Vol.2010-DBS-150 No.20
Vol.2010-IFAT-99 No.20
2010/8/4

ファイルRMC操作を考慮した関連ファイルの発見

呉　　怡[†1]　渡辺陽介[†2]　横田治夫[†1]

ストレージの大容量化により，ファイルシステム内のファイル数が日々増加しており，膨大なデータに対して手軽に管理できる仕組みが必要になってきた．それに対して，パソコン内におけるデータ検索・ファイル整理を目的とした研究が多く提案されてきたが，全文検索をベースとしたアプローチが多く，検索語を含まないファイルの検索が困難で，再現率の低下が問題となっていた．そこで，本稿ではユーザのファイル操作履歴に加え，RMC 操作を考慮した関連ファイルの発見手法を提案する．提案手法はまず同一作業に使用されたファイルの集合を生成して，次に RMC 操作やそのほかの要素を考慮したタスク間の関連度を算出する．最後にタスク間関連度を用いることで関連ファイルを発見する．

Finding Related Files by Considering the RMC Operations

Yi WU,[†1] Yousuke WATANABE[†2]
and Haruo YOKOTA [†1]

Since the capacity of storage devices is becoming larger and larger, the number of files in a file system is steadily increasing. Some simple mechanism for management vast amounts of data is required. To solve the problem, many systems has been proposed with the aim of discovering knowledge or finding data. Nevertheless, because of basing on full-text search, many of the systems cannot deal with files which do not include the search keywords and the recall of search result is unsatisfactory. So in this paper, we propose a method to find the related files by considering not only the history of file access, but also the rename, move and copy(RMC) operations. We first extract file groups used in the same tasks, and then calculate their scores based on RMC task operations and other factors. At the end, we find the related files by using the scores between tasks.

1. はじめに

ストレージの大容量化により，ファイルシステムにあるファイルの数が爆発的に増加しており，その中のほとんどが非構造化データで，ファイル形式も様々である．そのため，ファイルシステムにおける知識の発見や情報の探索が困難になってきた．その問題に対して，数多くのデスクトップ検索ツールが開発されてきた．Google デスクトップ[1)] ではコンピュータに保存されているメール，ファイル，ウェブの履歴などを対象にインデックスを作成し，パソコン上のデータを検索する機能を提供している．そのほかにも，マイクロソフトの Windows デスクトップ サーチ[2)]（WDS）や，Spotlight[3)] という Mac OS X に搭載されているデスクトップ検索アプリケーション，そして Linux 向けに開発されたデスクトップ検索ツール Beagle[4)] などがある．しかしこのようにファイル内のテキストまたはファイル名や作成時間といった簡単なメタ情報のみ考慮した既存のデスクトップ検索システムでは，検索語とのマッチングが必要なため，検索できないファイルが存在する．例えば，テキストを含まない画像，ビデオファイルがその一例である．

これまで我々の研究グループでは，論文を参照しながら発表資料を作成するような場合において，同時に使われたファイルのほうがより強い関連性を持つと考え，利用者の操作履歴にあるファイルの共起情報を考慮したファイル間関連度を利用して，キーワードを含まないファイルの検索を可能にし，評価実験により，従来の検索システムに比べてより良い結果を示してきた[5)–7)]．また，同一作業に関連するファイルが同時に使用されることが多いことから，頻繁に近い時間に使用されたファイル群を発見することで，ファイルを作業単位に分類する手法を提案している[8)–11)]．しかしファイルのリネーム（Rename）・移動（Move）・コピー（Copy）（以下 RMC）操作を考慮しなかったため，RMC 操作のあったファイルは元のファイルと全く異なるものとして扱われてしまい，例えばあるファイルをコピーしてほかのタスクで再利用した際，コピー操作を考慮しないと，元となるファイルやそれに関連するタスクの内容を調べることができない．そして共起情報だけ用いるとき，あるファイルを使った作業と，そのファイルを RMC 操作によって再利用したほかの作業との関連性が低く

†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
Department of Computer Science, Graduate School of Information Science and Engineering
2–12–1 Ookayama, Meguro-ku, Tokyo, 152–8552 JAPAN
†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
Global Scientific Information and Computing Center, Tokyo Institute of Technology
2–12–1 Ookayama, Meguro-ku, Tokyo, 152–8552 JAPAN

