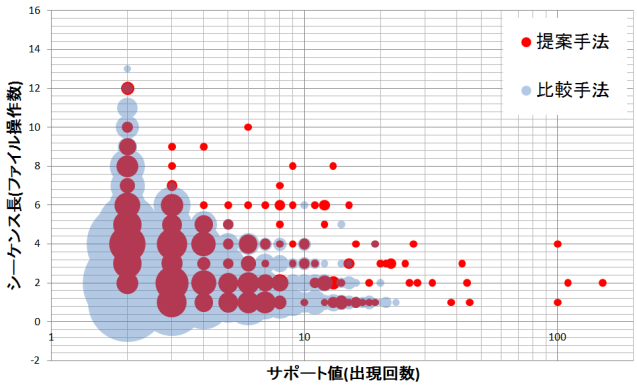

図 11　ワークフローの特徴の比較

表 3　提案手法と比較手法の比較

		比較手法	提案手法
全テストケース数		162	
全テストケース	平均精度	0.05	0.33
	平均再現率	0.03	0.25
	平均 F 値	0.04	0.26
レコメンド出来たテストケース数（割合）		32(0.2)	79(0.49)
レコメンド出来たテストケース	平均精度	0.24	0.67
	平均再現率	0.16	0.52
	平均 F 値	0.18	0.53

評価結果を表 3 に示す。比較手法に比べ、提案手法は全テストケースにおける平均精度、平均再現率、平均 F 値ともに優れ、それぞれおよそ 7 倍以上良くなっている。ここまで大きな差が開いたのは、評価用のログに含まれるファイル操作作業と、学習用のログに含まれるファイル操作作業の間で、同一のものが極端に少ないためである。比較手法では、過去に行ったことがある同一のファイル操作作業しか、レコメンド出来ないため、20%のテストケースでしかレコメンド結果を返すことができなかった。逆に、提案手法では、ファイル操作を抽象化し、過去に行ったことがある類似するファイル操作作業から、レコメンド可能なため、約 50%のテストケースでレコメンド結果を返すことができた。それが、このような評価結果の違いとなって表れたと考えられる。さらに、レコメンド出来たテストケースの中での平均精度、平均再現率、平均 F 値を見ても、提案手法は比較手法より 3 倍以上優れている。このことからも、先にも述べたように、提案手法ではより再利用性が高く、レコメンドに適したワークフローの抽出が可能であり、それを用いることで比較手法に比べ格段にレコメンド性能を上げることができた。最後に提案手法における実際のテストケースの一つのレコメンド結果例を表 4 に示す。ファイルの新規登録、チェックアウトの操作を入力した際に、更新や名前変更、チェックインなどの操作がレコメンド出来ている。

6. 関 連 研 究

アクセスログ（Web アクセスログ或いはファイルアクセスログ）から情報を抽出して、レコメンデーションを行う研究が多数あり、ここでは関連する研究を述べる。

Web のアクセスログを用いて Web ページの推薦を行う研究として、山本らが Web アクセスログ中のシーケンスの LCS を用いることにより、アクセスパターンのぶれを吸収した概括的なアクセス順序を利用して、Web ページの推薦精度を向上させる手法 [6] を提案している。しかし、レコメンドする対象は Web ページであり、本研究が対象とするファイルとは異なる。仮にこの研究の手法をそのままファイルレコメンドシステムに適用すると、ファイルの抽象化及びタスクの概念がないため、有効なレコメンド結果を得るのが難しいと考えられる。

岡本らの研究も Web のアクセスログを用いて Web ページの推薦を目的とする。Web ページそのものではなく、各 Web ページの持つ複数の属性に着目し、それらの組み合わせのパターンを抽出することで Web ページの推薦を行う手法を提案している [7]。複数の属性を対象とすることで、より多くのユーザに共通する傾向を適切に把握し、それを推薦に利用することができ、新規の Web ページでも対応できる。しかし、レコメンドする対象は Web ページである点は本研究と異なる。

田中らは、蓄積された資料の中から必要な資料を検索するユーザに対し、目的に合った資料を予測して推薦する推薦型技術資料検索システム [8] を提案している。田中らのシステムでは、他のユーザの閲覧履歴を、技術資料閲覧時の特徴を踏まえ

て分析することにより、閲覧者の目的や属性によって異なる閲覧傾向をとらえる。そのうえで、協調フィルタリングを応用した予測と推薦を行う。田中らのシステムでは、同時または連続して閲覧する資料は、ともに同じ作業と関連し、その資料同士にも何らかの関連が高いとし、このように連続して資料を閲覧している間は、同一目的による閲覧と考えている。そこで、閲覧目的の区切りを、ある資料を表示してから次の資料を表示するまでの時間で判断している。実験データから、30 分以上経過していれば新しい目的による閲覧に移行したと見なすことができるとしており、我々のタスク抽出を検討する上で参考としている。

