Vol.2018-DPS-176 No.4
Vol.2018-EIP-81 No.4
2018/9/13

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

西 良太[1]　乃村 能成[1]

概要：以前と同様の作業を行う際，前回の作業内容の確認とファイルの再利用を目的に過去のファイルを利用することが多い．このとき，過去に行った同様の作業のフォルダがまとめて管理されていれば，過去のファイルの利用が容易となる．そこで，本稿では，ファイル種別とファイル更新履歴に着目したフォルダのクラスタリング手法を提案する．これにより，フォルダを同様の作業のグループとしてまとめることでファイル整理を支援する．

キーワード：クラスタリング，アクセス履歴，ファイル整理

1. はじめに

我々は，以前と同様の作業を行う際，前回の作業内容の確認とファイルの再利用を目的に過去のファイルを利用することが多い．たとえば，文書作成の際に過去の同様の文書の一部を変更して作成したり，過去に作成したプログラムの一部を再利用したりする場合がある．このとき，過去に行った同様の作業のファイルが何らかの形で管理されていれば，作業を行う際に過去のファイルを探す手間を削減できる．また，フォルダの中にはユーザが作業を行うときに意識する主要なフォルダであるワーキングディレクトリが存在する．ワーキングディレクトリの例を**図 1** に示す．図 1 の「第 1 回」フォルダと「第 2 回」フォルダは，それぞれ「TEX による資料作成」と「Markdown による資料作成」という作業を代表するフォルダであり，ワーキングディレクトリである．

過去のファイルを利用する際は，同様の作業のワーキングディレクトリを確認することで，目的のファイルの発見が容易になると考える．たとえば，図 1 において新たに「第 3 回」フォルダを作成し，「TEX による資料作成」という作業を行う際，TEX 特有の記述方法や図の挿入方法等を確認するために，過去の「TEX による資料作成」の作業で作成したファイルを確認・再利用したいという要求がある．このとき，同様の作業のワーキングディレクトリである「第 1 回」フォルダを確認することで目的のファイルの発見が容易になる．

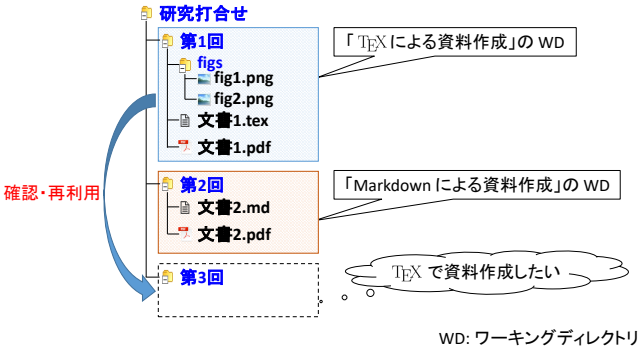

図 1　ワーキングディレクトリの例

しかし，計算機内のフォルダ分けは作業ではなく目的という観点で行われている場合がある．このとき，同様の作業のワーキングディレクトリであっても異なるフォルダ下に存在したり，異なる作業のワーキングディレクトリであっても同じフォルダ下に存在する場合がある．たとえば，図 1 の「第 1 回」フォルダと「第 2 回」フォルダは，「TEX による資料作成」と「Markdown による資料作成」という異なる作業のワーキングディレクトリである．しかし，どちらも研究打合せを目的に作成されており，「研究打合せ」フォルダの下で管理されている．この場合，「TEX による資料作成」の作業におけるファイルを確認・再利用する際には，「Markdown による資料作成」のワーキングディレクトリは参考にならないため，フォルダの中身を確認して「TEX による資料作成」のワーキングディレクトリを探す必要がある．このことから，同様の作業のワーキングディレクトリをまとめて管理したいという要求がある．

[1] 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University

本稿では，ファイル更新履歴の時系列に着目したフォルダのクラスタリング手法を提案する．同様の作業のフォルダでは，ファイル種別とファイル更新履歴が類似すると考える．たとえば，「TEX による文書作成」のフォルダでは「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，ユーザは TEX ファイルを編集し，コンパイル後の PDF を確認するため，「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．

アクセス履歴に着目したファイル検索を支援する研究として，ファイルアクセスログからファイル間関連度を算出する手法 [1] が提案されている．また，この関連度を用いたファイル検索手法 [2] が提案されている．これらは，キーワードによるファイル検索の支援を行っている．

