情報処理学会論文誌 Vol.62 No.2 527–537 (Feb. 2021)

# 作業を代表するフォルダの推定と分類による仮想フォルダ生成システム

西 良太[1,†1] 乃村 能成[1,a)]

受付日 2020年3月16日, 採録日 2020年11月5日

**概要**：計算機を利用して作業を行う際，過去のファイルを再参照することがある．いくつかのアプリケーションでは，ファイルの再利用支援の観点から，最近開いたファイルの一覧機能を持つ．しかし，複数のファイルが関連する作業においては，それらのファイルをフォルダに収めていることが通例で，ファイル単体よりも作業フォルダ（ワーキングディレクトリ）を一覧させるほうが有益なことがある．そこで，本論文では，ワーキングディレクトリを特定の観点でグループ化し，上位の仮想的なフォルダにまとめることで，再参照を支援するシステムを提案した．また，提案システムを実現するために，ワーキングディレクトリ推定手法と分類手法を述べ，評価を行った．最後に，提案システムの設計と実装を示した．

**キーワード**：仮想フォルダ，ファイルアクセス履歴

# Virtual Folder System by Discovering and Classifying Task Folders

Ryota Nishi[1,†1] Yoshinari Nomura[1,a)]

Received: March 16, 2020, Accepted: November 5, 2020

***Abstract:*** When working on computer, users often re-refer to past files. Some applications have a function to list recently opened files for file reuse support. However, when working using multiple files, it is generally to manage those files in folders, and it may be more useful to list the Working Directories than single files. Therefore, in this paper, we proposed a system that supports re-referencing by grouping Working Directories from a specific viewpoint and grouping them in a higher-level virtual folder. In order to realize the proposed system, the Working Directory discovering method and the classification method were described and evaluated. Finally, we show the design and implementation of the proposed system.

***Keywords:*** virtual folder, file access log

## 1. はじめに

計算機を利用して作業を行う際，過去のファイルを再参照することがある．たとえば，文書作成で過去の文書の一部を変更して用いたり，プログラミングで過去に作成したプログラムを参考にしたりすることがある．いくつかのアプリケーションでは，ファイルの再利用支援の観点から，最近開いたファイルの一覧機能を持つ．これらの支援は，ファイルをそれ単体で再利用したい場合には，有用である．しかし，複数のファイルが関連する作業においては，それらのファイルをフォルダに収めていることが通例で，ファイル単体よりも作業フォルダを一覧させるほうが有益なことがある．

そこで，本論文では，過去に行った，ユーザが意識する「作業」という粒度でフォルダを発見して利用者に提示する手法について述べる．このユーザが行った作業を代表するフォルダのことをここでは，**ワーキングディレクトリ**と呼ぶ．また，ワーキングディレクトリを特定の観点でグループ化し，上位のフォルダ内にまとめることで，再参照を支援する手法についても述べる．この上位のフォルダをここでは，**仮想フォルダ**と呼ぶ．

[1] 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University, Okayama 700–8530, Japan
[†1] 現在，サイボウズ株式会社
Presently with Cybozu, Inc.
[a)] nom@okayama-u.ac.jp

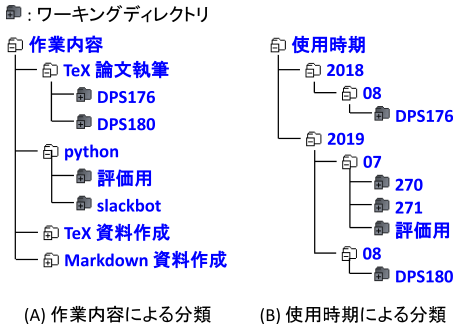


図 1 仮想フォルダによるワーキングディレクトリの分類
**Fig. 1** Classifying task folders with virtual folders.

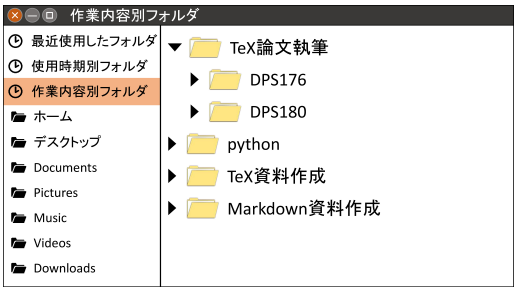


図 2 ファイルブラウザによる仮想フォルダの閲覧
**Fig. 2** Browsing virtual folders using file browser.

仮想フォルダの例を**図 1** に示す．仮想フォルダには，作業を代表するフォルダであるワーキングディレクトリを使用時期と作業内容ごとに分類し，提示する．たとえば，図 1 (A) の「DPS176」は，「TEX 論文執筆」という観点で分類された作業を代表するフォルダの 1 つであり，ワーキングディレクトリである．このように整理されていれば，たとえば，「TEX 論文執筆」という作業を行うとき，「作業内容/TeX 論文執筆」を確認することで，過去の同様の作業のワーキングディレクトリを発見できる．また，仮想フォルダは**図 2** のようにファイルブラウザを用いて閲覧できる．仮想フォルダは，ワーキングディレクトリを発見し，クラスタリングを行い，各ワーキングディレクトリにシンボリックリンクを生成することで実現している．

仮想フォルダを生成するには，以下の手順の詳細を検討する必要がある．

(1) 計算機内のワーキングディレクトリの推定
(2) ワーキングディレクトリの分類
(3) 分類結果を仮想フォルダとして提示

ユーザは，生成された仮想フォルダをファイルブラウザで閲覧することで，使用時期や作業内容といった別の観点でファイルを探すことができる．また，仮想フォルダには，ワーキングディレクトリを最小単位として提示するため，ユーザが作業を行う際に作成したフォルダ構造は破壊されない．これにより，ユーザが作業を行う際に，過去のファイルを参照することを支援できると考える．

以降では，2 章でワーキングディレクトリ推定手法を述べ，3 章でワーキングディレクトリの分類手法について述べる．次に，4 章でワーキングディレクトリ推定手法と分類手法の評価について述べる．最後に，5 章で仮想フォルダ生成システムの設計と実装について述べる．

## 2. ワーキングディレクトリ推定手法

### 2.1 目的

まず，仮想フォルダに表示するフォルダの候補となるワーキングディレクトリを発見する必要がある．そこで，本章では，計算機内のワーキングディレクトリを推定する手法 [1], [2], [3] について述べる．

ワーキングディレクトリの推定結果の良し悪しは，適合率と再現率により評価する．適合率 Precision と再現率 Recall は以下の式 (1) と式 (2) により定義される．

$$\text{Precision} = \frac{|\text{WD}_{\text{correct}} \cap \text{WD}_{\text{discovered}}|}{|\text{WD}_{\text{discovered}}|} \quad (1)$$

$$\text{Recall} = \frac{|\text{WD}_{\text{correct}} \cap \text{WD}_{\text{discovered}}|}{|\text{WD}_{\text{correct}}|} \quad (2)$$

