ファイルRMC操作に基づくタスク間関係を用いたファイル検索

呉　　怡† 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
†† 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: †{goi,watanabe}@de.cs.titech.ac.jp, ††yokota@cs.titech.ac.jp

あらまし　近年，ファイルシステム内に格納されているファイルの数が爆発的に増加している．しかし，キーワードによるデスクトップサーチによく用いられる全文検索では，テキストを含まないファイルが検索できない．我々はこれまで，キーワード検索の結果を改善するために，ユーザの操作を記録した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てきた．本研究では，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．また，ファイル間の改名・移動・コピー（RMC）操作を考慮したタスク間関連度の評価式を用いて，全文検索に提案手法を取り入れた新たなファイル検索システムを開発し，被験者実験により，手法の有効性を確認する．

キーワード　デスクトップ検索，全文検索，タスクマイニング

File Search by Considering Relationship between Tasks based on RMC Operations

Yi WU†, Yousuke WATANABE††, and Haruo YOKOTA†

† Department of Computer Science, Graduate School of Information Science and Engineering
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
E-mail: †{goi,watanabe}@de.cs.titech.ac.jp, ††yokota@cs.titech.ac.jp

Abstract　In recent years, there has been an explosive increase in the number of files stored in the file system. However, full-text search which refers to a technique commonly used for keyword-based desktop search is not available for files not containing text. To improve keyword search, we have been proposed a method to find related files by considering the history of file access. Assuming that files related to same work tend to be used at the same time frequently, we extract such files as a "task" and derive formulas for calculating the strength of association between tasks in consideration of the rename, move and copy (RMC) operations. In this study, we develop a new file retrieval system incorporating the formulas into full-text searching, and demonstrate validity of the method.

Key words　desktop search, full text search, task mining

1. ま え が き

情報爆発とも言われているように，ファイルシステム内にあるファイルの数が日々増加しており，ファイルの内容やファイル間の関係を把握することが容易ではない．この問題に対して，数多くのデスクトップ検索ツールが開発されてきた．よく知られているものとして，グーグルの Google デスクトップ [1] や，マイクロソフトの Windows デスクトップ サーチ [2]，Mac OS X に搭載されている Spotlight [3] などがある．これらのデスクトップ検索システムはどれも高速なインデックシングと検索を実現しているが，ファイル内のテキスト情報のほかに，ファイル名や作成時間といった簡単なメタ情報しか考慮していない．そのため，たとえ辞書を利用してもキーワードから図やビデオファイルのようなファイルを検索することが難しく，テキストを含まないファイルに対して有効とは言い難い．

一方で，ファイル間のアクセス共起に基づく相関関係を利用した研究も始まっている．例として，Connections [4]，FRIDAL [5]〜[7] などがある．しかし，偶然の同時アクセスにより，間違った相関関係が抽出されやすい．このような背景から我々は頻繁に近い時間にアクセスされたファイルを同一作

業に関連するものとし，このようなファイル群を「タスク」として抽出してから検索を行うことで，全文検索の課題を解決できると考えた．そして，作業間を結びつける要素として，ファイルの改名（Rename）・移動（Move）・コピー（Copy）（以降，RMC）操作を考慮したタスク間関連度を提案している [8]．なぜなら，RMC 操作があったファイル同士では，それぞれのファイルが使われた作業間でも強い関連性があると考えられる．例えば，論文執筆作業で使用した図をコピーして発表資料作成作業で再利用する場合において，コピー操作を考慮すれば作業間の依存関係がわかる．なお，本稿における作業とは書類作成のような複数のファイルをアクセスしながら行う論理的にひとつの仕事で，ある作業に関連する複数のファイルの集合を「タスク」と表す．

そこで，本研究ではファイルに対する参照・書込みなどの基本操作に加え，RMC 操作に着目したタスク間関連度と通常のキーワード検索と組合せた検索方式を SUGOI (Search by Utilizing Groups Of Interrelated files in a task) を提案する．SUGOI では検索前の準備段階において，ファイル間の共起情報や RMC 操作を考慮して同一作業に使用されたファイル群をタスクとして抽出する．抽出されたタスク間の関連の強さを表したタスク間関連度はファイルの重複度から算出されるタスク間類似度とファイル RMC 操作に基づくタスク間 RMC 関連度によって算出される．その際，時間，編集回数，ファイルサイズの変化による関連度の減衰についても考慮する．

検索する際には，まず全文検索によりキーワードを含むファイルを特定し，そのファイルが属するタスクを発見する．次に，タスク間の関連度に基づいて，全文検索で見つけたタスクとの関連が強い他のタスクを発見する．SUGOI 検索方式により，キーワードを全く含んでいないファイルであっても，関連するタスクに属していれば検索結果として提示することが可能となる．我々の研究グループが日常的に使っているファイルサーバにおけるファイルアクセスログを利用した評価実験により，タスク間関連度を用いたことで，検索結果の適合率，再現率共に向上させることが可能であることを示す．

以下に論文の構成を述べる．2. 節ではデスクトップ検索とファイル整理に関する既存研究を紹介する．3. 節でタスクとタスク間の関連を考慮したファイル検索手法について説明を行う．4. 節において評価実験を記述し，5. 節にてまとめと今後の課題について述べる．

2. 関 連 研 究

近年，個人が扱うファイルの数が増加し続け，パソコン上に散らばっているデータから必要な情報を手軽に見つけ出せるように，ファイル検索・整理を目的とした研究が数多くなされてきた．本節では，本稿と同様にファイルアクセスログを利用した既存研究について紹介する．

