同様の作業における
フォルダ構造の類似性に着目した
ファイル整理支援手法の提案

平成30年2月16日

岡山大学 工学部 情報系学科
西 良太

研究背景

同様の作業を行う際は過去のファイルを利用

➡ 作業と主要なフォルダを関連付けて整理したい

作業にはそれを代表するフォルダが存在

ワーキングディレクトリ (WD)

＜ファイル整理の現状＞

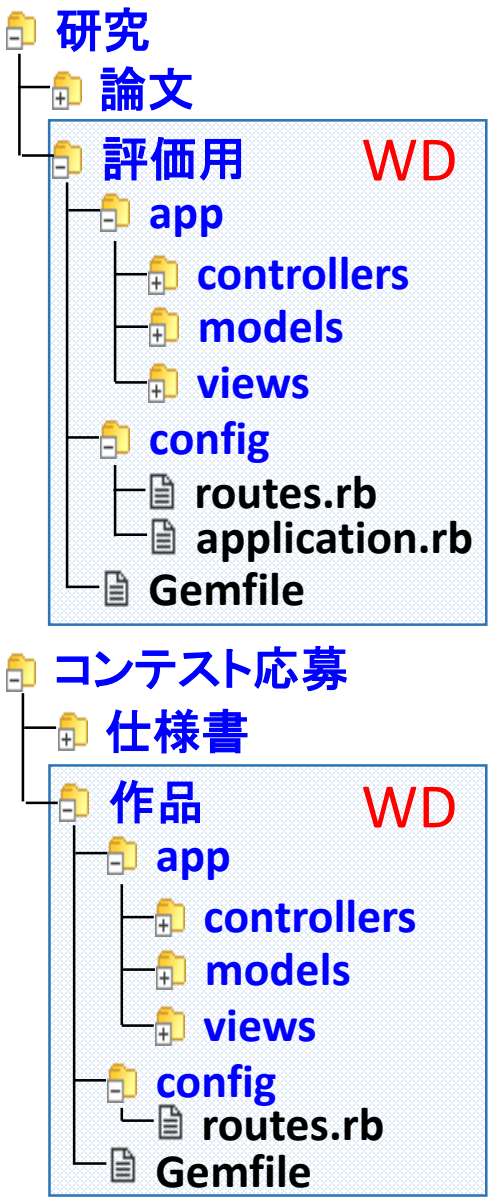
フォルダは，同様の作業ではなく，
同様の目的という観点で整理されている
∴ 同様の作業の WD が計算機上に散在