Vol.2010-DBS-150 No.20
Vol.2010-IFAT-99 No.20
2010/8/4

評価されてしまう．

そこで本稿ではファイルのアクセス履歴に加え，RMC 操作を考慮した関連ファイルの発見手法を提案する．なお，問題を単純にするために，本稿ではキーワードによる検索を行わず，あるファイルに関連するファイルの発見のみを目的とする．提案手法では，従来のデスクトップ検索にある再現率の低下問題を改善するために，アクセス履歴をある時間間隔でトランザクションに区切り，Apriori アルゴリズムを用いて頻出ファイル集合マイニングを行う．同じ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から，各タスクで使用されたファイルの集合として頻出ファイル集合を求める．次に，各タスクで使われたファイルの重複度とファイル RMC 操作から推定されるタスク間の関連，それに時間，編集回数，ファイルサイズの変化などの要素による関連性の変化を考慮しながら，タスク間の関連度を算出する．最後に，タスク間関連度を利用して関連ファイルを発見する．

以下に論文の構成を述べる．2 節でデスクトップ検索およびファイル整理を目的とした既存研究について紹介する．3 節では，提案手法の概要について述べたあと，4 節において頻出ファイル集合の生成方法を説明して，5 で関連ファイルの発見手法の詳細について説明を行う．最後に 6 節にてまとめと今後の課題について述べる．

2. 関 連 研 究

2.1 デスクトップ検索

パソコン内のリソースが増加し続け，コンピュータ上のデータを検索したいというユーザのニーズに応えようと，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 Google デスクトップ[1] では，メール，ファイル，音楽，写真，チャット，Gmail，閲覧したウェブページなど 17 種類以上のファイル形式をサポートし，ファイルの内容から作成された索引を使って検索を行う．また，ファイルや他のアイテムのキャッシュコピーをとることによって，削除したファイルも検索できることがその特徴である．また，Windows デスクトップ サーチ[2] や Spotlight[3] では，サポートするファイル形式がそれぞれ異なるが，いずれも全文検索をベースとしたアプローチになっている．

Soules らが提案したファイルシステム検索ツール Connections[12] では，ファイルを検索する際，従来のデスクトップ検索システムと同様にファイル内のテキスト情報を使用しているが，その他にファイル操作のためのシステムコールのログ情報を考慮している．Connections はシステムコールのログからファイル間の参照・被参照関係をマイニングし，ファイルのリレーショングラフを作成する．ファイル間の関連を表すエッジの重みを使って，ノード（ファイル）の重みを伝搬させ，それにより，既存のコンテントサーチの結果を拡張し，リランキングを行う．それに対して，本研究ではファイル間関係ではなく，タスク間の関係に重点を置き，タスクで使われたファイル集合の類似度と RMC 操作を考慮している．

Watanabe らが開発した FRIDAL[5)–7)] では，ファイルの open・close ログのみを考慮し，ファイル間の共起時間や，共起回数，共起間隔などの情報から算出されたファイル間関連度を用いて，キーワード非含有ファイルの検索を実現した．それに対して，本稿では Apriori アルゴリズムを使って，アクセス履歴からよく同時にアクセスされたファイルの組み合わせを同一作業で使用されたファイル集合として発見してから，RMC 操作を考慮してタスク間関連度を算出している点に違いがある．

Chen らが提案した iMecho[13] と XSearcher[14] ともファイル間の関連に注目している．iMecho では，利用者の操作をモニタリングし，ユーザ操作からファイル間に様々な相関リンクを作成する．相関リンクは 3 種類あり，ファイルの内容から得られるもの（例：内容の類似度）と，アクセスログから直接得られるもの（例：ファイルコピー）と，アクセスパターンから間接的に得られるもの（例：同じタスクに属するファイル）からなっている．本研究と同様にアクセスログを基にタスクマイニングを行っているが，その結果を相関リンクの生成のみに利用している．ただし，ランキングスコアの算出においては，PageRank[15] をベースとしたリンク解析の結果と全文検索のスコアの積を用いているため，検索語を含まないファイルはランキング結果に含まない．