2.1 アクセスログを用いたファイル検索

1. 節でも言及した Connections [4] は，Soules らによって提案されたファイル検索ツールで，ファイルシステムコールのログを使用している．Connections はある一定の時間内におけるファイル操作から，ファイル間における参照・被参照関係の有無を推定し，参照されたファイル（input）から参照したファイル（output）へと重み 1 のエッジを張る．ファイル検索を行う際は，エッジの向きと重みでノードであるファイルの重みを伝搬する．そうすることで全文検索の結果を拡張し，リランキングを実現する．Connections では参照・書込み操作のほかに，コピーや改名といった操作も使われているが，ファイル間のエッジの向きを決めるためだけに使用されている．これに対して，本研究では，RMC 操作をタスク間関連度の算出に利用している．

Watanabe らが開発した FRIDAL [5]〜[7] では，キーワードを含まないファイルを検索可能にするために，ファイルの open・close ログを使用している．FRIDAL では，同時に使用したファイルは互いに関連するとして，共起時間や共起回数といった情報に基づき，ファイル間関連度を数値化している．キーワード検索の際には，検索語を含むファイルのスコアをファイル間関連度によって検索語を含まないファイルへと伝搬させることで，キーワード非含有ファイルの検索が可能となる．これに対して，本研究では，アクセスログに基づき，よく近い時間にアクセスされたファイルのグループをタスクとして抽出してから，RMC 操作を考慮してタスク間関連度を算出している．

本研究と同じくタスクマイニングを行っている研究として，Chen らが提案した iMecho [9] というシステムがある．iMecho では，ファイルの内容とファイルアクセスログから以下 3 種類の相関リンクを生成する．

- ファイル内容に基づく相関リンク（例：内容の類似度）
- ファイルアクセスログから直接的に得られる相関リンク（例：ファイルコピー）
- アクセスパターンから間接的に得られる相関リンク（例：同じタスクに属するファイル）

このように，iMecho はコピー操作とタスクマイニングの結果をファイル間の相関リンクの生成に利用しているが，タスク間の関係を考慮していない．また，iMecho がランキングに使用するスコアは，PageRank [10] に基づくリンク解析の結果と全文検索のスコアの積によって定義されているため，キーワードを含まないファイルはキーワード検索の結果として提示されることはない．それに対して，本研究ではキーワードを含まないファイルであっても検索結果に含むことが可能である．

2.2 ファイルクラスタリング

小田切らは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する COFI (Clustering using Overlap of file-use time for Frequent Itemsets) 法 [11] を提案している．COFI 法では，同一作業に関連するファイルは頻繁に近い時間に使われるという性質を利用してファイルを作業単位にクラスタリングして仮想ディレクトリ―として提供している．作業ごとに使用されたファイル群を求めるため，COFI 法はまず，頻出アイテム集合抽出によく用いられる Apriori アルゴリズム [12] を使って，近い時間にアクセスが度々観測されるようなファイルの集合を発見する．次に，集合間のアクセス時間の重複度に基づき，階層的クラスタリングを行うことにより，ファ

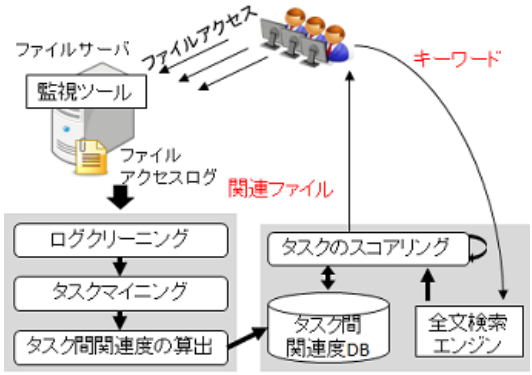

図 1　システムイメージ

イルを作業ごとにまとめた仮想ディレクトリを生成する．これに対して，本研究におけるタスクマイニングはファイルクラスタリングのためではなく，キーワード検索の結果を改善するためである．また，COFI 法ではファイルの open・close ログのみを利用しているが，本研究では RMC 操作まで考慮する．

3.　提案手法 SUGOI

我々は文献 [8] において，RMC 操作を考慮したタスク関連度を用いて，関連ファイルを発見する手法を提案している．本稿で用いるタスク間関連度は，文献 [8] に拡張を加えたものである．提案システムの構成は図 1 で示しているように，主な処理はタスク及びタスク間の関連を抽出する部分と，ユーザが与えた検索キーワードからファイルを検索する部分からなっている．

手法適用に必要なファイルアクセスログはファイルシステムに対するアクセスを監視することによって取得する．ただし，ログにはアクセスの時間，クライアントユーザを特定できる情報，アクセスしたファイル，行われた操作といった項目を含むものとする．本節において下記の処理について説明する．なお，ログクリーニングに関しては実験で使用した監視ツールに特化したものになっているため，4.1.1 節で記述する．

（1）　RMC 操作に着目したタスクマイニング (3.1 節)
（2）　RMC 操作を考慮したタスク間関連度の算出 (3.2 節)
（3）　タスクとタスク間関連度を用いたファイル検索 (3.3 節)

3.1　タスクマイニング

タスクマイニングの目的は作業ごとのファイル群を発見することである．抽出対象のタスクは下記の 2 種類ある．

FI タスク:　頻出アイテム集合 (Frequent Itemset) であるタスクである．その着眼点は同一作業に関連するファイルは頻繁に近い時間にアクセスされる傾向があることである．

