

大規模組織における適切な人材配置のための 開発者検索手法

オープンソースソフトウェア工学研究室

学籍番号：60191045

福井 克法

背景

- ソフトウェア開発は大規模化しているにも関わらず短納期で開発することが求められている
- ソフトウェア開発を円滑に行うためには、開発者の特性を理解して過去の実績や関わった開発を把握しておくことが望ましい

ソフトウェア開発現場での問題点

- 大規模な開発現場では、多数の開発者が在籍している
- 今までに関わったプロジェクトや開発をすべて把握することは難しい
 - ▶ プロジェクトに適任の開発者を割り当てるのが難しくなる
 - ▶ 企業が案件を提案する際に、開発者の知識や経験を生かせずに案件を逃してしまう

モジュール単位での開発者検索

.....

- ソースコードの変更履歴に着目したモジュール単位での適任開発者を推薦する手法が提案されている[1][2]
 - ▶ 特定のモジュールに精通した開発者を検索する
- **本研究との違い**
 - ▶ 既存手法は特定のモジュールを頻繁に開発している開発者を検索するための手法
 - ▶ 人材配置を行うためには全工程の開発者を検索する必要がある

[1] Audris Mockus and James D. Herbsleb. Expertise browser: A quantitative approach to identifying expertise. In Proceedings of the 24th International Conference on Software Engineering, ICSE '02, pp. 503-512, New York, NY, USA, 2002. ACM.

[2] Romain Robbes and David R'othlisberger. Using developer interaction data to compare expertise metrics. In Proceedings of the 10th Working Conference on Mining Software Repositories, MSR '13, pp. 297-300, Piscataway, NJ, USA, 2013. IEEE Press

目的とアプローチ

- 目的

- ▶ ソースコードを変更しない要件定義や設計に関わっている開発者も含めた開発者の検索手法の構築

- アプローチ

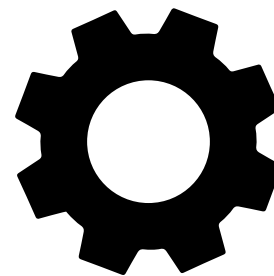
- ▶ 全工程の開発者が利用すると考えられる，開発を行うためのコミュニケーションツールのデータを用いて検索手法を構築する
 - 本研究ではメーリングリストを用いて手法を構築する

提案手法の概要

- コミュニケーションデータに含まれる用語を利用して開発者を検索する
 - ▶ 各開発者が使用した用語を、開発者の開発に関する経験とする
 - ▶ 検索語に関連のある用語を利用した開発者を検索する

検索に用いる
キーワード

(技術用語・クラス名・プロジェクト名)



提案手法

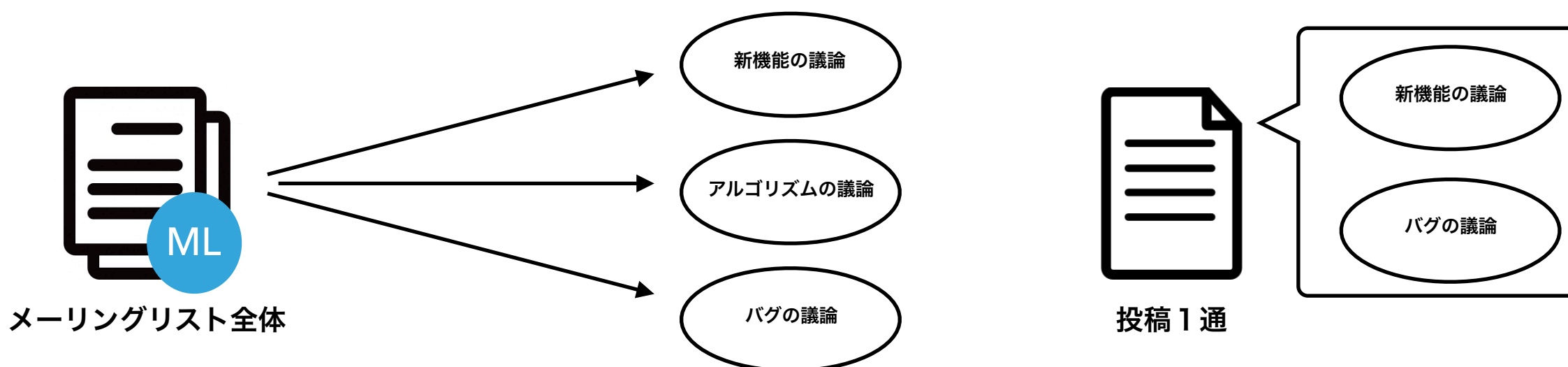


1. 
2. 
3. 
- ⋮
10. 