2.2 ファイル整理

小田切らの研究では，ファイルの open・close 履歴のみ用いて，アクセス時間が比較的近く，同一作業で使用されたファイルを発見する COFI 法[11] を提案してきた．COFI 法では，Apriori アルゴリズム[16] によって，同一作業に使われたと思われるファイル群を頻出ファイル集合として求め，集合としての使用時間の重複から算出された同一作業指数を使った階層的クラスタリングで仮想ディレクトリを生成している．本研究でも，ファイルアクセス時間から共起が頻繁に観測されるファイル群を求めているが，その結果をファイル整理ではなく，デスクトップ検索に利用する点と，RMC 操作を考慮する点は COFI 法との違いである．

3. 提案手法の概要

本稿では，頻繁に近い時間に使用されたファイルは同一作業に関係するファイルである可能性が高いことから，ファイルアクセス履歴から頻出ファイル集合を求めることで，タスクごとに使用されたファイルの集合を生成する．本稿における作業とは論文作成やレポート作

Vol.2010-DBS-150 No.20
Vol.2010-IFAT-99 No.20
2010/8/4

成などのような複数のファイルをアクセスする論理的な１つの作業である．次に，使用されたファイルの重複度や，RMC 操作によるファイルの再利用の有無，そして関連を変化させる様々な要素を考慮し，タスク間関連度を算出する．それらの情報をデスクトップ検索に活かすために，ここではあるファイルをクエリとしたときに，それに関連するファイルの発見を目的とする．また，前提として利用者が日常的に共有ファイルサーバへアクセスして作業を行う環境を想定し，ファイルサーバを監視することにより，利用者のアクセスログを取得する．ただし，ログにはアクセス日時，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，行われた操作などが記録されているとする．

提案手法は主に２部分からなっている．

(1)　頻出ファイル集合の生成（4 節）

(2)　関連ファイルの発見（5 節）

従来のデスクトップ検索システムの多くは高い適合率を実現しているが，再現率の低下が問題となっていた．そこで，本稿はまず頻繁に近い時間にアクセスされたファイルの集合を抽出する．このようなファイル集合は同一作業で使われたファイルである可能性が高いので，抽出された頻出ファイル集合，即ち同じタスクに使用されたファイルの集合に基づき，タスク間関連度を算出する．そうすることによって，直接共起がなく，異なるタスクに属するファイル同士でもタスク間の関連を考慮することで，より強い相関をもつことが可能となる．なお，タスク間関連度は，タスクで使用されたファイルの類似度，ファイル RMC 操作およびそのほかの要素を考慮して算出される．

次に，4 節で抽出した頻出ファイル集合とタスク間関連度に基づき，あるファイルに関連するファイルを発見する．

4. タスク間関連度

4.1 頻出ファイル群の生成

同一作業に関連するファイルは，頻繁に近い時間帯にアクセスされる傾向があるため，ここでは，2 節で紹介した COFI[11] と同様に，アクセス履歴をある時間幅 TransactionTime でトランザクション単位に区切り（**図 1**），バスケット解析でよく用いられてきた Apriori アルゴリズム[16] を適用して，頻出ファイル集合を生成する．以降では，このように同一作業に用いられたファイルの集合として抽出された頻出ファイル集合をタスク（$Task$）で表す．

Apriori アルゴリズムは頻出アイテム集合や相関ルールの抽出によく用いられる手法のひとつで，トランザクションにおけるアイテムの組み合わせの出現率（支持度）のしきい値を