ここで，$\text{WD}_{\text{correct}}$ は正解ワーキングディレクトリ集合で，$\text{WD}_{\text{discovered}}$ はワーキングディレクトリと推定されたフォルダ集合である．Precision，Recall は，0 から 1 の値をとり，評価値が大きいほど推定結果が優れていることを示す．

### 2.2 方針

ワーキングディレクトリとは，ユーザが行った作業を代表するフォルダである．ここで，「作業」とは，計算機内の特定のフォルダ下で行われるローカルファイルの作成・更新を指す．

本論文では，ファイルアクセス履歴に着目したワーキングディレクトリ推定手法を提案する．ファイルアクセス履歴とは，ファイルの作成，更新，削除，および移動を記録したものである．ワーキングディレクトリ推定では，アクセス履歴のうち，ユーザが行った作業にあたるファイルの作成・更新に着目して推定を行う．なお，本研究では，アクセス履歴数が多くなりすぎるため，ファイルの参照の履歴は記録しない．

ファイルアクセス履歴は，各 OS が提供するファイルシステムイベントの監視機構を用いて，イベント名，タイムスタンプ（秒単位），および対象ファイルの絶対パスをログファイルに記録することで収集する．

推定の手順を以下に示す．

**（手順 1）** ファイルアクセス履歴を作業ごとに分割
**（手順 2）** 機械的なファイル生成により作成されたアクセス履歴の削除
**（手順 3）** 残ったアクセス履歴内のファイルを包含するフォルダを抽出

（手順 3）では，**図 3** において「図 1.jpg」と「文書 1.tex」が履歴に含まれる場合，2 つのファイルを包含する「第 1

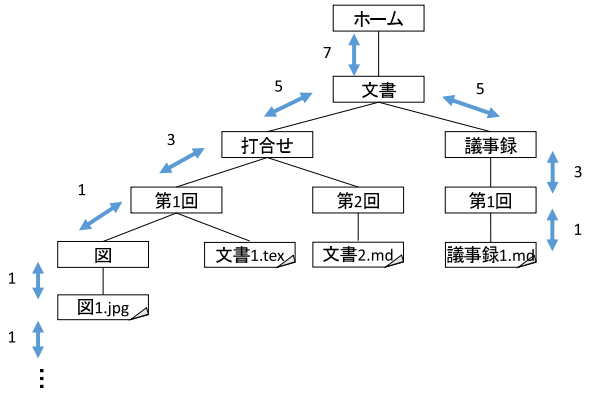


図 3 フォルダ構造の例とファイルパス間の移動コスト
**Fig. 3** Example of folder structure and cost of moving between file paths.

回」フォルダを抽出し，ワーキングディレクトリと推定する．以降で，（手順 1）と（手順 2）について詳細に説明する．

### 2.3 （手順 1）ファイルアクセス履歴を作業ごとに分割

#### 2.3.1 ファイルパス相違度による分割

ユーザが作業を切り替えた際には，作業を行うフォルダの階層が大きく変わる可能性が高い．そこで，ファイルアクセス履歴内で隣接する履歴のファイルパス相違度を定義し，相違度が閾値を超えた場合にアクセス履歴を分割する．ファイルパス相違度は以下の 2 つの観点から算出する．

(1) ファイルパス間の移動コスト
ファイルシステムをツリー構造ととらえた場合の，2 つのファイルノード間の移動コストとする．

(2) 階層の深さによる重み付け
(1) のみで相違度を算出する場合，フォルダ構造の下層部と上層部で移動コストは平等に扱われる．しかし，実際はフォルダ分けは上層になるにつれて，フォルダ間の関連は弱くなると考えられる．たとえば，図 3 では，「打合せ/第 1 回」と「打合せ/第 2 回」と比べ，「打合せ」と「議事録」の方がフォルダ間の関連が弱い．よって，上層部での移動コストは，下層部での移動コストに比べて大きくなるべきである．
そこで，ファイルパスの深さを (1) の移動コストに重み付けとして与える．重み付けの例を図 3 に示す．図 3 の例では，1–2 層目間の重みを 7，2–3 層目間の重みを 5，3–4 層目間の重みを 3，それ以降を 1 としている．

たとえば，図 3 において「図 1.jpg」と「議事録 1.md」のファイルパス相違度は 19 である．このファイルパス相違度が閾値を超えていれば，ファイルアクセス履歴を分割する．

ファイルパスの深さによる重み付けとファイルパス相違度の閾値は，ユーザごとに適切な値を設定する必要がある．これらの値を最適化する方法については，2.5 節で述べる．

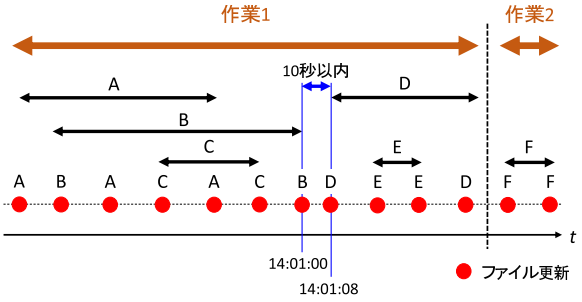


図 4 更新時刻の交錯によるアクセス履歴の分割
**Fig. 4** Dividing access log by crossing update times.

#### 2.3.2 更新時刻の交錯による分割

2.3.1 項のファイルパス相違度により分割された各ファイルアクセス履歴について，さらに更新時刻の交錯により分割する．分割の例を図 4 に示す．たとえば，図 4 では，ファイル A，B，C とファイル D，E の更新時刻がそれぞれ交錯している．

ここで，ファイル A，B，C とファイル D，E は更新時刻は交錯していないが，同じワーキングディレクトリのファイルである場合がある．このとき，これらの履歴を別の作業と見なしてしまうと，推定されるワーキングディレクトリの粒度が小さくなりすぎてしまうことが考えられる．そこで，ファイル B とファイル D のように更新時刻の差が一定時間以内の場合は同じ作業と見なし分割しない．これは，更新時刻が交錯しないファイルであっても，更新時刻の差が短い場合には同じ作業である可能性が高いと考えられるためである．本論文では，複数回の実験の結果より，更新時刻の差が 10 秒以内の場合に同じ作業と見なしている．