RMC タスク:　RMC 操作に基づいて抽出されるタスクで，本研究で新たに導入したものである．複数のファイルに対して，短時間のうちにまとめて RMC 操作が行われた場合，それらのファイル群は論理的に意味のあるまとまりであり，同一作業に由来するものである可能性が高い．そこで，このようなまとまった RMC 操作の行われるファイル群については，アクセスが頻出でなくてもタスクとして取り扱い，RMC タスクとして抽出する．

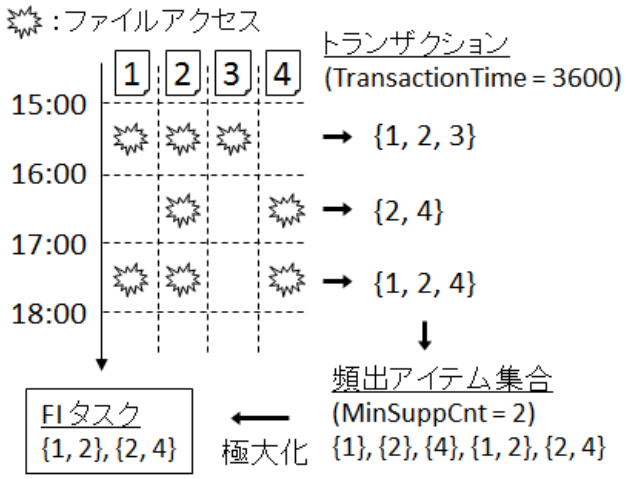

図 2　FI タスクの抽出

タスクマイニングによって，ユーザがアクセスした多くのファイルは少なくともひとつのタスクに属することになる．

3.1.1　FI タスク

論文執筆作業の際，tex ファイルのほかに pdf ファイルやグラフなどの図のファイルに対して，近い時間にアクセスを繰り返すことがあるように，同一作業に関連するファイルは，頻繁に近い時間にアクセスされることが多い．タスクを抽出するため，COFI 法 [11] と同様に頻出アイテム集合であるファイル群を発見する．

図 2 にて FI タスクの抽出手順を示す．我々はまず，ファイルアクセスログを一定の時間幅 ($TransactionTime$) でトランザクション単位に分割する．次に，Eclat アルゴリズム [13] を適用し，トランザクションにおける出現回数が $MinSuppCnt$ 回以上の頻出ファイル集合をタスク (T) として抽出する．提案手法では，長い期間にわたるファイルアクセスログを解析対象としているため，個別ファイルのアクセス回数に比べ，トランザクションの件数が遥かに大きく，$MinSuppCnt$ を低く設定する必要がある．そこで，低い $MinSuppCnt$ を設定してもパフォーマンスへの影響が小さい頻出アイテム集合マイニングアルゴリズム Eclat [13] を採用する．最後に，他の集合の部分集合でない集合のみを FI タスクとする．

3.1.2　RMC タスク

過去の作業で使われた複数のファイルをコピーなどをして，他の作業で再利用することがよくある．それ故，ある一定の時間幅の中で RMC されたファイル群を過去に行われたある作業に関連するファイルのグループと考えられ，本稿では，このようなファイルのグループをタスクとして抽出する (図 3)．

3.2　タスク間関連度の算出

本研究においてはタスク間の関係を，各タスクをノードとしたグラフ構造で表現する．タスク同士をつなぐエッジには，タスク間の関連の強さを表す重みの数値が付与される．タスク間の関連を数値化する際に，タスクにあるファイルの重複を考慮したタスク間類似度と，ファイル間 RMC 操作に着目したタスク間 RMC 関連度を用いる．特にタスク間 RMC 関連度は，異

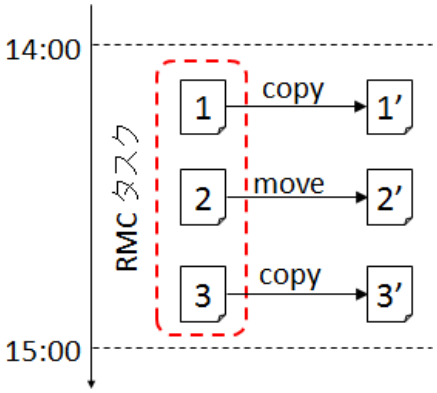

図 3　RMC タスクの抽出

なるタスクにまたがるファイル間の RMC 操作の種類や回数のみならず，RMC 操作が発生してからの経過時間，編集回数，ファイルサイズの変化による関連度の減衰についても考慮している．

両タスク間関連度 $R(T_m \to T_n)$ は，各作業で使われたファイルの重複度合いを表したタスク間類似度 $sim_t(T_m \to T_n)$ と，タスクをまたがったファイル間 RMC 操作の種類や回数を考慮したタスク間 RMC 関連度 $rmc_t(T_m \to T_n)$ によって算出される (式 (1))．

$$\begin{aligned} &R(T_m \to T_n) \\ &= \theta * sim_t(T_m \to T_n) + (1-\theta) * rmc_t(T_m \to T_n) \end{aligned} \tag{1}$$

ただし，θ はタスク間類似度とタスク間 RMC 関連度を考慮する度合いを調節するためのパラメータで，$(0 \leqq \theta \leqq 1)$ を満たす．