本稿では，フォルダ内のファイル種別とファイル更新履歴の特徴に基づいて，計算機内のフォルダを作業に関してクラスタリングする．これにより，同様の作業のワーキングディレクトリをまとめて管理し，過去の作業内容の確認とファイルの再利用を支援できると考える．

以降では，まず，ファイル整理の観点と整理支援手法について述べる．次に，フォルダの作業を表す特徴について述べ，その特徴を用いたフォルダのクラスタリング手法について述べる．最後に，提案手法の評価について述べる．

2. 作業を観点とするファイル整理

2.1 ワーキングディレクトリとは

我々は，日々の作業で計算機内に作成・編集するファイルをフォルダ分けにより整理している．フォルダ分けによりファイルを整理する際は，作業を単位としてフォルダを作成することが多い．これらのフォルダの中には，作業を代表する主要なフォルダが存在する．フォルダ構造の例を**図 2** に示す．図 2 では，「資料」フォルダと「スライド」フォルダは，それぞれ「講義資料作成」や「スライド作成」という作業を代表している．このように，ユーザが作業を行うときに意識する主要なフォルダを**ワーキングディレクトリ**と呼ぶ．このワーキングディレクトリは，同じ作業を行った場合でもユーザの意識する作業の粒度によって異なる主観的な存在である．たとえば，図 2 のフォルダ構成において，ユーザ A とユーザ B が意識する作業はそれぞれ以下の通りであるとする．

(1) ユーザ A が意識する作業
(a) 講義準備
(2) ユーザ B が意識する作業
(a) 資料作成
(b) スライド作成

この場合，ユーザ A にとってのワーキングディレクトリは「講義」のみとなり，ユーザ B にとってのワーキングディレクトリは，「資料」と「スライド」となる．このように，同じフォルダ構成であっても，ワーキングディレクトリはユーザの意識する作業の粒度によって異なる場合がある．

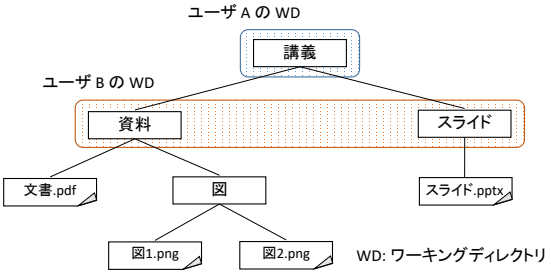

図 2 フォルダ構造の例

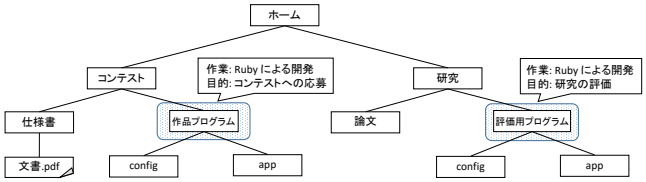

図 3 同様の作業のワーキングディレクトリが異なるフォルダ下に存在する場合

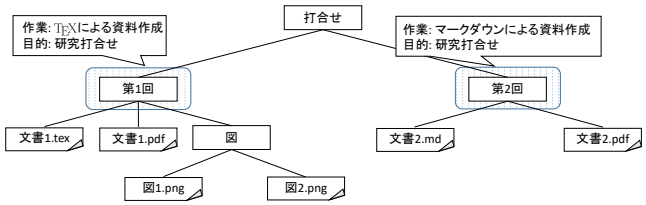

図 4 異なる作業のワーキングディレクトリが同じフォルダ下に存在する場合

2.2 ファイル整理の観点

計算機内のワーキングディレクトリの整理には以下の 2 つの観点が存在すると考える．

(観点 1) 目的を観点とする整理
同様の目的で作成されたワーキングディレクトリをまとめて管理する．

(観点 2) 作業を観点とする整理
同様の作業のワーキングディレクトリをまとめて管理する．

計算機内のフォルダは，(観点 1) により整理されている場合がある．たとえば，「Ruby による開発」という同様の作業のワーキングディレクトリであっても，「研究の評価」を目的とする場合と「コンテストへの応募」を目的とする場合では，異なるフォルダの下で管理される．

しかし，以前と同様の作業を行う際は，作業内容の確認とファイルの再利用を目的に過去のファイルを利用することが多く，同様の作業のワーキングディレクトリを確認したいという要求がある．このため，過去のファイルの再利