図 4 では，ファイル F のみ更新時刻が交錯していないため作業 2 として分割している．

### 2.4 （手順 2）機械的なファイル生成により作成されたアクセス履歴の削除

#### 2.4.1 ファイル更新密度による判定

（手順 1）の分割により得られたファイルアクセス履歴には，以下のような 3 種類の履歴が存在する．

(1) インストール等の機械的な生成による更新のみを含む履歴

(2) ユーザによる手動の更新とコンパイル等の機械的な生成による更新が混在する履歴

(3) ユーザによる手動の更新のみを含む履歴

上記のうち，(1) のようなソフトウェアのインストール等により機械的に生成されたアクセス履歴により推定されるワーキングディレクトリは，ユーザが意識するワーキングディレクトリとは異なる可能性が高い．このため，(1) のような機械的なファイル生成により作成されたファイルアクセス履歴は削除すべきであると考える．そこで，本節

では，ファイルの作成・更新の密度を用いて機械的なファイル生成により作成されたファイルアクセス履歴を判定する手法について述べる．

2.3 節で分割した更新履歴について，以下の式 (3) によりファイル更新密度 density を算出する．この density が閾値を超えていれば，機械的なファイル更新と判定する．

$$\text{density} = \frac{N}{T_{\text{end}} - T_{\text{first}}} [\text{times/sec}] \quad (3)$$

ここで，$T_{\text{first}}$ は，分割後の更新履歴の最初の更新時刻であり，$T_{\text{end}}$ は，分割後の更新履歴の最後の更新時刻である．また，$N$ は，分割後の更新履歴に含まれるファイルの作成・更新の回数である．

インストール等の機械的な生成による更新のみを含む履歴では，このファイル更新密度が，他のアクセス履歴と比較して高いとして閾値を設定する．

#### 2.4.2 ファイル更新密度の閾値の検討

機械的なファイル更新を判定可能なファイル更新密度の閾値を検討するために，閾値を少しずつ変化させてワーキングディレクトリ推定結果の評価を行う．評価の指標としては，適合率，再現率，およびこれら 2 つの調和平均である $F$ 値を用いる．閾値は，0.1〜2.0 の範囲で 0.05 ずつ変化させて評価を行う．評価には，2019 年 9 月 5 日〜2019 年 9 月 19 日の 2 週間に収集した 65,946 件のファイルアクセス履歴を用いる．

**図 5** にファイル更新密度の閾値を変化させたときの $F$ 値，適合率，および再現率の変化を示す．図 5 より，$F$ 値は，閾値 0.3 で最大となり，それ以降は徐々に低下していることが分かる．また，再現率は閾値 0.3 以降で一定に保たれていることから，$F$ 値の低下は適合率の低下によるものと分かる．これは，ファイル更新密度の閾値が大きすぎると，機械的なファイル生成による更新履歴を削除しきれず，ワーキングディレクトリでないフォルダが推定結果に増え，適合率が低下しているためと考えられる．一方，再現率については，閾値が大きくなっても，高い値を保っていることから，ユーザが意識的に作成したファイルによる更新履歴は削除されずに残っていることが分かる．よって，本実験においては，閾値は 0.3 が妥当であるといえる．このため，ファイル更新の密度が 0.3 以上であれば，機械的なファイル生成と見なし，ファイルアクセス履歴を削除する．

### 2.5 ワーキングディレクトリ推定手法におけるパラメータの最適化

ワーキングディレクトリ推定手法では，2.3.1 項のファイルパス相違度による分割で，以下の 2 つのパラメータを設定する必要がある．

(1) ファイルパス相違度の算出に用いる階層の深さによる重み付け

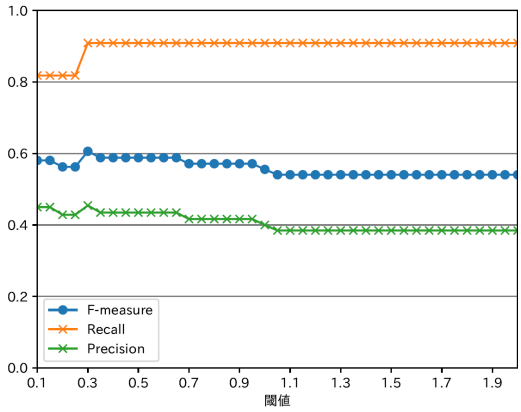


**図 5** ファイル更新密度の閾値を変化させたときの $F$ 値，適合率，および再現率の変化

**Fig. 5** Changes in F, Precision, and Recall when changing file update density threshold.

**表 1** ワーキングディレクトリ推定手法で用いるパラメータの探索範囲

**Table 1** Search range of parameters by bayesian optimization method.

| 要素 | 探索範囲 |
|---|---|
| 1–2 層目間の重み | 1〜15 |
| 1 層深くなるごとの重みの減少量 | 1〜5 |
| ファイルアクセス履歴を分割する閾値 | 1〜50 |

(2) ファイルアクセス履歴を分割するファイルパス相違度の閾値

これらのパラメータは，ユーザごとに適切な値をあらかじめ設定する必要がある．これは，計算機の使用者によってホームディレクトリのフォルダ構成が異なるためである．そこで，本節では，ワーキングディレクトリ推定手法において設定する必要のある上記の 2 つのパラメータを最適化する手法を述べる．

最適なパラメータの探索には，ベイズ最適化 [4] を用いる．ベイズ最適化とは，未知の形状のブラックボックス関数の最大値を求めるための手法である．本研究では，ベイズ最適化を用いてワーキングディレクトリ推定結果の $F$ 値を出力とするブラックボックス関数を最大化し，最適なパラメータを探索する．ベイズ最適化における獲得関数には，Expectation Improvement（EI）を用い，探索の試行回数は 30 回とする．

(1) のファイルパス相違度の算出に用いる階層の深さによる重み付けは，可変長のリストで表現するパラメータである．ベイズ最適化を行う際には固定長のパラメータの方が扱いやすい．このため，重み付けと閾値を固定長の 1 つのリストで表現する．本研究では，パラメータを**表 1** の 3 つの要素からなるリストで表現する．ここで，重みは階層が深くなるにつれて減少し，最小の重みは 1 とする．

たとえば，重みが `[7, 5, 3, 1]` で閾値が 19 であることを表現する場合，`[7, 2, 19]` と記述できる．これは，1–2

表 2 著者の計算機で収集したアクセス履歴を用いた最適化で得られたパラメータ
Table 2 Parameters used in evaluate.

| | 値 |
|---|---|
| ファイルパス相違度の重み付け | [15,11,7,3,1] |
| ファイルパス相違度の閾値 | 27 |
| ファイル更新密度の閾値 | 0.3 |

層目間の重みが 7，1 層深くなるごとに重みが 2 減少，閾値は 19 であることを表現している．本研究では，このような長さ 3 の固定長のリストで表現したパラメータを最適化する．また，ベイズ最適化による各パラメータの探索範囲は表 1 のとおりである．