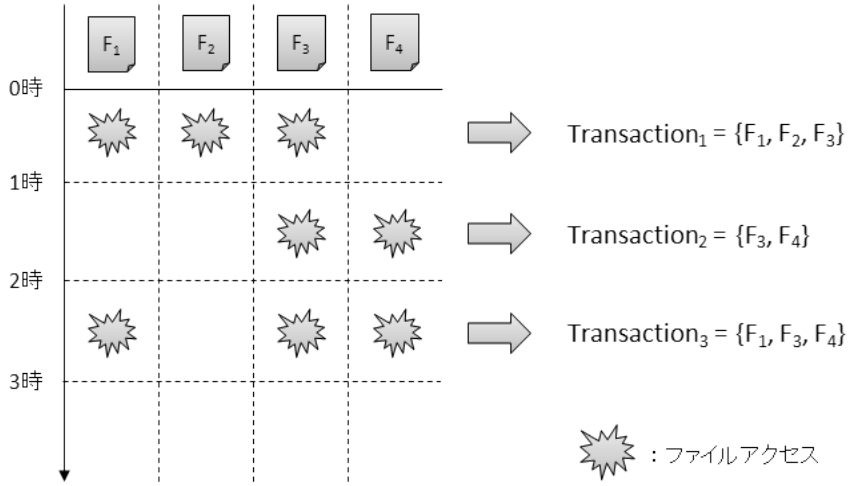

図 1　トランザクションへの分割（$TransactionTime = 1$ 時間）

与えることで頻出アイテム集合を抽出することができる．しかし，ファイルアクセスの場合では，トランザクションの件数に比べ，ファイルのアクセス回数は極めて小さいため，提案手法ではトランザクション中の出現回数のしきい値を与えることで，頻繁に出現するファイルの組み合わせ（頻出ファイル集合）を抽出する．図 1 において，しきい値が２回のときの頻出ファイル集合として，$\{F_1\}$，$\{F_3\}$，$\{F_4\}$，$\{F_1, F_3\}$，$\{F_3, F_4\}$ の５つの組み合わせが抽出される．このように，Apriori アルゴリズムによって抽出される頻出ファイル集合では，同じアイテムが複数の頻出アイテム集合に属することがあるので，ここでは他の頻出ファイル群の部分集合となったものを除去する．例では $\{F_1\}$，$\{F_3\}$，$\{F_4\}$ が除去対象に当たる．

4.2 タスク間関連度の算出

タスク間関連度（$R(Task_A \rightarrow Task_B)$）を算出するため，タスク間の類似度（$sim(Task_A \rightarrow Task_B)$）と RMC 操作の有無（$rcm(Task_A \rightarrow Task_B)$）を考慮する．特に RMC 操作に関しては，RMC 操作が発生してからの期間（$\Delta_{time}(Task_A \rightarrow Task_B)$），両タスクにおける編集回数（$\Delta_{edit}(Task_A \rightarrow Task_B)$），そしてファイルサイズの変化（$\Delta_{size}(Task_A \rightarrow Task_B)$）という３つの要素に基づき，関連度の変動を考慮する．

- タスク類似度（4.2.1 節）
- RMC 操作の考慮（4.2.2 節）

Vol.2010-DBS-150 No.20
Vol.2010-IFAT-99 No.20
2010/8/4

– 時間
– 編集回数
– ファイルサイズの変化

4.2.1 タスク類似度

例えば同じ論文を参照しながら行った作業は研究テーマが近いことが推測されるように，同じファイルを用いるタスクの関連が強いと考えられる．ここでは各タスクで使用されたファイルの重複度をタスク類似度として定義する．タスク A（$Task_A$）とタスク B($Task_B$) との類似度 $sim(Task_A \rightarrow Task_B)$ は式 1 で算出する．

$$sim(Task_A \rightarrow Task_B) = \frac{2 * |Task_A \cap Task_B|}{|Task_A|} \tag{1}$$

4.2.2 RMC 操作の考慮

タスク間関連度にかかわるもう一つの要素として，RMC 操作を用いる．例えば，あるタスクで使われたファイルをコピーして，ほかのタスクで再利用する場合では，2 つのタスクは強い関連性を持つと考えられる．そこで，タスク間関連度を算出するときに，ファイル間のリネーム・コピー・移動（RMC）関係の有無を考慮する．

ファイル i からファイル j へのリネーム・移動・コピー操作があった場合，$rmcf(f_i \rightarrow f_j)$ はファイル i からファイル j 間への rmc 値で，式 2 で定義する．ただし，α_1，α_2，β_1，β_2，γ_1，γ_2 は定数である．

$$rmcf(f_i \rightarrow f_j) = \begin{cases} \alpha_1 & \text{if } f_i \text{ was renamed to } f_j, \\ \alpha_2 & \text{if } f_i \text{ was renamed from } f_j, \\ \beta_1 & \text{if } f_i \text{ was moved to } f_j, \\ \beta_2 & \text{if } f_i \text{ was moved from } f_j, \\ \gamma_1 & \text{if } f_i \text{ was copied to } f_j, \\ \gamma_2 & \text{if } f_i \text{ was copied frrom } f_j, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

$rmcf(f_i \rightarrow f_j)$ を使ってタスク A とタスク B との rmc 値 $rmc(Task_A \rightarrow Task_B)$ を式 3 で定義する．

$$rmc(Task_A \rightarrow Task_B) = \sum_{(f_i, f_j) \in (Task_A, Task_B)} rmcf(f_i \rightarrow f_j) \tag{3}$$