3.2.1　タスク間類似度

多くの共通ファイルを使った作業間の関係が強いと推測できるため，各タスクに含まれるファイルの重複の度合いに基づいてタスク間類似度を算出する．

タスク m，n 間のタスク間類似度 $sim_t(T_m \to T_n)$ は式 (2) によって定義される．

$$sim_t(T_m \to T_n) = \frac{|T_m \cap T_n|}{|T_m|} \tag{2}$$

ただし，T_m と T_n はタスク A とタスク B に含まれるファイルの集合を表す．

3.2.2　タスク間 RMC 関連度

過去の作業で使用されたファイルをコピーなどをして，他の作業で再利用することがよくある．そのため，RMC 操作のあったタスク間では他より関連が強いと思われる．そこで，タスク間 RMC 関連度を算出するためのファイル間 RMC 関連度を式 (3) で定義する．

$$rmc_f(f_i \to f_j) = \begin{cases} \alpha_1 & \text{if } f_i \text{ was renamed to } f_j, \\ \alpha_2 & \text{if } f_i \text{ was renamed from } f_j, \\ \beta_1 & \text{if } f_i \text{ was moved to } f_j, \\ \beta_2 & \text{if } f_i \text{ was moved from } f_j, \\ \gamma_1 & \text{if } f_i \text{ was copied to } f_j, \\ \gamma_2 & \text{if } f_i \text{ was copied from } f_j, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

式 (3) では，ファイル f_i からファイル f_j へ RMC 操作が発生したことによって得られる f_i，f_j 間のファイル RMC 関連度を定義している．α_1，α_2，β_1，β_2，γ_1，γ_2 は全てパラメータである．

このように定義されたファイル間 RMC 関連度は定数になるが，実際，同じ RMC 操作でも，より最近に RMC 操作のあったファイル間の関連が強い．また編集が重なることにより，RMC 操作のあったファイル内容に差分が大きくなることで，関連が弱くなる可能性があると考えられる．そこで，経過時間，編集回数，ファイルサイズの増減に基づく下記の 3 式を用いて減衰の度合いを算出する．

$$T(f_i, f_j) = \Delta_{time}(f_i, f_j)^{-\tau} \tag{4}$$

$$E(f_i, f_j) = \Delta_{edit}(f_i, f_j)^{-\epsilon} \tag{5}$$

$$S(f_i, f_j) = \Delta_{size}(f_i, f_j)^{-\sigma} \tag{6}$$

ただし，$T(f_i, f_j)$ は経過時間，$E(f_i, f_j)$ は編集回数，$S(f_i, f_j)$ はファイルサイズの増減によってファイル間 RMC 関連度に与える影響を表す．$\Delta_{time}(f_i, f_j)$ はファイル f_i，f_j 間で RMC 操作が発生してから経過した時間で，τ は経過時間による影響の度合いを調節するためのパラメータである．$\Delta_{edit}(f_i, f_j)$ はファイル f_i，f_j 間で RMC 操作が発生してから両ファイルに対する書込み操作の回数の和を表し，ϵ は編集回数による影響の度合いを調節するためのパラメータである．$\Delta_{size}(f_i, f_j)$ はファイル f_i，f_j 間で RMC 操作が発生してから両ファイルのサイズの増加分と減少分の絶対値の和で，σ はファイルサイズの変化による影響の度合いを調節するためのパラメータである．

ファイル間 RMC 関連度に加え，RMC 操作があってからの経過時間，編集回数，そしてファイルサイズの増減による関連度の減衰を考慮し，タスク間 RMC 関連度を下記の式 (7) によって算出する．

$$\begin{aligned} &rmc_t(T_m \to T_n) \\ &= \sum_{(f_i, f_j) \in (T_m, T_n)} \{rmcf(f_i \to f_j) \\ &* T(f_i, f_j) * E(f_i, f_j) * S(f_i, f_j)\} \end{aligned} \tag{7}$$

3.3　キーワード検索の実現

本稿では，キーワードに関連するファイルを直接検索するだけではなく，タスク間関連度を用いて，キーワードに関連するタスクを探すことによって，キーワードを含まない関連ファイルの検索を実現する．以下においてキーワードによるファイル検索の処理手順を記述する．

STEP 1: 既存の全文検索エンジンを用いて，キーワード q を含むファイルを特定する．その際，全文検索エンジンによってキーワードに対するファイルのスコアが算出される．我々は全文検索エンジンが tf.idf 法 [14] に基づいて付けられたファイルスコア $score_f(q, f_j)$ を利用して，そのファイルが所属するタスク T_m のタスクスコアの初期値 $score_t^0(q, T_m)$ を算出する (式 (8))．

$$score_t^0(q, T_m) = \sum_{f_j \in T_m} score_f(q, f_j) \tag{8}$$

STEP 2: タスク間関連度を用いて，キーワードに関連するタスクを検索するための処理を行う．本稿では Connections や FRIDAL で用いられた関連度算出方法をベースに，タスク間関連度に基づき，全タスクに対して，タスクスコア $score_t^k(q, T_m)(1 \leqq k \leqq K)$ を伝搬させる処理を K 回繰り返す．

$score_t^k(q, T_m)$ は式 (9) に従って算出される．

$$score_t^k(q, T_m) = score_t^{k-1}(q, T_m) + \sum_{T_n \in InLink(T_m)} score_t^{k-1}(q, T_n) * R(T_n \to T_m) \tag{9}$$

ただし，$InLink(T_m)$ は T_m に対するタスク関連度 $R(T_n \to T_m) > 0$ となる T_n の集合を表す．

STEP 3: タスクスコアの値を正規化し，スコア $score_t^K(q, T_m) > TH_{score}$ を満たす全ての T_m をキーワードに対する検索結果として出力する．