2019 年 9 月 5 日〜2019 年 9 月 19 日に著者の計算機で収集したアクセス履歴を用いてパラメータの最適化を行った．これにより得られたパラメータを**表 2** に示す．

本手法により最適化したパラメータを用いたワーキングディレクトリ推定手法の評価については，4.2 節で述べる．

### 2.6 本手法で想定するフォルダ構成

本手法では，ファイルアクセス履歴を作業ごとに分割することで，分割された履歴内のファイルを包含するフォルダをワーキングディレクトリと推定する．アクセス履歴の分割では，まず，ファイルパス相違度を用いて分割する．

ファイルパス相違度による分割だけでは，同じ階層にワーキングディレクトリが複数存在する場合，うまく分割できない．そこで，分割された履歴を，さらに更新時刻の交錯を用いて分割している．

しかし，更新時刻の交錯を用いて分割したとしても，フォルダ構成の最上層に近い位置に深い階層を持つワーキングディレクトリが存在する場合，提案手法ではうまく推定できない可能性がある．これは，上層の方がファイルパス相違度を算出するための移動コストの重み付けが大きく，不必要に分割されてしまうためである．

このようなフォルダ構成の場合以外は，問題なく本手法を適用可能であると考える．

## 3. ワーキングディレクトリの分類

### 3.1 分類観点と分類手法

作業内容や使用時期によって整理した仮想フォルダを生成するためには，ワーキングディレクトリを分類する必要がある．本節では，ワーキングディレクトリの分類の観点を述べ，それぞれの観点に対する分類手法を述べる．分類の観点としては，以下の 2 つが考えられる．

**（観点 1）** 作業内容による分類

図 1 (A) のようにワーキングディレクトリで行われた作業の内容よって分類する．このように分類されていれば，たとえば，「TEX による論文執筆」という作業を行う際に，過去に作成した同様の作業のワーキングディレクトリを設定ファイルや TEX 固有の記法の確認のために参照することがある．このとき，図 1 (A) のように分類されていれば，「TeX 論文執筆」という仮想フォルダを確認することで，目的のワーキングディレクトリを発見できる．

**（観点 2）** 使用時期による分類

ワーキングディレクトリを使用した時期によって分類する．この分類には，図 1 (B) のような年月による分類と，最近使用したワーキングディレクトリといった分類が考えられる．計算機上で行う作業には，周期を持って繰り返し行われる作業が存在する．たとえば，新年度の初めである毎年 4 月に行う必要のある作業や，毎月 1 回行われる打合せのために資料を作成するという作業が考えられる．このため，使用時期によって分類されていれば，たとえば，「使用時期/2019/04」のような仮想フォルダを確認することで，毎年 4 月に使用するワーキングディレクトリを発見できる．

（観点 1）の作業内容による分類では，ワーキングディレクトリと推定されたフォルダをクラスタリングし，分類結果を提示する．フォルダのクラスタリング手法 [5] について，3.2 節で詳細に説明する．

（観点 2）の使用時期による分類では，ファイルアクセス履歴からワーキングディレクトリの使用時期を特定することで分類する．また，以下の 2 つの方法で提示する．

(1) 年月による分類の提示
(2) 最近使ったワーキングディレクトリの提示

上記 (2) の提示方法については，どの程度過去のものを提示するか検討する必要がある．これについて，調査を行った結果，過去 3 週間分を提示するのが妥当であるという結果が得られた [1].

### 3.2 フォルダのクラスタリング手法

#### 3.2.1 フォルダの作業を表現する特徴

フォルダをクラスタリングするためには，フォルダで行われた作業を表現する特徴が必要となる．フォルダの作業を表現する特徴として，フォルダ内のファイル種別，フォルダの階層構造の形状，ファイルの名称，およびファイルアクセス履歴等があげられる．本論文では，これらの特徴の中から以下の 2 つを用いる．

**（特徴 1）** ファイル種別

**（特徴 2）** ファイル更新の時系列

（特徴 1）のファイル種別は，ファイルの拡張子により識別する．フォルダ内に存在するファイル種別は，そのフォルダでユーザが行った作業を表していると考えられる．たとえば，「TEX による文書作成」のフォルダ内には，「`tex`」や「`pdf`」といった拡張子を持つファイルが存在する可能性が高い．また，（特徴 2）について，同様の作業のフォ

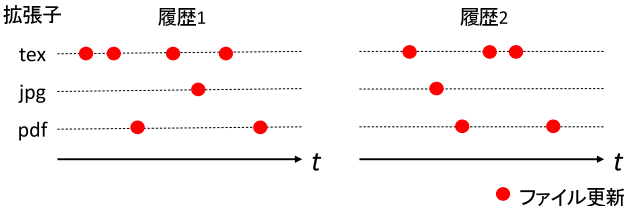


図 6 ワーキングディレクトリ推定手法で作業ごとに分割されたファイル更新履歴

**Fig. 6** Example of access log devide by working directory discovering method.

表 3 隣接する更新履歴のファイル拡張子の組を集計した結果

**Table 3** Number of pairs of file extensions in adjacent update history.

| | tex-tex | tex-pdf | pdf-tex | tex-jpg | jpg-tex | jpg-pdf |
|---|---|---|---|---|---|---|
| 履歴 1 | 1 | 2 | 1 | 1 | 1 | 0 |
| 履歴 2 | 1 | 1 | 1 | 1 | 0 | 1 |
| 合計 | 2 | 3 | 2 | 2 | 1 | 1 |

ルダでは，ファイル更新の時系列に類似性が存在すると考える．たとえば，「TEX による文書作成」という作業では，ユーザはソースファイルを編集し，それをコンパイルすることで PDF の文書を生成する．このため，「tex」ファイルが更新された後「pdf」ファイルが更新されるという更新順序が頻出する可能性が高い．

次項で，（特徴 1）と（特徴 2）を表現する特徴ベクトルの作成方法を述べる．

#### 3.2.2 特徴ベクトルの作成

ファイル種別とファイル更新の時系列を用いて，フォルダの作業を表現する特徴ベクトルを作成する方法を**図 6** および**表 3** に示し，手順を以下に説明する．

**（手順 1）** 対象のフォルダのファイルアクセス履歴を抽出
ワーキングディレクトリ推定手法において作業ごとに分割されたファイルアクセス履歴を使用する．図 6 にアクセス履歴の例を示す．

**（手順 2）** 隣接するファイルアクセス履歴の集計
（手順 1）により得られた履歴に対して，隣接する更新履歴のファイル拡張子の組を集計する．図 6 の集計結果は，表 3 となる．そして，履歴 1 と履歴 2 について集計した結果を合計し，特徴ベクトルを作成する．