用には（観点 2）による整理が有用である．

（観点 1）による整理においては，ワーキングディレクトリの上層のフォルダは，ワーキングディレクトリで行われる作業の目的に応じて異なる．このことから，目的を観点とする（観点 1）による整理では，以下の 2 つの場合が発生する．

(場合 1) 同様の作業のワーキングディレクトリが異なるフォルダ下に存在

例を**図 3** に示す．図 3 では，2 つの「Ruby による開発」という同様の作業のワーキングディレクトリが「コンテストへの応募」と「研究の評価」という異なる目的で作成されており，異なるフォルダの下で管理されている．このため，計算機内から「Ruby による開発」のワーキングディレクトリを探す際，複数のフォルダを確認する必要がある．

(場合 2) 異なる作業のワーキングディレクトリが同じフォルダ下に存在

例を**図 4** に示す．図 4 では，2 つの「TEX による資料作成」と「Markdown による資料作成」という異なる作業のワーキングディレクトリが「研究打合せ」という同様の目的で作成されており，同じフォルダの下で管理されている．このとき，次回の研究打合せの資料を TEX によって記述する場合，図 4 において「TEX による資料作成」という同様の作業が行われている「第 1 回」フォルダを参考にする．しかし，「打合せ」フォルダには「Markdown による資料作成」のフォルダも存在する．このため，ユーザが「TEX による文書作成」のフォルダを発見するには 1 つ 1 つフォルダの中身を確認する必要がある．

また，（場合 1）と（場合 2）は同時に起こりうる．このため，過去の同様の作業の振り返りやファイルの再利用を行う際には，計算機内に存在する同様の作業のワーキングディレクトリを 1 つ 1 つ探す必要があり手間となる．この問題への対処として，計算機内のワーキングディレクトリを作業に関して自動分類したり，分類先の作業をサジェストするなどの整理を支援する仕組みが必要であると考える．次節にて，計算機内に存在するワーキングディレクトリを作業に関して整理するための支援について述べる．

2.3 作業を観点とするファイル整理の支援

2.2 節で述べたように，計算機内のフォルダ分けが，同様の作業ではなく同様の目的という観点で行われている場合，同様の作業のワーキングディレクトリを探す際に手間となる．そこで，計算機内の同様の作業のワーキングディレクトリをまとめて管理するための支援について検討する．

本稿では，同様の作業のフォルダのファイル種別とファイル更新履歴の類似性に着目し，計算機内のフォルダを作業に関してクラスタリングすることで，計算機内の同様の作業のワーキングディレクトリをまとめて管理することを支援する．

ファイル整理支援の研究としては，ワーキングディレクトリを推定し，一覧を表示することで，ワーキングディレクトリをユーザに管理させる整理手法 [3] が提案されている．文献 [3] では，一覧表示されたワーキングディレクトリを手動で作業に関して分類する．本稿では，計算機内のフォルダを作業に関してクラスタリングすることで，同様の作業のワーキングディレクトリをまとめて管理することの支援を行う．

また，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利用して仮想ディレクトリを生成する手法 [4] やユーザのファイル操作履歴から関連性を求めることでファイル検索を支援する手法 [5] が存在する．これらの研究では，計算機内のある作業やキーワードに関連するファイルを提示することで支援を行っている．本稿では，計算機内のファイルがフォルダ分けによりある程度整理されていることに着目し，同様の作業のフォルダを集約することで，過去の同様の作業内容の確認とファイルの再利用を支援する．

フォルダを作業に関してクラスタリングするには，フォルダの作業を表現する特徴量が必要となる．このため，次章で，フォルダの作業を表現する特徴ベクトルの作成方法について述べる．

3. フォルダの作業を表現する特徴ベクトル

3.1 フォルダの作業を表現する特徴

本章ではフォルダで行われた作業を表現する特徴について述べる．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挙げられる．本稿では，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) ファイルアクセス履歴

（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．

（特徴 2）のファイルアクセス履歴とは，ファイルの作成，更新，および削除といったユーザによるファイル操作の履歴である．本稿では，ファイルアクセス履歴の内，ファイル更新履歴に着目する．同様の作業では，ファイル更新履歴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