4. 評価実験

タスクとタスク間関連度を用いたキーワード検索方式の性能を確認するために実験を行う．

4.1 実験環境

4.1.1 ファイルアクセスログ

評価実験を行うため，我々の研究グループが日常的に使用している共有ファイルサーバ（Windows Server 2003 SP2, NTFS）にアクセス監視を目的としたツール FAccLog [15] をインストールし，サーバへのファイルアクセスをモニタリングする．アクセスのモニタリングは OS からの情報収集と LAN アダプタからアクセス情報を基に行うという．ログ収集はほぼリアルタイムで行うことができ，ログにはアクセスした時間，クライアントのユーザ名または使用されたマシンの名前，アクセス元の IP アドレス，アクセスしたファイルのフルパス，およびファイルに対する操作，ファイルサイズなどの情報が含まれている．記録されるファイル操作は参照，書込，新規作成，削除，改名の５種類である．また，改名の場合では，パスは「改名前のパス ≫ 改名後のパス」の形で記録される．

このように集めたファイルアクセスログの多くはユーザ名が記録されておらず，移動とコピーの情報も含まれていない．また，機械アクセスや対象外のファイルに対するアクセスが含まれている．そこでログクリーニングによって手法適用に必要な情報を補完・検出し，不要なアクセスを除去する．

（1） ユーザ名の補完：ユーザ名が欠如しているログに対して，記録された IP アドレスからアクセス元のマシンを割り出し，使用したユーザを特定する．

（2） 移動・コピー操作の検出：移動操作は，改名としてログに記録されているため，ここではディレクトリの変更を伴なう改名操作を移動とする．またコピー操作に関しては，分単位で区切った一連のファイルアクセスログの中，あるファイル f に対する新規作成操作の前，他のディレクトリにある同名のファイル f' に対する参照操作が記録されていたら，f は f' のコピーとする．

（3） 対象外アクセスの除去：収集されたファイルアクセスログの中には，アプリケーションなどによる機械アクセスや直接作業に関係のないアクセスが多く含まれている．機械アクセスの例として，ファイルの編集や検索によって生成されたテンポラリファイルへのアクセスや，プレビューの表示によるアクセスなどがある．また，バックアップやプログラムのコンパイルで発生するファイルアクセスは直接作業に関係のないファイルアクセスの一例である．そこで，２種類のスレッショルドを用いて機械アクセスを除去する．

- TH_{min}：一分間におけるアクセス回数の上限値
- TH_{sec}：一秒間におけるアクセス回数の上限値

その他に拡張子を用いたフィルタリングも行う．本稿では，拡張子が.db，.dll，.ico，.ini，.class などのファイルを対象外とする．そのほかに，フォルダへのアクセスもログに含まれるが，本稿ではキーワードに関連するファイルを検索することが目的のため，フォルダへのアクセスも除去対象とする．

4.1.2 全文検索

キーワード検索では，全文検索エンジンである Hyper Estraier [16] を使用している．Hyper Estraier は N-gram 方式による検索が可能で，インデックスを使って高速に大量の文章を検索できる．実験では設定により，プレーンテキスト，HTML のほかに，拡張子が.pdf，.doc，.xls，.ppt，.docx，.xlsx，.pptx のファイルを検索対象にした．

4.2 実験方法

実験では，7 名の被験者の協力を得てアクセスログに記録されたファイルの中から，キーワードに関連するファイルを選出して正解集合を作成する．評価では正解集合と比較して行っている．なお，本稿執筆の時点でファイルシステムから削除されたファイルは評価対象外とする．実験用データセットの概要を表 1 にまとめた．表中「全文」とは Hyper Estraier によって索引が抽出できたファイルの数で，ファイル数の半分にも満たないケースが多いことが分かる．ただし，被験者 E, F, G に関してはプライバシーの理由で一部のファイルに対してのみインデクシングを行った．

固定したパラメータとして，ログクリーニングでは $TH_{min} = 30$，$TH_{sec} = 5$ とし，関連ファイル検索におけるスコアの閾値を $TH_{score} = 0$ に設定した．また，スコア伝搬の繰り返し回数 $K = 3$ とした．

4.3 タスクマイニングに関する実験

本稿では，キーワード検索では探せないファイルを検索可能にするため，アクセスログに基づき，同一作業に関連するファ

表 1　実験用データセット

被験者	ログ数	ファイル	全文	キーワード	正解数
A	3591	201	137	デスクトップ検索	70
B	2808	416	113	日本支部	25
C	3424	318	276	RAPoSDA	32
D	5911	764	311	XCP	84
E	8203	642	335	語学番組	244
F	5123	3422	1152	MTTDL	227
G	13102	3258	338	video slide keyword	73

表 2　FI タスクの実験結果

TransactionTime [秒]	900	1800	3600	5400	7200
MinSuppCnt=2 [F 値]	0.543	0.513	0.507	0.470	0.506
MinSuppCnt=3 [F 値]	0.452	0.432	0.450	0.411	0.453

イルをタスクとして抽出し，同じタスクにあるファイルを同等に扱っている．そこでまずタスクマイニング用パラメータ $TransactionTime$, $MinSuppCnt$, $RMCTaskTime$ の値を決めるために実験を行う．なお，タスク間関連度算出用パラメータは下記の値に設定した．$\theta = 0.5$, $(\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2) = (1.0, 1.0, 1.0, 1.0, 1.0, 1.0)$, $(\tau, \epsilon, \sigma) = (0.0, 0.0, 0.0)$.