また，フォルダのファイルアクセス履歴数による差を軽減するため，$L^2$ ノルム（長さ）が 1 になるように正規化する．本研究では，これにより得られた特徴ベクトルを用いて階層的クラスタリングにより，フォルダを作業ごとに分類する．

## 4. 評価

### 4.1 評価項目

本章では，以下の 2 つの項目について評価を行う．これにより，提案システムにおいて，どの程度正しくワーキングディレクトリの推定と分類を行えるかを評価する．

**（評価 1）** ワーキングディレクトリ推定手法の精度

**（評価 2）** 作業を表現する特徴ベクトルの評価

以降では，それぞれの評価について詳細に説明する．

表 4 実験期間中に収集したファイルアクセス履歴数

**Table 4** Number of file access log collected in experiment.

| | 12/2（月） | 12/3（火） | 12/4（水） | 12/5（木） | 12/6（金） | 合計 |
|---|---|---|---|---|---|---|
| 著者 | 1,856 | 725 | 544 | 1,007 | 202 | 4,334 |
| 学生 A | 1,955 | 330 | 401 | 1,638 | 99 | 4,423 |
| 学生 B | 34 | 41,351 | 365 | 407 | 35 | 42,192 |
| 学生 C | 102 | 96 | 406 | 129 | 500 | 1,233 |
| 学生 D | 2 | 1,000 | 276 | 2,551 | 397 | 4,226 |
| 学生 E | 555 | 48,521 | 2,953 | 392 | 56,410 | 108,831 |

表 5 実験期間中に各ユーザが作業を行ったワーキングディレクトリ数

**Table 5** Number of Working Directories used during the experiment.

| | 12/2（月） | 12/3（火） | 12/4（水） | 12/5（木） | 12/6（金） |
|---|---|---|---|---|---|
| 著者 | 2 | 4 | 2 | 3 | 2 |
| 学生 A | 2 | 1 | 3 | 2 | 2 |
| 学生 B | 1 | 2 | 2 | 1 | 1 |
| 学生 C | 1 | 1 | 1 | 1 | 1 |
| 学生 D | 0 | 1 | 1 | 1 | 2 |
| 学生 E | 1 | 1 | 3 | 1 | 2 |

### 4.2 ワーキングディレクトリ推定手法の評価

#### 4.2.1 評価方法

本評価では，2019 年 12 月 2 日～2019 年 12 月 6 日の 5 日間に著者の計算機で収集したアクセス履歴を使用して評価を行う．また，著者以外のユーザに対してもワーキングディレクトリ推定手法が有効かを確認するために，著者の在籍する研究室の学生 5 名にも同じ期間にファイルアクセス履歴を収集してもらい評価を行う．

ワーキングディレクトリ推定手法では，いくつかのパラメータをあらかじめ設定する必要がある．これらのパラメータは，ユーザごとに最適な値を設定する必要がある．しかし，提案システム導入初期の環境を想定すると，最適化に必要なデータが蓄積されていないため，一般的なパラメータを初期値として設定しておく必要がある．そこで，本評価では，著者のデータを用いて最適化した表 2 に示すパラメータを用いる．また，機械的な生成と見なすファイル更新密度の閾値としては，2.4.2 項の実験で得た 0.3 を使用する．

収集したファイルアクセス履歴を**表 4** に示す．また，**表 5** に，各日のワーキングディレクトリ数を示す．本評価では，このワーキングディレクトリを正解データとして評価を行う．

ワーキングディレクトリの推定手法を評価するための指標としては，2.1 節の式 (1)，式 (2) で定義した，適合率 Precision と再現率 Recall を用いる．

表 6 ワーキングディレクトリ推定手法の評価結果
**Table 6** Evaluation results of working directory discovering method.

| | Precision | Recall | ReductionRate |
|---|---|---|---|
| 著者 | 0.45 | 1.00 | 0.84 |
| 学生 A | 0.26 | 1.00 | 0.61 |
| 学生 B | 0.26 | 1.00 | 0.55 |
| 学生 C | 0.42 | 1.00 | 0.49 |
| 学生 D | 0.62 | 1.00 | 0.49 |
| 学生 E | 0.17 | 0.67 | 0.88 |
| 学生 A～E の平均 | 0.33 | 0.93 | 0.66 |

また，どの程度候補を減らせたかを示す削減率を ReductionRate として，以下の式 (4) により評価する．ここで，$\mathrm{DIR_{all}}$ は，アクセス履歴に含まれるすべてのフォルダ集合である．

$$\mathrm{ReductionRate} = \frac{|\mathrm{DIR_{all}}| - |\mathrm{WD_{discovered}}|}{|\mathrm{DIR_{all}}|} \quad (4)$$

上記の Precision，Recall，および ReductionRate は，0 から 1 の値をとり，評価値が大きいほど推定結果が優れていることを示す．

#### 4.2.2 評価結果

ワーキングディレクトリ推定手法の評価結果を**表 6** に示す．表 6 には，著者の評価結果，学生 A～E の評価結果，および学生 A～E の評価結果の平均を示す．

表 6 より，著者の評価結果の Recall の平均は 1.00 と高く，5 日間に使用したすべてのワーキングディレクトリを正しく推定できたことが分かる．また，著者の評価結果の Precision は 0.45 となっており，推定されたフォルダのうち約半分は，著者がワーキングディレクトリと考えていないフォルダであった．また，著者の評価結果の ReductionRate の平均は 0.84 と高く，ワーキングディレクトリの候補を大幅に絞れたことが分かる．

次に，著者以外の学生の結果では，Precision の 5 名の平均は 0.33，Recall の 5 名の平均は 0.93 である．この結果は，著者の結果より評価値が少し低いが，ほとんど同じ結果といえる．

しかし，学生ごとの結果を個別に見た場合，学生 E の結果のみ評価値が低くなっていることが分かる．学生 E は，実験期間中にソフトウェアのインストールのみを行った日があり，学生 E はこれを作業として認識していた．しかし，提案手法では，このような更新履歴は，機械的な生成として作業とは見なしていない．このように，ユーザの認識する作業と提案手法における作業の定義に差があったことにより，評価結果が低くなっている．

著者の結果より評価値が少し低かった理由としては，使用したパラメータが著者のデータで最適化されているためと考えられる．各学生ごとにパラメータの最適化を行うことで，評価結果が改善されることが予想される．

また，ReductionRate については，著者と学生 5 名の結果でばらつきがあった．ReductionRate は，ユーザが行った作業によって大きく変化すると考えられる．これは，ソフトウェアのインストール等を多く行った場合，ワーキングディレクトリ推定手法（手順 2）において多くの履歴が削除されることにより，ReductionRate が高くなるためである．