ファイルをレコメンドするシステムとして、Chin-Hui Lai らが Knowledge flow(KF) モデルに基づいたハイブリッドなレコメンド手法 [9] を提案している。この研究では、ファイルを抽象化した直後に、一連のファイル操作群から KF を求める。レコメンド段階で目的ユーザと似ているユーザを探してから、相関ルールマイニングとシーケンシャルパターンマイニングによって頻出するシーケンシャルパターンを求めて、レコメンデーションを行う。しかし、この研究ではファイルをグループへ抽象化した後に KF（我々の抽象ワークフローに相当する）を求めているのに対して、我々の提案手法では抽象化されたファイルをまず抽象タスクとして変換してから、抽象タスクのシーケンスを抽象頻出ワークフローとしている点が異なる。こうすることによって、より精度が高いレコメンデーションが考えられる。

ファイルのレコメンデーションではなく、ファイル検索を目的とした研究として、小田切らによる FI 法 [10] がある。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。FI 法のアプローチとしては、よく一緒にアクセスされたファイル同士の関連度が高いことである。本研究のレコメンド段階での「よく一緒にアクセスされたファイル同士に関連度が高く、このようなファイルを優先的にレコメンドする」点では同じ考え方である。

Wu らもファイル検索を目的にして、ユーザのファイルアクセスログを使って関連ファイルを発見する手法を提案した [11]。この研究では、同一作業に関連するファイルは頻繁に近い時間に使用される傾向があることから、このようなファイル集合を「タスク」として抽出する。しかし、我々の提案手法ではタスクを抽象化する点と抽象化されたタスクのシーケンスを抽象ワークフローと定義する点が違う。また、Wu らの研究ではタスク間の関連度を計算する際に、ファイル間の改名、移動、コピー操作を考慮した。我々の研究でのファイル間類似度計算にコピー操作による類似度を加味する点では同じ考え方である。

7. まとめと今後の課題

本研究では、ユーザが必要な文書や組織内のノウハウを探すために掛ける大量な時間や労力を削減することを目的とする。ファイル操作履歴に対する解析からワークフロー抽出を行い、その知識に基づいてユーザが現在遂行している業務がどのワークフローに当てはまるのかを推測することで、ユーザが次に必要とするファイルのレコメンデーションを行う新しい手法を提案した。更に、実ログデータを使って評価実験を行い、提案手法の有効性を確認した。

既存技術では、過去に行ったことのある同一のファイル操作作業しかレコメンドできない問題点があった。本研究はタスク及び、ワークフローを抽象化することによって、過去に行った類似作業を推定しレコメンドすることが可能となる。また、既存技術では、本質的に同じファイルを別のものとして扱うため、汎用的なパターンの抽出が困難になる。本研究の提案手法では、本質的に同じファイルを抽象化して扱うために再利用性が高い汎用的なパターンのが抽出が可能となる。さらに、既存技術ではシーケンスの重要な部分と不要な部分を切り分けられないために、直列な作業と、並列な作業が混在したパターンを抽出することができない。本研究ではシーケンスを排除した抽象タスクを取り入れることで直列と、並列な作業が混在したパターンを抽出することが可能となる。

今後の課題として、精度を上げるためのパラメータ調整とログを分割する際に一定時間の代わりに操作頻度或いは操作種類などの情報で区切る方法の検討などが挙げられる。

文　献

[1] 吉川日出来. サーチアーキテクチャ. *SoftBank Creative*, pp. 15–17, 2007.

[2] G.Linden, B.Smith, and J.York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, Vol. 7, No. 1, pp. 76–80, 2003.

[3] Daniel S. Hirschberg. Algorithms for the longest common subsequence problem. *Journal of the ACM (JACM)*, Vol. 24, No. Issue 4, pp. 664–675, 1977.

[4] Anil K. Jain and Richard C. Dubes. Algorithms for clustering data. *Prentice-Hall*, 1988.

[5] Jianyong Wang and Jiawei Han. Bide: Efcient mining of frequent closed sequences. *Proceedings. 20th International Conference on Data Engineering*, pp. 79–90, 2004.

[6] 山本理絵, 小林大, 吉原宏, 小林隆志, 横田治夫. アクセスログに基づく web ページ推薦における lcs の利用とその解析. 情報処理学会論文誌：データベース, Vol. 48, No. Sig 11(TOD 34), pp. 38–48, 2007.

[7] 岡本拓明, 横田治夫. 複数属性に着目したアクセス履歴からのページ推薦手法. 情報処理学会論文誌：データベース, Nov. 2009.

[8] 田口浩, 坂上聡子, 岩田雅史. 閲覧者の目的と属性に応じた技術資料の推薦. 第三回データ工学と情報マネジメントに関するフォーラム (DEIM2011), 2011.

[9] Chin-Hui Lai and Duen-Ren Liu. Integrating knowledge flow mining and collaborative filtering to support document recommendation. *The Journal of Systems and Software82*, pp. 2023–2037, 2009.

[10] 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.

[11] Yi Wu, Kenichi Otagiri, Yousuke Watanabe, and Haruo Yokota. A file search method based on intertask relationships derived from access frequency and rmc operations on files. *22nd International Conference on Database and Expert Systems Applications (DEXA 2011)*, pp. 364–378, 2011.