4.3.1　FI タスク抽出用パラメータの選定

FI タスクの抽出に用いる $TransactionTime$, $MinSuppCnt$ の値を決めるため，FI タスクのみを検索対象とし，$TransactionTime = \{900, 1800, 3600, 5400, 7200\}$ [秒], $MinSuppCnt = \{2, 3\}$ の組み合わせで実験を行った．

被験者全員の F 値の平均を表 2 に示すように，$TransactionTime$ が同じの場合，$MinSuppCnt = 2$ のほうが $MinSuppCnt = 3$ より良い性能を示している．それは $MinSuppCnt$ を上げることで，タスクに属さないファイル数が増え，タスク間関連度でも辿り着けないファイルが多くなったことが原因だと考えられる．また，$TransactionTime$ の増大により，F 値に低下傾向が見られたが，$MinSuppCnt$ ほどの影響はなかった．その原因は $TransactionTime$ が増えることにより，使用時間に多少の間隔があっても同じトランザクションに入ることができ，他のファイルと関連付けやすくなるが，$TransactionTime$ 内における複数回のアクセスが 1 度としかカウントされないため，$MinSuppCnt$ を満たさなくなる可能性が大きくなるからである．

被験者別の実験結果の詳細を省略するが，解析の結果被験者 A, B, E では $TransactionTime = 3600$ において高い F 値を示し，被験者 C, D, F, G では $TransactionTime = 900$ での良い性能を示している．このことから被験者の作業パターンの違いが推測でき，以降の実験ではそれぞれ異なる $TransactionTime$ を使用して実験を行う．

4.3.2　RMC タスク抽出用パラメータの選定

RMC タスクの抽出に用いる $RMCTaskTime$ の値を決めるために FI タスクに加え，$RMCTaskTime = \{60, 180, 300, 600, 1800\}$[秒] の時に抽出された RMC タスクを用いて検索を行った．

表 3 で実験結果を示しているように，$RMCTaskTime$ の

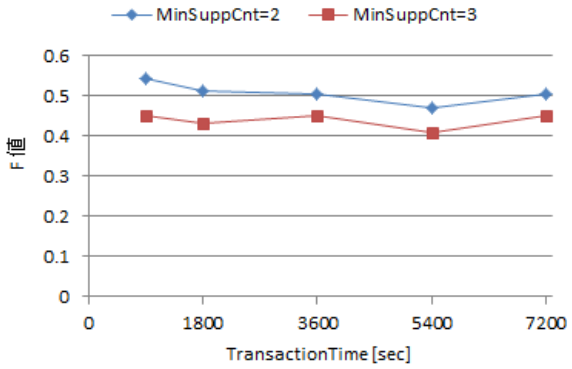

図 4　FI タスクの実験結果

表 3　RMC タスクの実験

RMCTaskTime [秒]	60	180	300	600	1800
適合率	0.776	0.758	0.762	0.762	0.756
再現率	0.669	0.674	0.673	0.675	0.684
F 値	0.684	0.681	0.679	0.684	0.684

表 4　ファイル間 RMC 関連度に関する実験の構成

	改名		移動		コピー	
実験構成	α_1	α_2	β_1	β_2	γ_1	γ_2
RMC なし	0	0	0	0	0	0
改名	1	1	0	0	0	0
移動	0	0	1	1	0	0
コピー	0	0	0	0	1	1
RMC 考慮	1	1	1	1	1	1

増加によって再現率が上昇していくものの，適合率がわずかに低下した．RMC タスクを抽出することで関係のないファイル同士が近い時間に RMC されたことにより，同一作業に関連するものとされてしまうことによる適合率の低下が原因である．しかし，RMC 操作は頻繁に行われていないため，その影響が小さくて F 値の変化が少ない．また，以降の実験では $RMCTaskTime = 60$ に設定する．

4.4　タスク間関連度に関する実験

タスク関連度の算出において多くのパラメータを使用している．各パラメータの特性を調べるために実験を行った．

4.4.1　タスク間 RMC 関連度に関する実験

タスク間関連度の算出に用いるタスク間 RMC 関連度 (式 (7)) は RMC 操作の種類と向きを考慮したファイル間 RMC 関連度を用いている．ここでは，タスク間 RMC 関連度の算出に関わるパラメータの適切な値を調べるために実験を行う．

ファイル間 RMC 関連度は式 (3) によって定義されており，ファイル間 RMC 操作の種類と向きを区別するために α_1，α_2，β_1，β_2，γ_1，γ_2 の 6 つのパラメータを使用している．実験では，$(\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2)$ を表 4 の構成で行った．

実験結果を図 5 にて示している．今回実験で使用したデータセットでは RMC のうち改名と移動の効果があまり見られなかった．改名，移動したファイルはファイルシステムに存在しないため，評価対象でないことや，ファイルに対する改名と移