同様の作業におけるフォルダ構造の類似性に着目

➡ フォルダ構造の特徴を用いてクラスタリング

（特徴1）フォルダ内のファイル種別

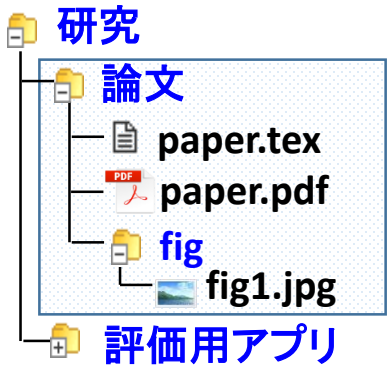
（特徴2）フォルダの階層構造の形状



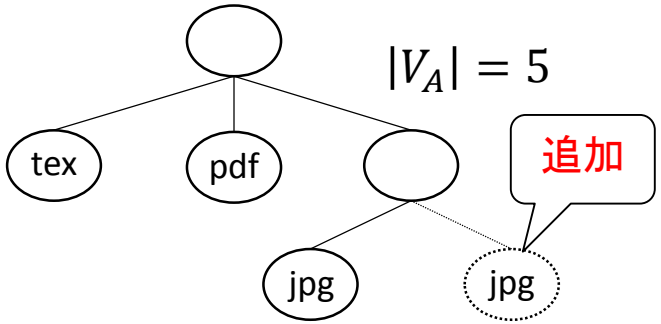
フォルダ間距離の算出手法

クラスタリングにおいて，フォルダ間の距離を評価する尺度が必要

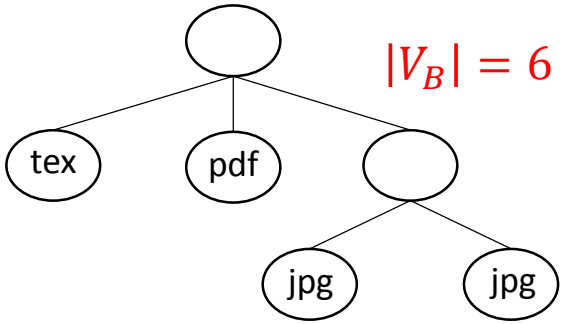
フォルダ構造の特徴を木で表現し，木構造編集距離を尺度とする



「論文」の木 T_A



「仕様書」の木 T_B



$\mathrm{TED}(T_A, T_B) = 1$

$\therefore \mathrm{dist} = \frac{1}{6-1} = 0.2$

クラスタリング結果の評価

＜評価観点＞

(1) 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

Purity: クラスタの一貫性を評価

(2) 同様の作業のフォルダがどの程度同じクラスタに分類されたか

InversePurity: 正解データの集中性を評価

総合的な評価のため，2つの指標の調和平均である *F* 値を使用

評価環境

実験データ: 計算機内の41個の WD であるフォルダ

以下の6種類の作業への分類を正解として評価を行う

	作業内容	フォルダ数
作業1	シェルスクリプト作成	1
作業2	Python スクリプト作成	4
作業3	Rails アプリケーション開発	5
作業4	Node.js による開発	1
作業5	TEX による文書作成	19
作業6	マークダウンによる文書作成	11
合計		41

評価方法

以下の3つの方法による分類を比較

＜提案手法を用いない場合＞

(分類1) 上層のフォルダによる分類

各 WD がどのフォルダの下に属しているかにより分類

＜提案手法を用いる場合＞

(分類2) Ward 法を用いたクラスタリングによる分類

(分類3) 群平均法を用いたクラスタリングによる分類

評価結果

分類手法	Purity	InversePurity	F 値	実行時間
(分類1) 上層のフォルダによる分類	0.6341	0.5854	0.6088	
(分類2) 提案手法（Ward 法）	0.9512	0.9024	0.9262	663.35 sec
(分類3) 提案手法（群平均法）	1.0000	0.9512	0.9750	667.87 sec

＜ (分類1)と提案手法による分類の比較＞

(分類2)では，*F* 値が 52.1 % 向上

(分類3)では，*F* 値が 60.2 % 向上

∴ 提案手法は，作業と WD の関連付けに有用

分類1 上層のフォルダによる分類

( )内はフォルダ数

	作業1	作業2	作業3	作業4	作業5	作業6
フォルダ1	100%(1)	100%(4)	100%(5)	100%(1)		
フォルダ2					36.8%(7)	27.3%(3)
フォルダ3					31.6%(6)	54.5%(6)
フォルダ4					26.3%(5)	
フォルダ5					5.3%(1)	
フォルダ6						9.1%(1)
フォルダ7						9.1%(1)

(1) 異なる種類の作業の WD が1つのフォルダに混在

(2) 1種類の作業の WD が複数のフォルダに散在

∵ 目的を観点としてフォルダ分けされている

⇒ 作業という観点で見ると繁雑

分類2 提案手法（Ward 法）による分類

( )内はフォルダ数

	作業1	作業2	作業3	作業4	作業5	作業6
クラスタ1					78.9%(15)	
クラスタ2						100%(11)
クラスタ3	100%(1)		100%(5)	100%(1)		
クラスタ4					21.1%(4)	
クラスタ5		100%(4)				

(1) 異なる種類の作業の WD が1つのクラスタに混在

(2) 作業5「TEX による文書作成」の WD がクラスタ1と4に散在

∵ クラスタ4では，図をまとめたサブフォルダが存在

分類3 提案手法(群平均法)による分類

( )内はフォルダ数

	作業1	作業2	作業3	作業4	作業5	作業6
クラスタ1			60.0%(3)			
クラスタ2			20.0%(1)			
クラスタ3			20.0%(1)			
クラスタ4					100%(19)	
クラスタ5						100%(11)
クラスタ6		100%(4)				
クラスタ7	100%(1)					
クラスタ8				100%(1)		