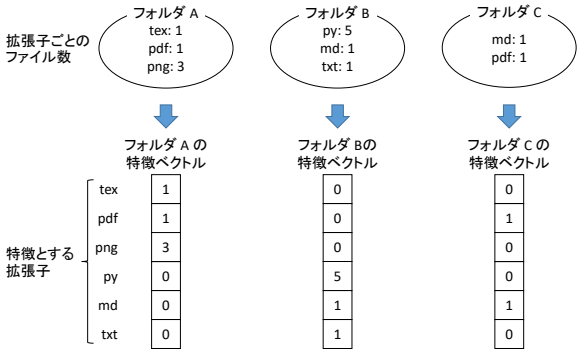

図 5 （方法 1）における特徴ベクトルの作成

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．

次節にて，（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 特徴ベクトルの作成

3.2.1 方針

本稿では，以下の 2 つの方法により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 1) ファイル種別のみを用いる方法

(方法 2) ファイル種別とファイル更新履歴を用いる方法

（方法 1）では，フォルダ内のファイル種別のみを用いて特徴ベクトルを作成する．ファイル種別は，ファイルの拡張子により識別する．この方法では，拡張子ごとのファイル数を数え上げることでベクトルを作成する．

また，（方法 2）では，ファイル種別とファイル更新履歴を用いて特徴ベクトルを作成する．ファイル拡張子ごとのファイル更新履歴の順序関係を用いて特徴ベクトルを作成する．具体的には，時系列順に並んだファイル更新履歴において，隣接する更新履歴のファイル拡張子の組合せを数え上げることでベクトルを作成する．

次項以降で，（方法 1）と（方法 2）による特徴ベクトルの作成方法についてより詳細に述べる．

3.2.2 ファイル種別のみを用いる方法

ファイル種別のみを用いて，フォルダごとの特徴ベクトルを作成する方法を**図 5** に示して以下に説明する．この方法では，対象のフォルダ内のファイルについて拡張子ごとにファイル数を求め，ベクトルの要素とする．

たとえば，図 5 のフォルダ A では「tex」拡張子を持つファイルが 1 個，「pdf」拡張子を持つファイルが 1 個，および「png」拡張子を持つファイルが 3 個存在しており，それぞれをベクトルの要素としている．

また，フォルダ全体のファイル数による差を軽減するため，ベクトルの L^2 ノルム（長さ）が 1 になるように正規化する．

3.2.3 ファイル種別とファイル更新履歴を用いる方法

ファイル種別とファイル更新履歴を用いて，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

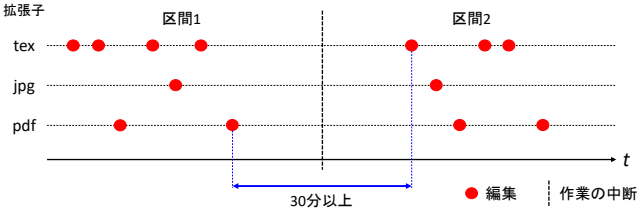

図 6 ファイル更新履歴の分割

表 1 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

	tex-tex	tex-pdf	pdf-tex	tex-jpg	jpg-tex	jpg-pdf
区間 1	1	2	1	1	1	0
区間 2	1	1	1	1	0	1
合計	2	3	2	2	1	1

表 1 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から更新履歴のみを抽出

図 6 では，横軸に時刻を取り，拡張子ごとのファイル更新を丸印で示している．

(手順 2) ファイル更新履歴を作業区間に分割

（手順 1）で作成した履歴について，履歴間に一定時間以上の空きがある場合，作業の中断とみなしてファイル更新履歴を分割する．図 6 では 30 分以上空きがある場合を作業の中断とみなして 2 つの区間に分割している．

(手順 3) 隣接するファイル更新履歴の集計

（手順 2）により得られた区間に対して，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 1 となる．そして，区間 1 と区間 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更新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．

4. フォルダのクラスタリング手法

4.1 方針

本章では，フォルダを 3.1 節で述べた特徴を用いてクラスタリングすることで，作業ごとに分類する手法について述べる．

本稿では，計算機内のフォルダをクラスタリングする際，いくつのクラスタに分類されるかは分かっていないことを想定する．そこで，事前にクラスタ数を指定する必要がない階層的クラスタリング手法を用いる．階層的クラスタリングは，すべての対象データを，データ 1 つからなるクラスタとみなし，すべてのノードが 1 つのクラスタに併合されるまで，クラスタ間距離が最も小さいクラスタ同士の併合を繰り返すことでクラスタリングを行う．次節で，階層的クラスタリング手法を用いた分類方法について述べる．