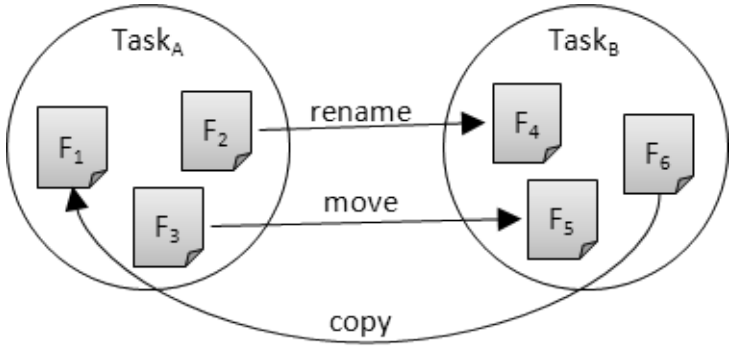

図 2 $rmc(Task_A \rightarrow Task_B)$ の算出例

$(\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2) = (1, 2, 3, 4, 5, 6)$ として，**図 2** を用いて例を示すと，$rmcf$ の値がゼロ以上の組み合わせは $rmcf(f_1 \rightarrow f_6) = 6$，$rmcf(f_6 \rightarrow f_1) = 5$，$rmcf(f_2 \rightarrow f_4) = 1$，$rmcf(f_4 \rightarrow f_2) = 2$，$rmcf(f_3 \rightarrow f_5) = 3$，$rmcf(f_5 \rightarrow f_3) = 4$ となる．これらにより，$rmc(Task_A \rightarrow TaskB) = 6+1+3 = 10$ となり，$rmc(Task_B \rightarrow TaskA) = 5+2+4 = 11$ となる．

4.2.3 タスク間関連度の変動

同じく RMC 操作によって関連付けられたタスク同士でも，より最近に RMC 操作があったほうが関連性が強いと考えられる．また，時間が経つとそれぞれのタスクに異なる編集が重なれば，関連が弱くなることも推測できる．そこで，タスク間関連度の変動を算出するため，rmc 値に時間，編集回数，ファイルサイズの変化に基づき，新たに $rmcf(f_i \rightarrow f_j)$ を定義する．

時間の要素の考慮 ファイル間の rmc 値に時間の要素を加えた $rmcf_t(f_i \rightarrow f_j)$ は式 4 で定義する．$\Delta_{time}(f_i \rightarrow f_j)$ はファイル i からファイル j への RMC 操作が発生してからの時間で，τ は時間の影響度を調節するための定数である．式 4 により，$\tau = 0$ のとき，$rmcf_t(f_i \rightarrow f_j) = rmcf(f_i \rightarrow f_j)$ となり，$rmc_t(Task_A \rightarrow Task_B) = rmc(Task_A \rightarrow Task_B)$ となることがわかる．

$$rmcf_t(f_i \rightarrow f_j) = rmcf(f_i \rightarrow f_j) * \Delta_{time}(f_i \rightarrow f_j)^{-\tau} \tag{4}$$

ただし，以下では $rmcf_t(f_i \rightarrow f_j)$ を式 3 に代入して算出されるタスク間の rmc 値を

Vol.2010-DBS-150 No.20
Vol.2010-IFAT-99 No.20
2010/8/4

$rmc_t(Task_A \rightarrow Task_B)$ で表す．

編集回数の考慮 RMC 操作によって関連付けたファイルでは，重なる編集により，内容が変わって関連度が低くなる可能性がある．そこで，編集回数を考慮した $rmcf_e(f_i \rightarrow f_j)$ を下記のように定義する．

$$rmcf_e(f_i \rightarrow f_j) = rmcf(f_i \rightarrow f_j) * \Delta_{edit}(f_i \rightarrow f_j)^{-\epsilon} \quad (5)$$

ただし，ϵ は定数で，$\Delta_{edit}(f_i \rightarrow f_j)$ はファイル i からファイル j への RMC 操作が発生してから，ファイル i への書き込み回数とファイル j への書き込み回数との和で，以下では $rmcf_e(f_i \rightarrow f_j)$ を式 3 に代入して算出されるタスク間の rmc 値を $rmc_e(Task_A \rightarrow Task_B)$ で表す．