(1) すべてのクラスタが1種類の作業の WD で構成

(2) 1種類の作業の WD が複数のクラスタに散在

まとめ

＜実績＞

(1) フォルダ構造の特徴の検討

　(A) フォルダ内のファイル種別

　(B) フォルダの階層構造の形状

(2) フォルダの作業に関するクラスタリング手法の検討

　(A) 木構造編集距離によるフォルダ間距離の算出

(3) 提案手法の評価

　(A) 分類結果において，提案手法の有用性を示した

　(B) 計算機内の全フォルダを対象とすると計算量大

＜今後の課題＞

(1) 計算量の改善（現在は、41個の WD の分類で約11分）

(2) ファイルへのアクセス履歴の利用

(3) クラスタリング結果をユーザに提示するインタフェースの検討

予備スライド

評価指標

(指標1) Purity

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

適合率: $\text{Precision}(C_i, A_j) = \frac{|C_i \cap A_j|}{|C_i|}$

(指標2) InversePurity

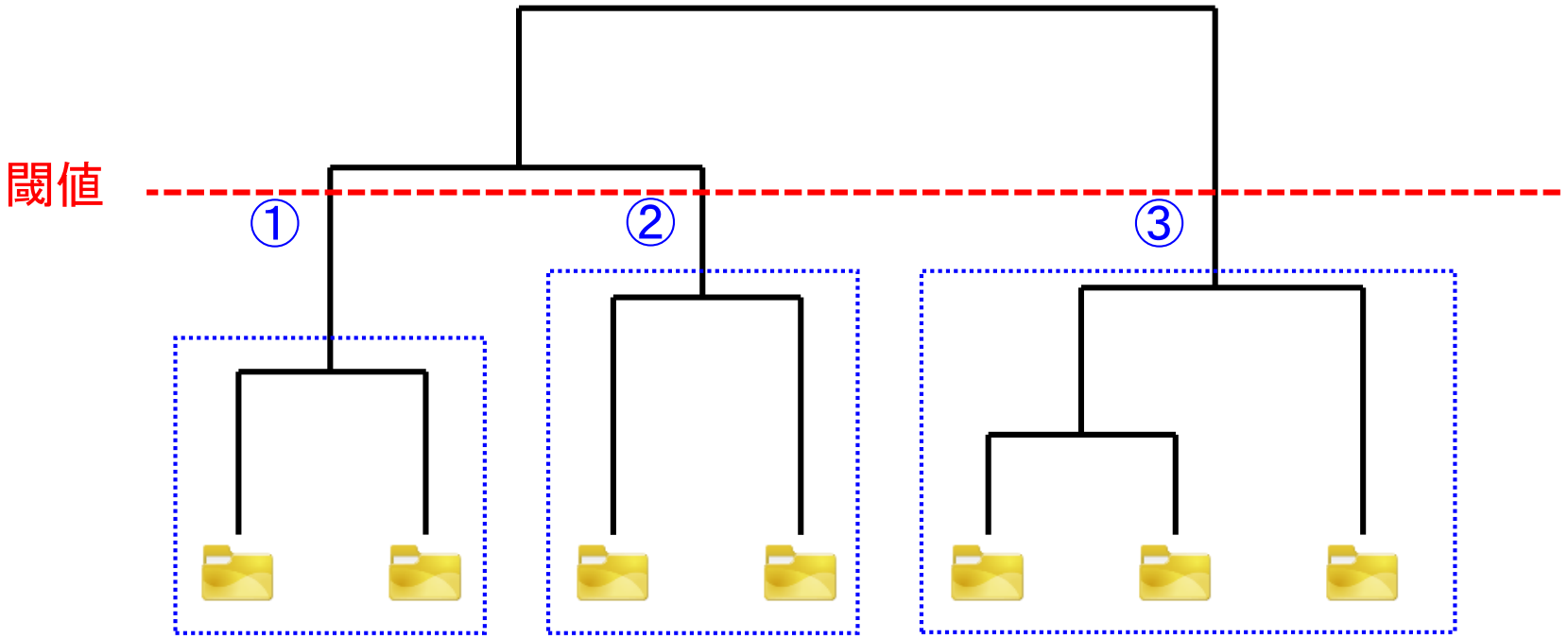
$$\text{InversePurity} = \sum_{j=1}^{|A|} \frac{|A_j|}{N} \max_i(\text{Recall}(C_i, A_j))$$