4.2 階層的クラスタリングによる分類

階層的クラスタリングでは，クラスタの併合の様子をデ

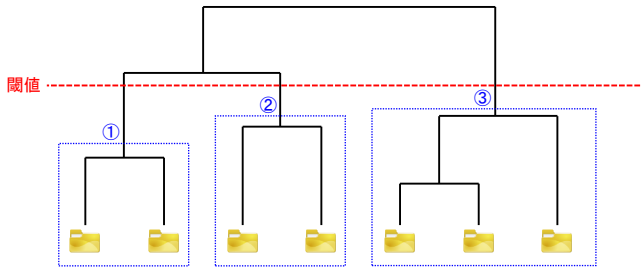

図 7 階層的クラスタリングによって得られるデンドログラム

ンドログラムと呼ばれる樹形図で表せる．**図 7** に，7 つのフォルダの階層的クラスタリングにより得られるデンドログラムの例を示す．また，デンドログラムの高さはクラスタ間距離を表しており，これを閾値で分割することでクラスタ数を決定することができる．図 7 では，閾値によりデンドログラムを分割することで 3 つのクラスタを得ている．この閾値を変化させることで，クラスタ数を調整できる．

本稿では，3.2 節で述べた方法により作成した特徴ベクトル V_A と V_B の距離をコサイン類似度を用いて，式 (1) により定義する．

$$\mathrm{dist}(V_A, V_B) = 1 - \frac{V_A \cdot V_B}{||V_A||_2||V_B||_2} \quad (1)$$

また，階層的クラスタリングにおいて，クラスタ間距離を群平均法により算出する．群平均法とは，2 つのクラスタ同士で, すべての組合せのサンプル間距離の平均をクラスタ間の距離とする方法である．

4.3 クラスタリング対象の選出

本稿では，計算機内のフォルダをクラスタリングすることで同様の作業のワーキングディレクトリの発見を容易にする．しかし，計算機内のすべてのフォルダをクラスタリング対象にすると，ワーキングディレクトリではないフォルダにより構成されるクラスタが大量にできたり，計算時間が増加したりする問題が発生する．このため，計算機内のフォルダからクラスタリングの対象とするフォルダを選出する仕組みが必要である．

クラスタリング対象としては，2.1 節で述べたワーキングディレクトリのみを対象にすることが考えられる．ワーキングディレクトリを推定する方法としては，ファイル更新履歴を用いたワーキングディレクトリ推定手法 [3] が提案されているため，これを利用してクラスタリング対象を選出することが考えられる．

本稿では，著者の 1 人が手動で選択したワーキングディレクトリを対象にクラスタリングを行い，クラスタリング結果の評価を行う．

5. 評価

5.1 評価観点

フォルダのクラスタリング結果を以下の 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

この観点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) 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

この観点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．ただし，1 つのクラスタを確認するだけでより多くの同様の作業のフォルダが発見できるかどうかは保証しない．つまり，すべてのクラスタが 1 つのフォルダにみから構成される場合でも，この観点に対する評価は高くなる．

同様の作業のフォルダを探すことを目的にクラスタリング結果を利用する場合，複数のクラスタに散在している同様の作業のフォルダを探すよりも，同様の作業のフォルダが多く存在する 1 つのクラスタから異なる作業のフォルダを除去する方が手間が小さい．このため，（観点 1）に対する評価が高いクラスタリング結果の方が有用といえる．しかし，（観点 1）ではクラスタ内に異なる作業のフォルダどれだけ多く混在していても，ある作業のフォルダが 1 つのクラスタに多く分類されていれば評価が高くなる．このため，（観点 2）によりクラスタ内に異なる作業のフォルダが存在しすぎないことを保証する必要がある．

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．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．

本評価では，（観点 1）を評価する尺度として，InversePurity，（観点 2）を評価する尺度として，Purity を用いる．

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 (2) と式 (3) 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，

表 2 ホームディレクトリ以下のフォルダ数，ファイル数，およびファイルアクセス履歴数

項目	数
フォルダ数	30,174
ファイル数	196,410
ファイルアクセス履歴数	5,069,648

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する．

$$\mathrm{InversePurity} = \sum_{j=1}^{|A|} \frac{|A_j|}{N} \max_i(\mathrm{Recall}(C_i, A_j)) \qquad (2)$$

$$\mathrm{Purity} = \sum_{i=1}^{|C|} \frac{|C_i|}{N} \max_j(\mathrm{Precision}(C_i, A_j)) \qquad (3)$$