以上のことより，著者以外のユーザに対してもワーキングディレクトリ推定手法は有効であるといえる．

### 4.3 作業を表現する特徴ベクトルの評価

#### 4.3.1 評価方法

3.2 節で述べた，フォルダで行われた作業を表現する特徴ベクトルの評価を行う．本評価では，3.2 節で述べた手法により得られた特徴ベクトルを用いて，フォルダのクラスタリングを行い，クラスタリングの精度を評価する．フォルダのクラスタリング結果は，以下の 2 つの観点から評価を行う．

**（観点 1）** 同様の作業のフォルダがどれだけ同じクラスタに分類されたか

**（観点 2）** 生成されたクラスタがどの程度同様の作業のフォルダで占められているか

（観点 1）に対する評価が高いほど，同様の作業のフォルダが単一のクラスタに集中して存在しており，同様の作業のフォルダを探す際に，少数のクラスタを確認するだけでよいという利点がある．ただし，クラスタ内に異なる作業のフォルダが混在するかどうかは保証しない．つまり，すべての対象のフォルダが 1 つのクラスタに分類されたとしても，この観点に対する評価は高くなる．そこで，クラスタの純度を評価する（観点 2）が必要となる．（観点 2）に対する評価が高いほど，クラスタが同様の作業のフォルダで占められており，異なる作業のフォルダを取り除く手間が小さくなる．

クラスタリング結果の評価には，InversePurity と Purity という指標 [6] が存在する．本評価では，（観点 1）を評価する指標として，InversePurity，（観点 2）を評価する指標として，Purity を用いる．これらの指標を用いて，上記の（観点 1）と（観点 2）を定量的に評価する．

InversePurity は，ある正解クラスが 1 つのクラスタに集中している度合いを表す．これにより，ある作業のフォルダが，どの程度 1 つのクラスタに集中して分類されているかを評価できる．また，Purity は，生成されたクラスタが単一の正解クラスのデータで占められている度合いを表す．これにより，生成されたクラスタの純度，すなわち，クラスタ内に望ましくない作業のフォルダが含まれていないかを評価できる．

InversePurity と Purity は，以下の式 (5) と式 (6) で定義される．以降では，$N$ をクラスタリング対象のデータ数，$C$ を生成されたクラスタの集合，$A$ を正解クラスの集合と

表 9 フォルダのクラスタリング手法による実験データの分類結果
**Table 9** Classification result of experimental data by folder clustering method.

| | 作業 1 | 作業 2 | 作業 3 | 作業 4 | 作業 5 | 作業 6 | 作業 7 | 作業 8 | 作業 9 | 作業 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| クラスタ 1 | - | - | - | - | - | 100%(3) | - | - | - | - |
| クラスタ 2 | - | - | - | - | 100%(4) | - | 66.7%(2) | - | - | - |
| クラスタ 3 | - | - | - | - | - | - | - | 100%(1) | - | - |
| クラスタ 4 | - | - | - | - | - | - | - | - | 100%(3) | 100%(3) |
| クラスタ 5 | - | 100%(2) | - | - | - | - | 33.3%(1) | - | - | - |
| クラスタ 6 | 100%(2) | - | - | - | - | - | - | - | - | - |
| クラスタ 7 | - | - | 100%(1) | - | - | - | - | - | - | - |
| クラスタ 8 | - | - | - | 100%(1) | - | - | - | - | - | - |

表 7 フォルダのクラスタリングに用いる実験データ
**Table 7** Experimental data used to evaluate folder clustering method.

| 通番 | 作業内容 | フォルダ数 |
|---|---|---|
| 作業 1 | Rails による開発 | 2 |
| 作業 2 | Python による開発 | 2 |
| 作業 3 | Node.js による開発 | 1 |
| 作業 4 | シェルスクリプト作成 | 1 |
| 作業 5 | TEX による打合せ資料作成 | 4 |
| 作業 6 | TEX による報告書作成 | 3 |
| 作業 7 | TEX による講義レポート作成 | 3 |
| 作業 8 | Markdown による打合せ資料作成 | 1 |
| 作業 9 | Markdown による報告書作成 | 3 |
| 作業 10 | Markdown による議事録作成 | 3 |
| 計 | - | 23 |

する．

$$\text{InversePurity} = \sum_{j=1}^{|A|} \frac{|A_j|}{N} \max_i(\text{Recall}(C_i, A_j)) \quad (5)$$

$$\text{Purity} = \sum_{i=1}^{|C|} \frac{|C_i|}{N} \max_j(\text{Precision}(C_i, A_j)) \quad (6)$$

式 (5) において $|A|$ は正解クラスの数であり，$|A_j|$ は $j$ 番目の正解クラスに属するデータ数である．また，ある正解クラス $A_j$ に対するクラスタ $C_i$ の再現率 $\text{Recall}(C_i, A_j)$ は，式 (7) によって定義される．

$$\text{Recall}(C_i, A_j) = \frac{|C_i \cap A_j|}{|A_j|} \quad (7)$$

式 (6) において $|C|$ は生成されたクラスタ数，$|C_i|$ は $i$ 番目のクラスタに属するデータ数である．また，ある正解クラス $A_j$ に対するクラスタ $C_i$ の適合率 $\text{Precision}(C_i, A_j)$ は，以下の式 (8) によって定義される．

$$\text{Precision}(C_i, A_j) = \frac{|C_i \cap A_j|}{|C_i|} \quad (8)$$

また，（観点 1）と（観点 2）を総合的に評価するために，InversePurity と Purity の調和平均である $F$ 値を用いる．これは，以下の式 (9) によって定義される．

$$F = \frac{2}{\text{InversePurity}^{-1} + \text{Purity}^{-1}} \quad (9)$$

表 8 フォルダのクラスタリングの評価結果
**Table 8** Evaluation results of folder clustering method.

| 指標 | 評価値 |
|---|---|
| InversePurity | 0.9565 |
| Purity | 0.7391 |
| $F$ 値 | 0.8339 |

#### 4.3.2 評価環境

評価に用いる実験データについて述べる．実験には，著者の 1 名が日常的に大学の講義や研究活動等で使用する計算機を用いた．ファイルアクセス履歴の収集期間は 2017 年 6 月 14 日～2018 年 8 月 8 日である．この計算機内のいくつかの作業のワーキングディレクトリを実験データとして用いる．

**表 7** に，クラスタリング対象のワーキングディレクトリの作業内容とフォルダ数を示す．表 7 の 23 個のフォルダは，計算機の使用者の主観により 10 種類の作業に分類した．評価においては，表 7 に示した作業を正解の分類とする．本評価では，これら 23 個のフォルダを 10 種類の作業にどのくらい正確にクラスタリングできるかを評価する．