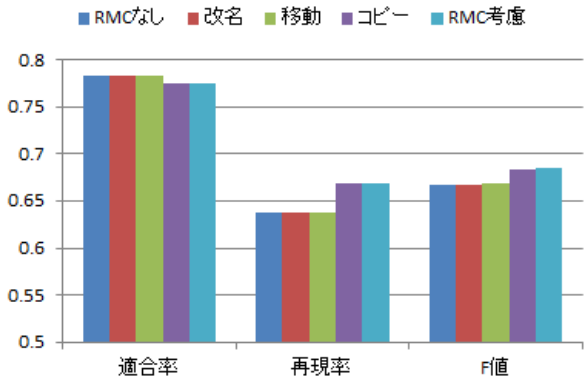

図 5　ファイル間 RMC 関連度に関する実験の結果

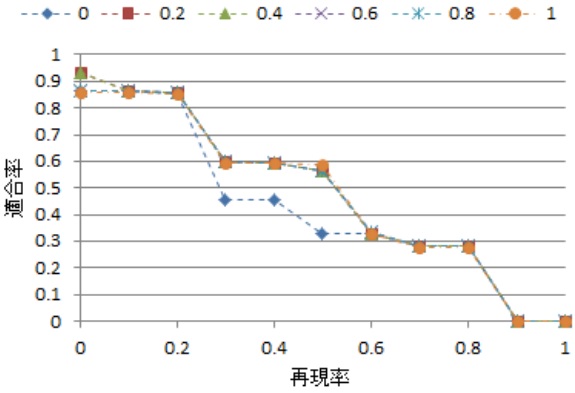

図 6　θ に関する実験結果

表 5　θ による点平均適合率の推移

θ	0.0	0.2	0.4	0.6	0.8	1.0
11 点平均適合率の平均値	0.430	**0.482**	**0.482**	0.476	0.476	0.475

動操作が少なかったこともその一因である．また，コピー操作を考慮したことで再現率と F 値の改善につながった．

4.4.2　θ に関する実験

タスク間関連度を算出する際に，タスク間類似度とタスク間 RMC 関連度を使用し，パラメータ θ で重要視する指標を調節している．θ による検索結果の変化を調べるために，$\theta = \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$ に設定した時の 11 点平均適合率の平均値で評価を行った．ただし，その他のパラメータは $(\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2) = (1.0, 1.0, 1.0, 1.0, 1.0, 1.0)$, $(\tau, \epsilon, \sigma) = (0.0, 0.0, 0.0)$ に設定した．

図 6 から分かるように，$\theta = 0.0$ 以外の場合におけるランキング結果が優れているが，θ による影響が少なかった．それは提案手法では他のタスクと密な関連にあるタスクが上位にランキングされやすくため，タスク間関連度が多少変動してもその性質により多くのスコアが付くためだと思われる．また，$\theta = \{0.2, 0.4\}$ では最も高い 11 点平均適合率の平均値を得られた (表 5)．$\theta = 0.0$ における性能が低かった原因として RMC 操作が少なかったため，タスク間 RMC 関連度だけ使用しては十分にタスク間を関連付けることが困難であることが挙げられる．

表 6　実 験 構 成

実験構成	タスクマイニング	タスク間関連度
SUGOI 構成 1	FI タスクのみ使用	類似度のみ使用
SUGOI 構成 2	FI タスクのみ使用	類似度＋ RMC
SUGOI 構成 3	FI タスク＋ RMC タスク	類似度のみ使用
SUGOI 構成 4	FI タスク＋ RMC タスク	類似度＋ RMC

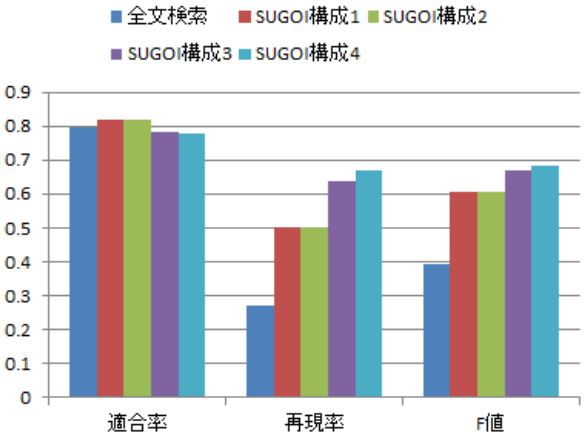

図 7　RMC 操作を考慮した効果の検証実験

表 7　実 験 結 果

実験構成	適合率	再現率	F 値
全文検索のみ	0.795	0.273	0.392
SUGOI 構成 1	0.820	0.500	0.606
SUGOI 構成 2	0.820	0.500	0.606
SUGOI 構成 3	0.784	0.638	0.668
SUGOI 構成 4	0.776	0.669	0.684

4.5　キーワード検索に関する実験

本稿ではタスクマイニングとタスク関連度の算出において RMC 操作を考慮している．タスクマイニングでは，頻出ファイル集合のほかに近い時間に RMC されたファイルをタスクとして抽出する．また，タスク間関連度の算出では，タスク間類似度のほかに RMC 操作を考慮したタスク間 RMC 関連度を用いている．RMC 操作を考慮したことによる結果が改善を確認するため，全文検索のみの場合のほかに，検索に用いるタスクの種類とタスク関連度を組み合わせた表 6 で示す 4 つの構成での検索結果を用いて比較を行った．ただし，タスク間 RMC 関連度を用いる場合のパラメータは $\theta = 0.5$ に設定し，タスク間類似度のみ用いる場合では $\theta = 1.0$ とした．

実験結果を図 7 で示すように，提案手法のすべての構成において全文検索のみ使用した場合に比べ，再現率及び F 値の向上が確認された．その理由として，ファイルアクセスログを考慮することで，キーワードを含まないファイルが，キーワードを含むファイルと同じタスクまたはその関連するタスクに入っていたことが挙げられる．

FI タスクのみ使用した SUGOI 構成 1 と構成 2 では，適合率の改善も見られた．FI タスクは頻繁にアクセスされたファイルの組み合わせからなっており，そのサイズが小さく，平均は 3.0 未満になっている．そのため，無関連のファイルが非常に少