ファイルサイズの考慮 編集回数のほかに，ファイルサイズの変化もタスクの内容の変化を示す指標のひとつで，関連度の変化を推測できる要素である．ここでは，ファイルサイズの変化を考慮した $rmcf_s(f_i \rightarrow f_j)$ を下記のように定義する．

$$rmcf_s(f_i \rightarrow f_j) = rmcf(f_i \rightarrow f_j) * \Delta_{size}(f_i \rightarrow f_j)^{-\sigma} \quad (6)$$

ただし，σ は定数で，$\Delta_{size}(f_i \rightarrow f_j)$ はファイル i からファイル j への RMC 操作が発生してから，ファイル i におけるファイルサイズの変化分とファイル j におけるファイルサイズの変化分との和で，以下では $rmcf_s(f_i \rightarrow f_j)$ を式 3 に代入して算出されるタスク間の rmc 値を $rmc_s(Task_A \rightarrow Task_B)$ で表す．

4.3 タスク間関連度

タスク間類似度と RMC 操作による関連に基づき，あるタスク A（$Task_A$）とタスク B($Task_B$) との関連度 $R(Task_A \rightarrow Task_B)$ を下記のように定義する．ただし，ここで使用する $sim(Task_A \rightarrow Task_B)$，$rmc_t(Task_A \rightarrow Task_B)$，$rmc_e(Task_A \rightarrow Task_B)$，$rmc_s(Task_A \rightarrow Task_B)$ は正規化したもので，θ_1，θ_2，θ_3，θ_4 は定数である．

$$\begin{aligned} R(Task_A \rightarrow Task_B) \quad = \quad & sim(Task_A \rightarrow Task_B)^{\theta_1} + \\ & rmc_t(Task_A \rightarrow Task_B)^{\theta_2} + \\ & rmc_e(Task_A \rightarrow Task_B)^{\theta_3} + \\ & rmc_s(Task_A \rightarrow Task_B)^{\theta_4} \end{aligned} \quad (7)$$

5. 関連ファイルの発見

以下において，関連研究である Connections[12] でも用いられた関連度算出の手法をベースにタスク間関連度を使って，あるファイル f_q に関連するファイルの発見方法について述べる．処理手順は 3 ステップからなっている．

STEP 1: ファイル f_q を含むタスクの集合を $Set_T(f_q)$ として，タスクのスコア $score_t$ を下記のようにセットする．

- $\forall Task_i \in Set_T(f_q), score_t^0(Task_i) = 1$
- $\forall Task_i \notin Set_T(f_q), score_t^0(Task_i) = 0$

STEP 2: 全てのタスクに対して，タスク間関連度を用いて，$score_t$ を更新する．ここではタスクをノードとし，タスク間関連度をノードをつなぐリンクの重みとする．
あるノード $Task_m$ に指すリンクの集合を L_m で表す．ノード $Task_n$ から $Task_m$ に指すリンクを $l_{Task_n \rightarrow Task_m}$ とし，$R(Task_n \rightarrow Task_m)$ はリンクの重みを表している．$Task_m$ の $score_t(Task_m)$ を以下のように更新していく．ただし，$k > 0$ である．

$$score_t^k(Task_m) = score_t^{k-1}(Task_m) + \sum_{l_{i \rightarrow m} \in L_m} score_t^{k-1}(Task_i) * R(Task_i \rightarrow Task_m) \quad (8)$$

STEP 3: スコア $score_t$ がスレッショルド T_{ST} 以上のタスクにある全てのファイルを関連ファイルとする．

図 3 を用いて，タスクスコアの算出例を解説する．例では，4 つのタスク A，B，C，D 間の関連度を重み付きリンクで表している．そのうち，クエリである F_q はタスク A のファイル集合のみに含まれている．図 3 により，$k = 0$ のとき，$score_t^0(Task_A) = 1$，$score_t^0(Task_B) = score_t^0(Task_C) = score_t^0(Task_D) = 0$ となる．

$k = 1$ のとき，

$$\begin{aligned} score_t^1(Task_A) &= 1 \\ score_t^1(Task_B) &= 0 + 1 * 0.2 = 0.2 \\ score_t^1(Task_C) &= 0 + 1 * 0.8 = 0.8 \\ score_t^1(Task_D) &= 0 \end{aligned}$$