#### 4.3.3 評価結果

クラスタリング結果の InversePurity，Purity，および $F$ 値を**表 8** に示す．表 8 より，InversePurity は 0.9565 であり，Purity は 0.7391 であった．InversePurity が高いことから，同様の作業のフォルダが 1 つのクラスタに集中して分類されていることが分かる．また，$F$ 値もある程度高いことより，Purity も保たれているといえる．

**表 9** に具体的な分類結果を示す．表 9 では，表 7 の各作業のワーキングディレクトリがどのような割合で各クラスタに分類されているかを示しており，括弧内の数字はフォルダ数である．分類の結果，表 7 のワーキングディレクトリから 8 個のクラスタを得た．表 9 より，1 つのクラスタ内に最大で 2 種類の作業が混在していることが分かる．また，作業 7 の「TEX による講義レポート作成」以外の作業はすべて 1 つのクラスタに対して分類されており，1 つのクラスタを確認するだけで同様の作業のフォルダをすべて発見できる．

よって，1 つのクラスタを確認することで同様の作業の

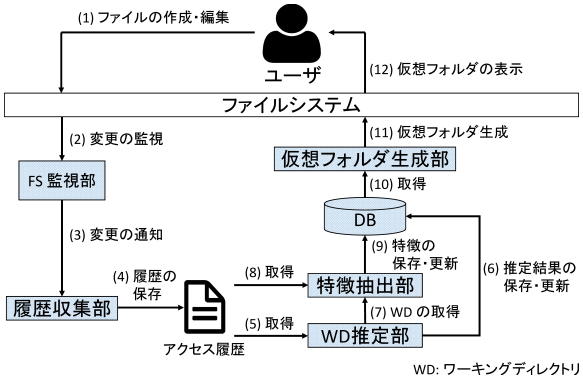


図 7 仮想フォルダ生成システムの構成
**Fig. 7** System design.

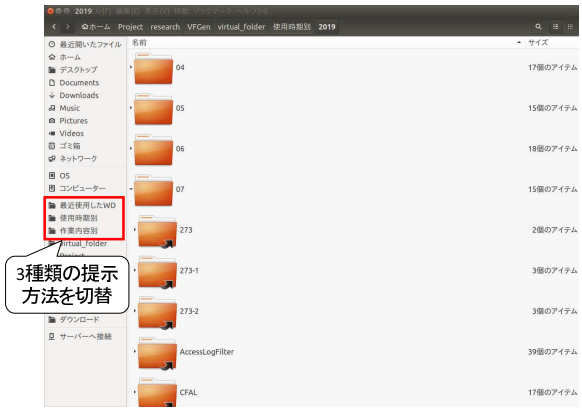


図 8 提案システムによって生成された仮想フォルダ
**Fig. 8** Virtual folder generated by proposal system.

ワーキングディレクトリを多く発見できるといえる．このことより，同様の作業のフォルダを探すことを目的に分類結果を利用する場合，クラスタリング結果は有用であるといえる．

以上より，作業を表現する特徴ベクトルは，フォルダを作業ごとに分類するのに有用であるといえる．

## 5. 設計と実装

### 5.1 提案システムの構成

本章では，2 章で述べた手法により推定したワーキングディレクトリを 3 章で述べた手法により分類し，仮想フォルダとしてまとめて提示するシステムの設計と実装について述べる．提案システムの構成を図 7 に示す．提案システムは，以下の 6 つの処理部から構成される．

(1) FS 監視部
ファイルシステムの変更を監視し，履歴収集部に変更を通知する．

(2) 履歴収集部
ファイルアクセス履歴を収集し，保存する．

(3) WD（ワーキングディレクトリ）推定部
2 章で述べた手法によりワーキングディレクトリを推定し，データベースに保存する．

(4) 特徴抽出部
3.2 節で述べたクラスタリングに用いる特徴を抽出し，データベースに保存する．

(5) DB（データベース）
ワーキングディレクトリのパス，使用期間，およびクラスタリングのための特徴量を保存する．

(6) 仮想フォルダ生成部
ワーキングディレクトリをデータベースから取得し，シンボリックリンクを作成することで仮想フォルダを生成する．ユーザは，生成された仮想フォルダを確認することで，ワーキングディレクトリにアクセスできる．

### 5.2 実装

仮想フォルダ生成システム polaris [7] の実装を行った．提案システムは，Linux，macOS，および Windows で動作する．図 8 に，仮想フォルダをファイルブラウザで閲覧した様子を示す．提案システムでは，「最近使用した WD」，「使用時期別」，および「作業内容別」の 3 種類の提示方法を切り替えて閲覧できる．図 8 は，「使用時期別/2019」の仮想フォルダを閲覧している様子であり，さらに月ごとに分類された仮想フォルダが表示されている．目的の月の仮想フォルダを開くことで，その月に使用したワーキングディレクトリの一覧が表示される．たとえば，図 8 では，「使用時期別/2019/07」の仮想フォルダを開いており，2019 年 7 月に使用したワーキングディレクトリの一覧が表示されている．

## 6. 関連研究

仮想フォルダを生成することでファイル検索を支援する手法として，1 つの作業に関連するファイルが頻繁に近い時間に参照されることを使用した仮想フォルダ生成手法が提案されている [8]．文献 [8] は，計算機内の 1 つの作業に関連するファイルを仮想フォルダに集約して提示するため，仮想フォルダの中身は実際のファイル配置とは異なる．一方，本論文では，ワーキングディレクトリに着目し，それらを作業内容と使用時期によって再分類した仮想フォルダを生成することでファイル検索を支援する．したがって，提案システムでは，使用者に馴染みのあるファイル配置で閲覧できる．

また，ファイルブラウザを拡張することでファイル検索を支援する手法として，ユーザが次にアクセスする可能性が高いフォルダを予測して，ファイルブラウザ上で強調表示する手法が提案されている [9], [10]．文献 [9], [10] では，強調表示されたフォルダをクリックして階層移動することで目的のファイルを探す．一方，本論文では，仮想フォルダにより計算機内のワーキングディレクトリを再分類して

提示している．このため，ユーザが次にアクセスする可能性が高いフォルダを予測できれば，予測結果を提示する仮想フォルダを生成することで，文献 [9], [10] の手法よりも階層移動の回数が少なくて済むと考えられる．

また，過去のファイルの再参照を支援する研究として，ファイルに対して時間，場所，および作業内容の注釈を付与することで，ファイルを使用した状況から検索する手法が提案されている [11]．文献 [11] では，時間，場所，および作業内容をキーワードとして過去に使用したファイルを検索する．一方，本論文では，ファイルブラウザを用いて，使用時期と作業内容ごとに分類された仮想フォルダにより目的のフォルダを探索する．文献 [11] では，著者らが提案した検索システムの利用調査を実施しており，ファイルを使用した時間，場所，および同時に行っていた作業内容がファイル検索の手がかりとして有用であることが示されている．本論文においても，使用時期や作業内容によってフォルダを分類しており，この分類はファイル参照の際に有用であると考えられる．