再現率: $\text{Recall}(C_i, A_j) = \frac{|C_i \cap A_j|}{|A_j|}$

フォルダのクラスタリング手法

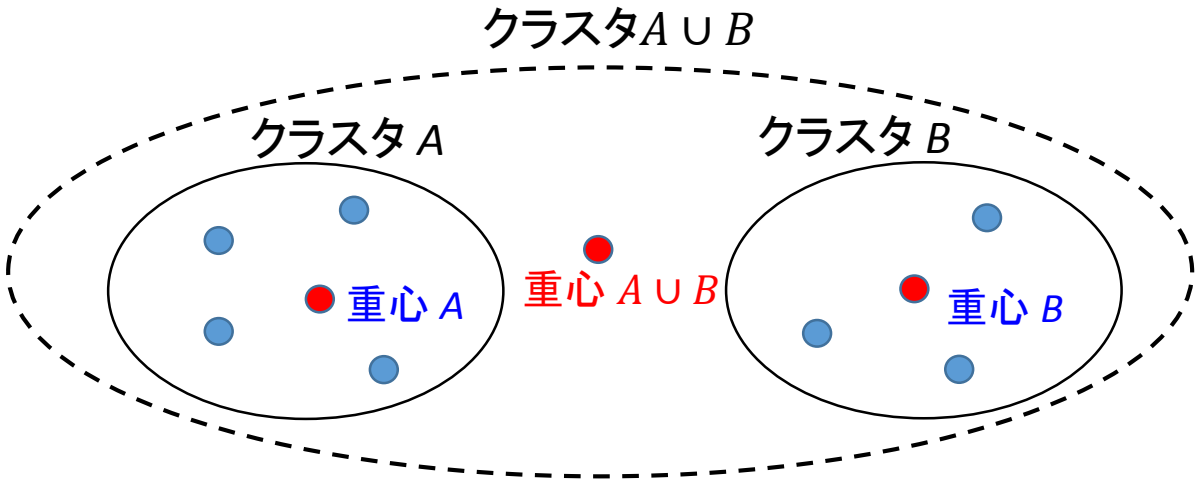
あらかじめ何種類の作業に分類すべきかは不明

➡ クラスタ数を与える必要が無い階層的クラスタリングを用いる



クラスタ間距離

(1) Ward 法

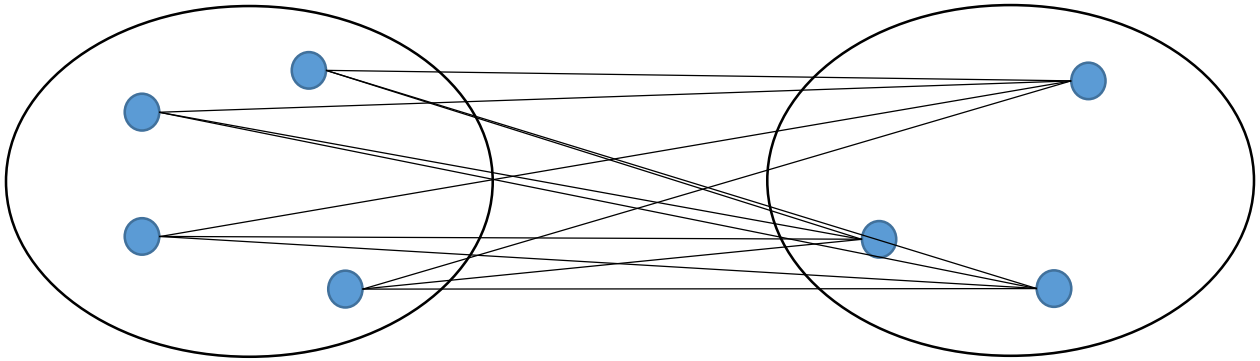


$$\Delta = L(A \cup B) - L(A) - L(B)$$

$L(X)$: クラスタ X の各サンプルと重心との距離の二乗和

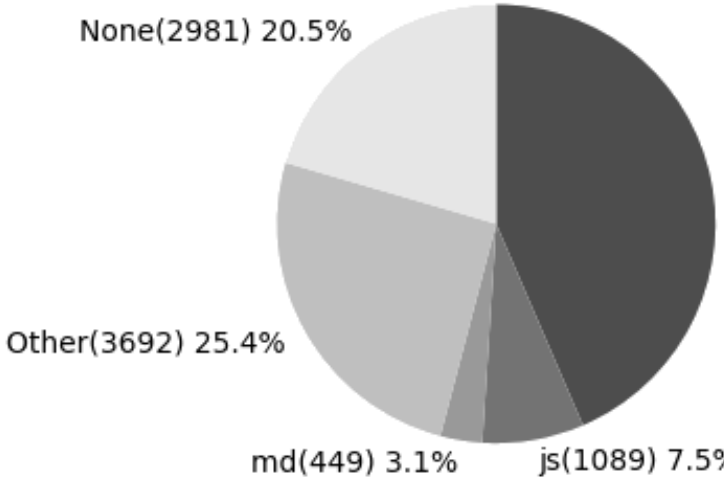
(2) 群平均法

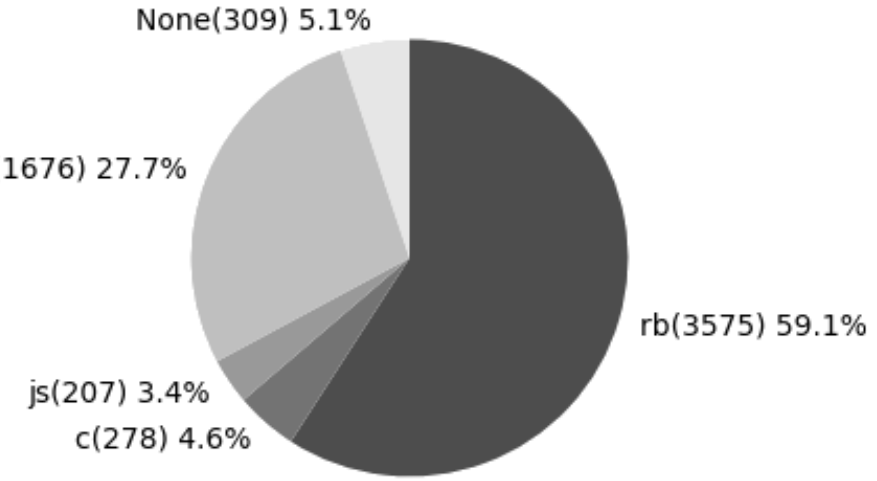
全ての組合せのサンプル間距離の平均をクラスタ間距離とする



フォルダ内のファイル種別の調査

「Rails アプリケーション開発」の WD 内の拡張子





「rb」と「js」が多く存在するという特徴が類似

∴ 同様の作業では，フォルダ内に存在する拡張子が類似

フォルダの階層構造の形状の調査

$$\mathrm{dist}(\mathrm{A},\mathrm{B}) = \frac{\mathrm{TED}(T_A, T_B)}{\max(|V_A|, |V_B|) - 1}$$

	Python1	Python2	Rails1	Rails2	TeX1	TeX2
Python1		**0.1935**	0.9548	0.8837	0.6774	0.6129
Python2	**0.1935**		0.9620	0.8977	0.6296	0.5556
Rails1	0.9548	0.9620		**0.6781**	0.9892	0.9855
Rails2	0.8837	0.8977	**0.6781**		0.9721	0.9628
TeX1	0.6774	0.6296	0.9892	0.9721		**0.250**
TeX2	0.6129	0.5556	0.9855	0.9628	**0.250**	