ない．それに対して，RMC タスクでは近い時間に RMC されたファイル群からなっているため，人手によるバックアップ操作やファイル整理などの偶然の共起によって無関係のファイルが混入しやすい．ただし，今回実験に使用したデータセットでは作業に無関係な RMC 操作が少なかったため，全文検索のみの場合に比べ，適合率がわずか 0.02（構成４）低下した (表 7)．

適合率の変化がわずかであったことに対して，再現率の改善幅が 0.3 以上と大きく，F 値の上昇につながり，RMC 操作を考慮してタスクを抽出する効果が見られた（SUGOI 構成３，構成４）．また，タスク間 RMC 関連度を用いた SUGOI 構成２と構成４を比較すると，RMC タスクを利用している場合のみ，タスク間 RMC 関連度の効果が見られたことが分かる．

タスクマイニング及びタスク間関連度の算出で RMC 操作を考慮した SUGOI 構成４では，再現率と F 値が最も高い値となった．その理由は RMC 操作を考慮したことにより，キーワードを含むファイルがなくて類似度だけでは見つからなかったタスクにもスコアが配分されたと考えられ，提案手法の有効性を示した．

4.6 実験のまとめ

4. 節において被験者実験により，提案手法 SUGOI の有効性を確認し，パラメータの特性を調べ，以下の結論が得られた．

- タスクマイニングに関する実験では，$MinSuppCnt = 2$ における F 値が $MinSuppCnt = 3$ の時より高く，トランザクションにおける共起回数が 2 回以上あれば，関連するファイル同士である可能性が高いことが分かった．

また，RMC タスク抽出用の $RMCTaskTime$ を増やすことで適合率が低下し，再現率の増加傾向が見られたが，今回使用したデータセットにおいて RMC 操作が頻繁に行われていなかったためその影響は小さい．

- タスク間 RMC 関連度に関する実験では，改名，移動，コピーの 3 操作のうち，コピー操作が最も有効であることが分かった．元のファイルに変更を加えることなく，他の作業で使用するという作業パターンが多いことが推測できる．

式 (1) において，タスク間類似度とタスク間 RMC 関連度を考慮する度合いを調節するためのパラメータ θ に関する実験では，$\theta = \{0.2, 0.4\}$ におけるランキングの精度が最も良かった結果となった．

- RMC 操作を考慮した効果を検証するための実験では，RMC タスク及びタスク間 RMC 関連度を利用した構成が既存の全文検索エンジンに比べ，F 値が 0.292 上昇し，RMC 操作を考慮した提案手法 SUGOI の有効性を示した．

5. まとめと今後の課題

パソコン内の情報が急速に増加しているが，全文検索だけでは検索できないファイルが存在する．本稿では，ファイルアクセスログに基づき，ファイルを作業ごとにまとめ，ファイルの改名・移動・コピー（RMC）操作を考慮したタスク間関連度をタスク間の関係を量る指標とした．そして，全文検索とタスク間関連度を組合せた検索方式を提案した．被験者実験を行い，提案手法 SUGOI によって適合率と再現率が改善したことを確認した．

今後の課題として，長期間にわたるアクセスログを用いて評価実験を行い，ユーザのアクセスパターンに合ったパラメータの自動設定方法を検討していきたい．また，タスク間関連度の算出式及びそれを利用したスコアリング手法の改善や，タスクマイニングとタスク間関連度の情報を活かした検索結果の適切な提示方法などを考えていきたい．

謝　　辞

本研究の一部は，日本学術振興会科学研究費補助金基盤研究 (A)(#22240005) および文部科学省科学研究費補助金特定領域研究 (#21013017) の助成により行われた．

文　　献

[1] Google. Google デストップ. http://desktop.google.com.
[2] Microsoft Corporation. Windows デスクトップ サーチ. http://www.microsoft.com/japan/windows/desktopsearch/default.mspx.
[3] Apple Inc. Spotlight. http://www.apple.com/jp/macosx/what-is-macosx/spotlight.html.
[4] Craig A. N. Soules and Gregory R. Ganger. Connections: using context to enhance file search. *SIGOPS Oper. Syst. Rev.*, Vol. 39, No. 5, pp. 119–132, 2005.
[5] 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. 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.
[6] 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.
[7] Tetsutaro Watanabe, Takashi Kobayashi, and Haruo Yokota. A method for searching keyword-lacking files based on interfile relationships. In *OTM '08*, pp. 14–15, Berlin, Heidelberg, 2008. Springer-Verlag.
[8] 呉怡, 渡辺陽介, 横田治夫. ファイル RMC 操作を考慮した関連ファイルの発見. 第 150 回 データベースシステム研究発表会, 第 2010-DBS-150 巻, 2010.
[9] Jidong Chen, Hang Guo, Wentao Wu, and Wei Wang. iMecho: an associative memory based desktop search system. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pp. 731–740, New York, NY, USA, 2009. ACM.
[10] Larry Page, Sergey Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University, 1998.
[11] 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.
[12] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. *SIGMOD Rec.*, Vol. 22, No. 2, pp. 207–216, 1993.
[13] Mohammed J Zaki, Srinivasan Parthasarathy, Mitsunori Ogihara, and Wei Li. New algorithms for fast discovery of association rules. In *KDD-97 Proceedings*, pp. 283–286, 1997.
[14] Gerald Salton, editor. *Automatic text processing*. Addison-Wesley Longman Publishing Co., Inc., 1988.
[15] だいこくネット. FAccLog. http://www2s.biglobe.ne.jp/~masa-nak/fal_down.htm.
[16] Mikio Hirabayashi. Hyper estraier. http://fallabs.com/hyperestraier/.