式 (2) 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 $\mathrm{Recall}(C_i, A_j)$ は，式 (4) によって定義される．

$$\mathrm{Recall}(C_i, A_j) = \frac{|C_i \cap A_j|}{|A_j|} \qquad (4)$$

式 (3) 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 $\mathrm{Precision}(C_i, A_j)$ は，以下の式 (5) によって定義される．

$$\mathrm{Precision}(C_i, A_j) = \frac{|C_i \cap A_j|}{|C_i|} \qquad (5)$$

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 (6) によって定義される．

$$F = \frac{2}{\mathrm{InversePurity}^{-1} + \mathrm{Purity}^{-1}} \qquad (6)$$

5.2 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 人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．計算機のホームディレクトリ以下のフォルダ数，ファイル数，およびファイルアクセス履歴数を**表 2** に示す．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 3** に，評価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 3 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．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．

表 3 評価に用いる実験データ

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

5.3 評価実験

5.3.1 評価方法

本評価では，3.2 節で述べた特徴ベクトルの作成方法として，ファイル種別のみを用いる（方法 1）とファイル種別とファイル更新履歴を用いる（方法 2）のそれぞれで実験を行いクラスタリング結果を比較する．ここで，階層的クラスタリングによって得られるデンドログラムを分割するための閾値には，生成されたクラスタの F 値が最大となる値を選択する．これにより得られたクラスタリング結果について，InversePurity，Purity，および F 値を評価し比較する．評価においては，表 3 に示した作業を正解の分類とする．

また，表 3 の 23 個のフォルダ内のファイルには 197 種類の拡張子が存在し，特徴ベクトルの次元数は，（方法 1）で 197 次元，（方法 2）で 3,835 次元であった．この特徴ベクトルを主成分分析により次元圧縮して利用する．圧縮後の次元数は（方法 1）では 10 次元，（方法 2）では 8 次元とした．

5.3.2 評価結果

（方法 1）の特徴ベクトルを用いて分類した場合と（方法 2）の特徴ベクトルを用いて分類した場合の InversePurity，Purity，および F 値の比較を**図 8** に示す．

（方法 1）による分類結果の評価値は，InversePurity は 0.7826 であり，Purity は 0.8261 であった．（方法 2）と比較すると，InversePurity が低く，Purity が高い．このことから，（方法 2）の分類結果よりも同様の作業のフォルダが複数のクラスタに分散して分類されているが，クラスタの純度は高いことが分かる．

（方法 2）による分類結果の評価値は，InversePurity は 0.9565 であり，Purity は 0.7391 であった．（方法 1）と比較すると，InversePurity が高く，Purity が低い．このことから，（方法 1）の分類結果よりも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が，クラスタの純度は低いことが分かる．

また，F 値は（方法 1）では 0.8038，（方法 2）では 0.8339

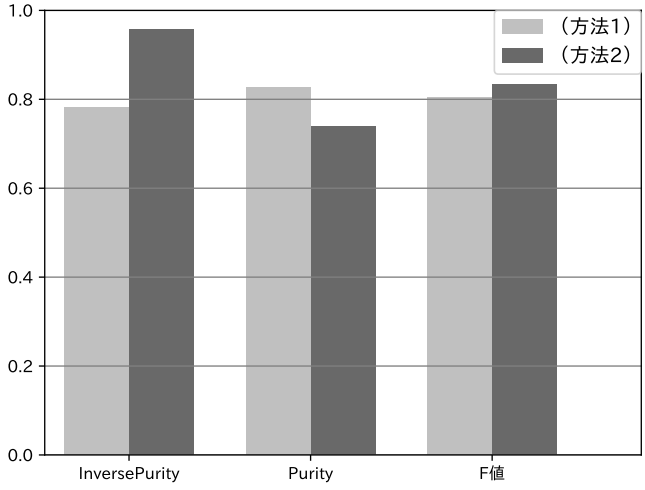

図 8 評価結果の比較

であった．F 値の比較では（方法 2）の方が若干良好な結果となった．（方法 2）では，（方法 1）より Purity が少し小さくなったが，InversePurity はかなり向上している．このため，同様の作業のフォルダを探すことを目的にクラスタリング結果を利用する場合（方法 2）の特徴ベクトルを使用する方が有効と考えられる．また，（方法 2）の方が F 値が良好なことから Purity もある程度保たれていることが分かる．次項にて，（方法 1）と（方法 2）の具体的な分類結果を分析し，実際に（方法 2）の結果の方が現実的な利用において有用かどうかを確認する．