となる．

Vol.2010-DBS-150 No.20
Vol.2010-IFAT-99 No.20
2010/8/4

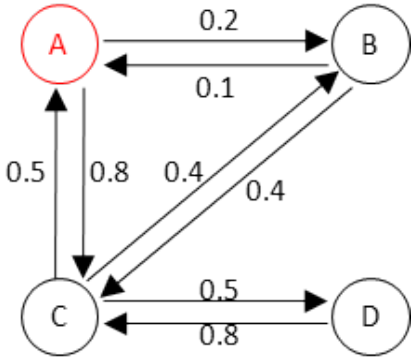

図 3 $score_t$ の更新（$F_q \in Task_A\ and\ F_q \notin Task_B, Task_C, Task_D$）

$k = 2$ のとき，

$$\begin{aligned} score_t^2(Task_A) &= 1 + 0.2 * 0.1 + 0.8 * 0.5 = 1.42 \\ score_t^2(Task_B) &= 0.2 + 1 * 0.2 + 0.8 * 0.4 = 0.72 \\ score_t^2(Task_C) &= 0.8 + 1 * 0.8 + 0.2 * 0.4 = 1.68 \\ score_t^2(Task_D) &= 0 + 0.8 * 0.5 = 0.4 \end{aligned}$$

となる．この時，$T_{ST} = 1.0$ にすると，$Task_A$ と $Task_C$ に含むファイルが F_q に関連するファイルとして発見できる．

6. まとめと今後の課題

これまでのデスクトップ検索ツールの多くは全文検索に基づくもので，そのため検索キーワードを含まないファイルの検索が困難で，再現率低下の問題があった．また，ファイルの共起関係を利用した研究でも，頻繁かつ近い時間にアクセスされないファイル同士の関連度が低くなる傾向がある．そのため，あるタスクで使用したファイルをコピーしてほかのタスクで再利用するような場合では，前後のタスクに強い関連性があっても共起が少ないことになってしまう．このように，リネーム・移動・コピー（RMC）操作では，他のファイル操作と違う意味を持つことが多く，本稿ではユーザによるファイルの操作履歴を記録したアクセスログに加え，RMC 操作を考慮した関連ファイルの発見手法を提案した．

今後の課題として，以下４点が挙げられる．

(1) 評価実験を実施し，提案手法の有効性を確認する．

(2) 提案手法を利用して，全文検索に基づく既存のデスクトップ検索の結果を拡張して，再現率の改善を行う．

(3) RMC 操作に関係する６つのパラメータについて，適切な値を調べることが今後の課題となった．

(4) 本手法では，関連するファイルを発見する際，タスク間の関連度を指標としたため，同じタスクに属するファイルでは全て同じ扱いとなっている．今後はタスク内におけるファイル間関連度についても考えていきたい．

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．

参 考 文 献

1) Google: Google デスクトップ. 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) Beagle Team: Beagle. http://beagle-project.org/.
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 (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).
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) Watanabe, T., Kobayashi, T. and Yokota, H.: A Method for Searching Keyword-Lacking Files Based on Interfile Relationships, *OTM '08: Proceedings of the OTM Confederated International Workshops and Posters on On the Move to Meaningful Internet Systems*, Berlin, Heidelberg, Springer-Verlag, pp.14–15 (2008).
8) 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).
9) 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).
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

Vol.2010-DBS-150 No.20
Vol.2010-IFAT-99 No.20
2010/8/4

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) 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) Soules, C. A. N. and Ganger, G. R.: Connections: using context to enhance file search, *SIGOPS Oper. Syst. Rev.*, Vol.39, No.5, pp.119–132 (2005).
13) Chen, J., Guo, H., Wu, W. and Wang, W.: iMecho: an associative memory based desktop search system, *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, New York, NY, USA, ACM, pp. 731–740 (2009).
14) Chen, J., Guo, H., Wu, W. and Xie, C.: Search your memory ! - an associative memory based desktop search system, *SIGMOD '09: Proceedings of the 35th SIGMOD international conference on Management of data*, New York, NY, USA, ACM, pp.1099–1102 (2009).
15) Page, L., Brin, S., Motwani, R. and Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web, Technical report, Stanford University (1998).
16) Agrawal, R., Imieliński, T. and Swami, A.: Mining association rules between sets of items in large databases, *SIGMOD Rec.*, Vol.22, No.2, pp.207–216 (1993).