∴ 階層構造の形状は，ユーザごとのフォルダ構造の特徴を表す

実験に用いた計算機の環境

項目名	環境
OS	Ubuntu 16.04 LTS
CPU	Intel(R) Core(TM) i7-6500U @ 2.50GHz
メモリ	8GB

実行時間

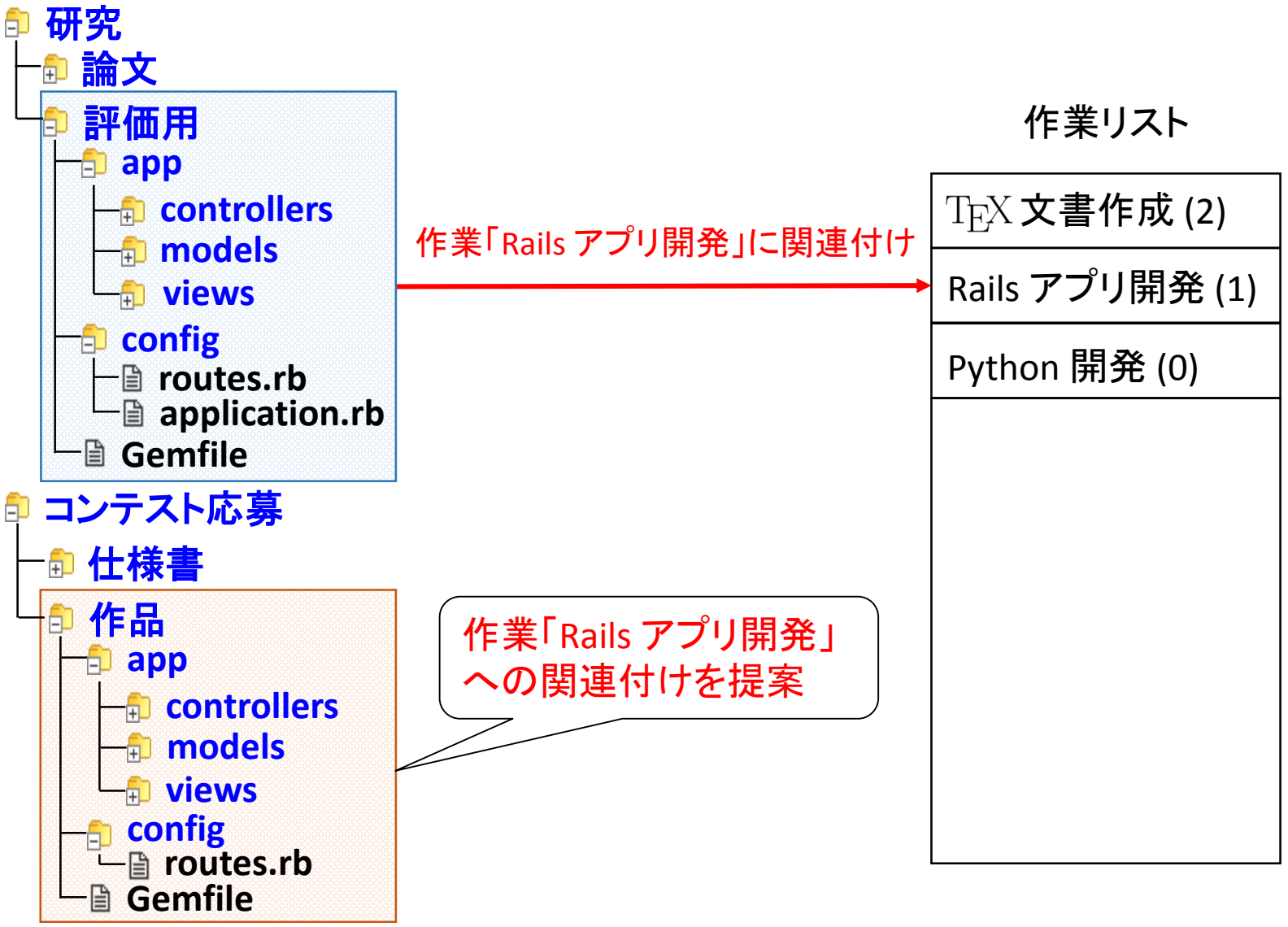
分類手法	Purity	InversePurity	F 値	実行時間
(分類1) 上層のフォルダによる分類	0.6341	0.5854	0.6088	
(分類2) 提案手法（Ward 法）	0.9512	0.9024	0.9262	663.35 sec
(分類3) 提案手法（群平均法）	1.0000	0.9512	0.9750	667.87 sec