また，ファイル検索に関する研究としては，ファイルアクセス履歴から算出されるファイル間関連度を用いたファイル検索手法 [12], [13] が提案されている．このほかにも，ファイルアクセス履歴から 1 つの作業に関連するファイルを発見し，作業間の関連性を求めることでファイル検索を支援する手法 [14] が提案されている．これらの研究では，キーワードを入力することで，全文検索の結果とともに，それらに関連が強いファイルを提示することで，キーワードを含まないファイルの検索を実現している．本論文では，1 つの作業に関連するファイルがワーキングディレクトリとしてまとめられていることに着目し，ワーキングディレクトリを再分類した仮想フォルダを生成することでファイルの探索を支援する．このため，本論文で提案する仮想フォルダ生成システムにおいても，ワーキングディレクトリで行った作業を手がかりにファイルを探索することで，キーワードを含まないファイルを参照したいという要求に対処可能であると考える．

## 7. おわりに

本論文では，計算機内のワーキングディレクトリを作業内容や使用時期ごとに再分類した仮想フォルダを生成することで，過去のファイル参照を支援するシステムを提案した．また，提案システムを実現するために，ワーキングディレクトリ推定手法とフォルダのクラスタリング手法について述べ，評価を行った．

ワーキングディレクトリ推定手法の評価においては，著者の計算機を使用した評価と，著者の所属する研究室の学生 5 名の計算機を使用した評価を行い，比較を行った．評価には，適合率 Precision，再現率 Recall，およびどの程度ワーキングディレクトリの候補を絞れたかを示す削減率 ReductionRate を用いた．評価の結果，どちらの評価結果も Recall が非常に高く，ワーキングディレクトリをほとんど正しく推定できていることを示した．

フォルダクラスタリング手法の評価においては，同様の作業のフォルダがどれだけ同じクラスタに分類されたかを示す指標として InversePurity，生成されたクラスタがどの程度同様の作業のフォルダで占められているかを示す指標として Purity を用いた．また，これらを総合的に評価するために 2 つの評価値の調和平均である $F$ 値を用いた．評価の結果，InversePurity が高く，1 つのクラスタを確認することで同様の作業のワーキングディレクトリを多く発見できることが分かる．また，$F$ 値もある程度高い結果を示しており，Purity も保たれていることが分かった．このことより，同様の作業のフォルダを探すことを目的に分類結果を利用する場合，フォルダのクラスタリング手法は有用であるといえる．

最後に，提案システムの設計と実装について述べた．

残された課題としては，フォルダのクラスタリング結果に適切なフォルダ名を付与する手法の検討，ユーザからフィードバックを得る手法の検討，および提案システムを使用した場合にどの程度ファイル参照にかかる時間を削減できるかの評価がある．

### 参考文献


[1] 西 良太，乃村能成：作業を代表するフォルダの推定による仮想フォルダ生成システムの提案，情報処理学会研究報告，Vol.2019-DPS-180, No.20, 第 180 回マルチメディア通信と分散処理研究会，pp.1–7 (2019).

[2] 池田ゆう子，乃村能成：Inbox による文書整理システム，情報処理学会研究報告，Vol.2016-DPS-167, No.30, pp.1–7 (2016).

[3] 西 良太，乃村能成：ファイル更新履歴に着目した作業を代表するフォルダの推定手法，第 18 回情報科学技術フォーラム（FIT2019）講演論文集，第 4 分冊，pp.295–296 (2019).

[4] Mockus, J., Tiesis, V. and Zilinskas, A.: The application of Bayesian methods for seeking the extremum, *Towards global optimization*, Vol.2, No.117-129, p.2 (1978).

[5] 西 良太，乃村能成：ファイル更新の時系列に着目したフォルダのクラスタリング手法，情報処理学会研究報告，Vol.2018-DPS-176, No.4, pp.1–8 (2018).

[6] Amigó, E., Gonzalo, J., Artiles, J. and Verdejo, F.: A comparison of extrinsic clustering evaluation metrics based on formal constraints, *Information Retrieval*, Vol.12, No.4, pp.461–486 (online), DOI: 10.1007/s10791-008-9066-8 (2009).

[7] nomlab: polaris, GitHub (online), available from ⟨https://github.com/nomlab/polaris⟩ (accessed 2020-03-13).

[8] 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).

[9] Fitchett, S., Cockburn, A. and Gutwin, C.: Improving Navigation-based File Retrieval, *Proc. SIGCHI Conference on Human Factors in Computing Systems, CHI '13*, pp.2329–2338, ACM (online), DOI: 10.1145/

2470654.2481323 (2013).
[10] Fitchett, S., Cockburn, A. and Gutwin, C.: Finder Highlights: Field Evaluation and Design of an Augmented File Browser, *Proc. SIGCHI Conference on Human Factors in Computing Systems, CHI '14*, pp.3685–3694, ACM (online), DOI: 10.1145/2556288.2557014 (2014).
[11] Deng, T., Zhao, L., Wang, H., Liu, Q. and Feng, L.: ReFinder: A Context-Based Information Refinding System, *IEEE Trans. Knowledge and Data Engineering*, Vol.25, No.9, pp.2119–2132 (online), DOI: 10.1109/TKDE.2012.157 (2013).
[12] 渡部徹太郎，小林隆志，横田治夫：ファイル検索に向けたアクセスログからのファイル間関連度の導出，日本データベース学会 Letters，Vol.6, No.2, pp.65–68 (2007).
[13] 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，全国大会講演論文集，Vol.70, pp.231–232（オンライン），入手先 ⟨https://ci.nii.ac.jp/naid/110006863963/⟩ (2008).
[14] Wu, Y., Otagiri, K., Watanabe, Y. and Yokota, H.: A File Search Method Based on Intertask Relationships Derived from Access Frequency and RMC Operations on Files, *Database and Expert Systems Applications*, Hameurlain, A., Liddle, S.W., Schewe, K.-D. and Zhou, X. (Eds.), pp.364–378, Berlin, Heidelberg, Springer Berlin Heidelberg (2011).

**西 良太** （正会員）

平成 30 年岡山大学工学部情報系学科卒業．令和 2 年同大学大学院電子情報システム工学専攻博士前期課程修了．同年サイボウズ株式会社入社．現在，同社にてグループウェアの開発に従事．

**乃村 能成** （正会員）

平成 5 年九州大学工学部電子工学科卒業．平成 7 年同大学大学院情報工学専攻修士課程修了．同年九州大学工学部助手を経て，現在，岡山大学工学部准教授．博士（情報科学）．本会シニア会員．