5.3.3 分析

（方法 1）と（方法 2）による具体的な分類結果について分析を行う．

まず，（方法 1）により特徴ベクトルを作成した場合の分類の結果を**表 4** に示す．表 4 では，表 3 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．

（方法 1）による分類の結果，表 3 のワーキングディレクトリから 11 個のクラスタを得た．表 4 から，同様の作業のフォルダが複数のクラスタに散在していることが分かる．具体的には，作業 1，作業 5，および作業 7 のフォルダが最大 3 個のクラスタに散在している．これは，図 8 における（方法 1）の InversePurity が低いことと対応している．このため，同様の作業のフォルダをすべて発見するには，複数のクラスタを確認しなければならない可能性が高い．

次に，（方法 2）により特徴ベクトルを作成した場合の分類結果を**表 5** に示す．

分類の結果，表 3 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 5 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．これは，図 8 における（方法 2）の InversePurity が高いことと対応している．

（方法 1）と（方法 2）の分類結果を比較した結果，（方法 1）の方が生成されたクラスタが単一の作業のフォルダで構成されている可能性が高く，（方法 2）の方が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可能性が高い傾向にあった．これは，図 8 の評価値と対応した結果となっている．同様の作業のフォルダを探すことを目的にクラスタリング結果を利用する場合，（方法 1）の分類結果のように複数のクラスタに散在したワーキングディレクトリを探すよりも，（方法 2）のように 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方が有用である．このため，InversePurity が高く，Purity がある程度保たれている（方法 2）のクラスタリング結果の方が有用といえる．

6. おわりに

本稿では，計算機内のフォルダを作業に関してクラスタリングする手法について述べ，フォルダ内のファイル種別に着目した特徴ベクトルとフォルダのファイル更新の時系列に着目し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った．これら 2 つの方法によるクラスタリングの結果を比較し，評価を行った．

評価では，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．同様の作業のフォルダを探すことを目的にクラスタリング結果を利用する場合，複数のクラスタに散在している同様の作業のフォルダを探すよりも，同様の作業のフォルダが多く存在する 1 つのクラスタから異なる作業のフォルダを除去する方が手間が小さい．このため，現実の利用を考えた場合 InversePurity の高いクラスタリング結果の方が有用であるといえる．しかし，クラスタ内に異なる作業のフォルダが存在しすぎないことを保証するために Purity が必要である．そこで，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，ファイル種別のみを用いた分類では Purity が高くなり，ファイル種別とファイル更新履歴を用いた分類では InversePurity が高くなった．また，F 値はファイル更新履歴を用いた分類の方が少し良好だった．このことから，InversePurity が高く，ある程度 Purity が保たれているファイル更新履歴を用いた分類の方が有用といえる．

今後の課題としては，クラスタリング対象のフォルダを計算機内から自動で選出する手法の検討とクラスタリング精度の向上が挙げられる．

表 4 （方法 1）ファイル種別を用いたクラスタリング結果

	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	-	-	-	-	50.0%(2)	-	33.3%(1)	-	-	-
クラスタ 2	-	-	-	-	25.0%(1)	100%(3)	33.3%(1)	-	-	-
クラスタ 3	-	-	-	-	-	-	-	100%(1)	-	-
クラスタ 4	-	-	-	-	25.0%(1)	-	-	-	-	-
クラスタ 5	-	-	-	-	-	-	-	-	100%(3)	-
クラスタ 6	-	-	-	-	-	-	-	-	-	100%(3)
クラスタ 7	-	-	-	100%(1)	-	-	-	-	-	-
クラスタ 8	-	100%(2)	-	-	-	-	-	-	-	-
クラスタ 9	-	-	-	-	-	-	33.3%(1)	-	-	-
クラスタ 10	50.0%(1)	-	100%(1)	-	-	-	-	-	-	-
クラスタ 11	50.0%(1)	-	-	-	-	-	-	-	-	-

表 5 （方法 2）ファイル種別とファイル更新履歴を用いたクラスタリング結果

	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)	-	-	-	-	-	-

参考文献

[1] 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).
[2] 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).
[3] 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).
[4] 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).
[5] 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.), Berlin, Heidelberg, Springer Berlin Heidelberg, pp. 364–378 (2011).
[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).