41個の WD の分類に要した時間は約11分

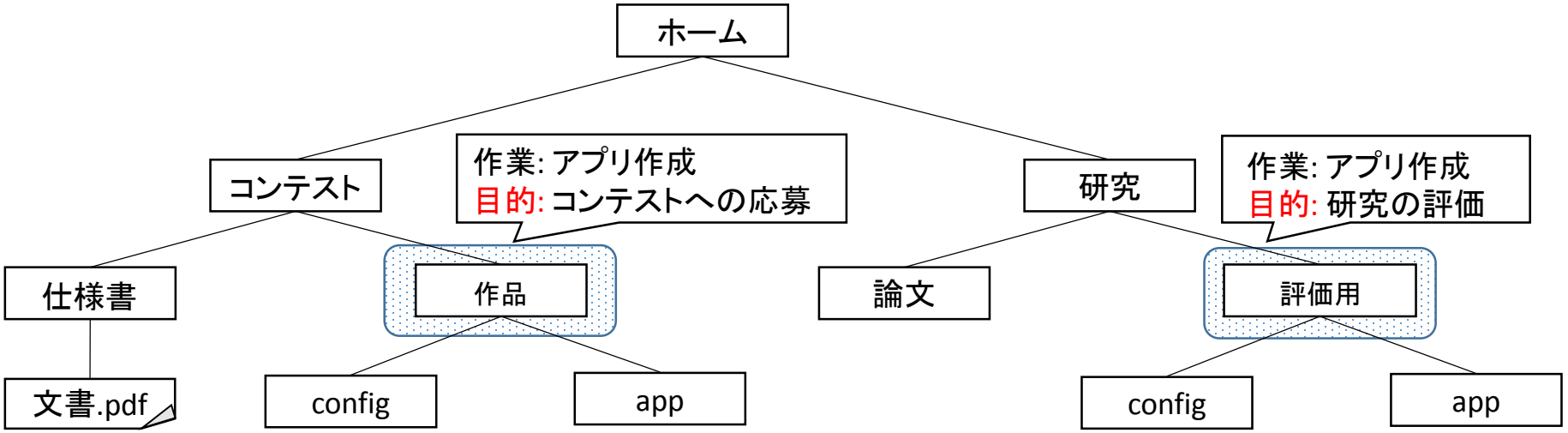
∴ 計算機内の全フォルダを対象とすると計算量大

➡ 計算量を改善する必要あり

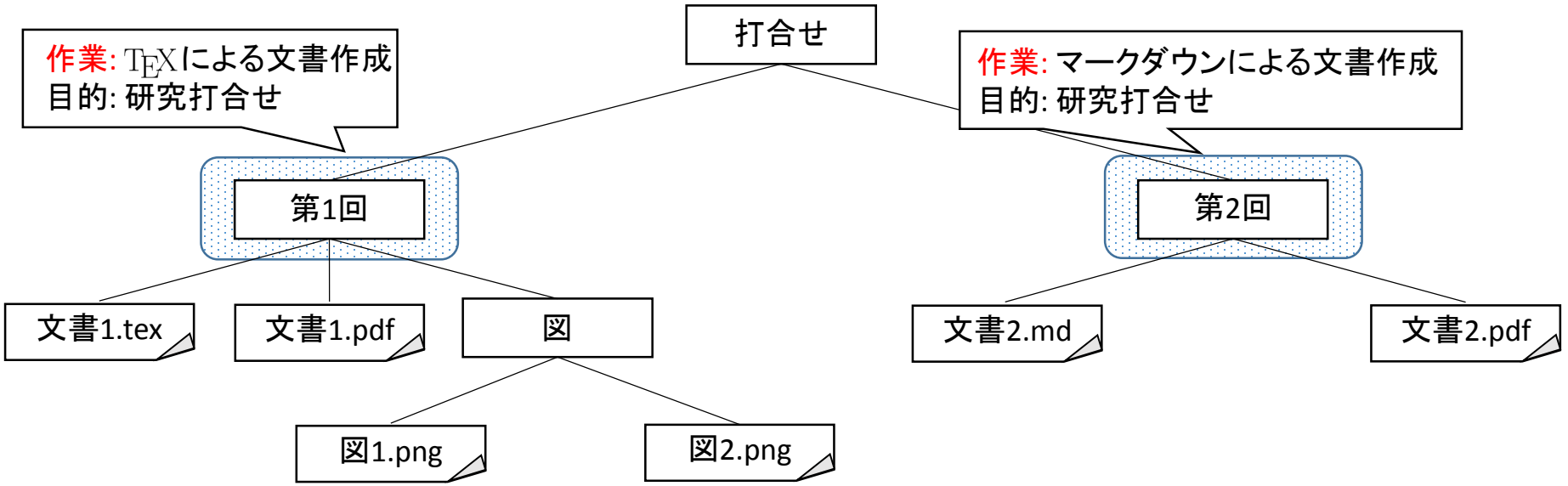
ファイル整理支援の様子



同様の作業の WD が散在



異なる作業の WD が混在



木構造編集距離の計算量

Zhang-shasha のアルゴリズム

時間計算量: $O(|T_1||T_2| \min(L_1, D_1) \min(L_2, D_2))$

$|T_n|$: 木のノード数

L_n: 木のはノードの数

D_n: 木の深さ