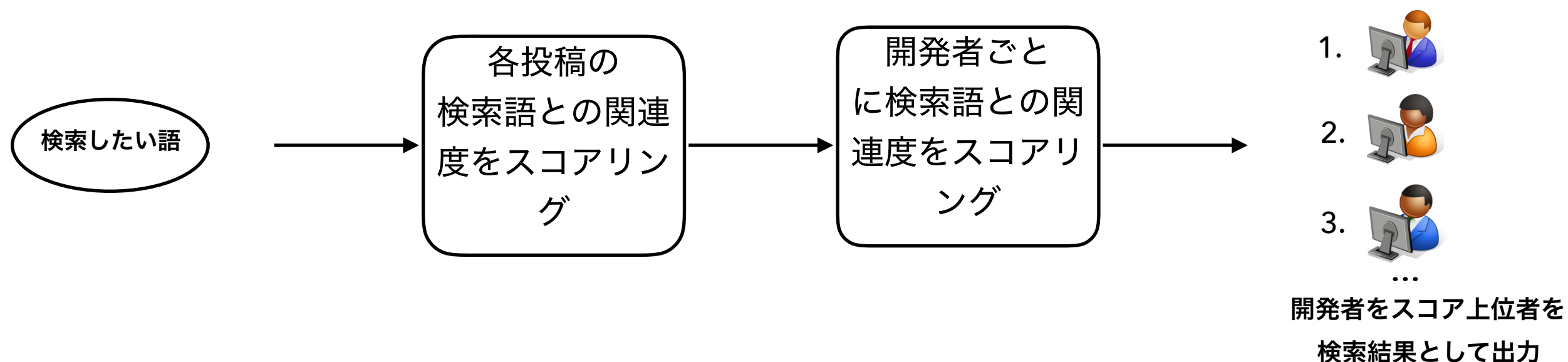
キーワードに関連する
開発者を出力

提案手法の構築の流れ

1. メーリングリスト全体に含まれる内容・各投稿の内容を特定

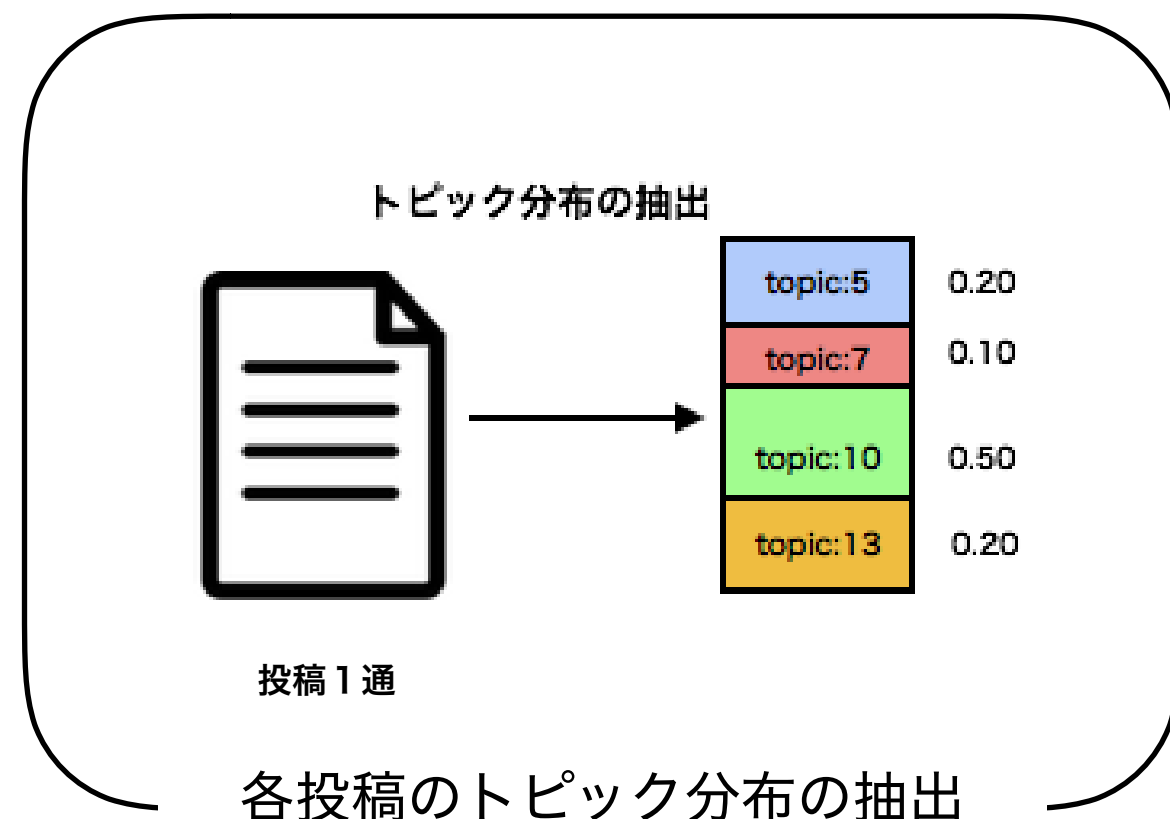
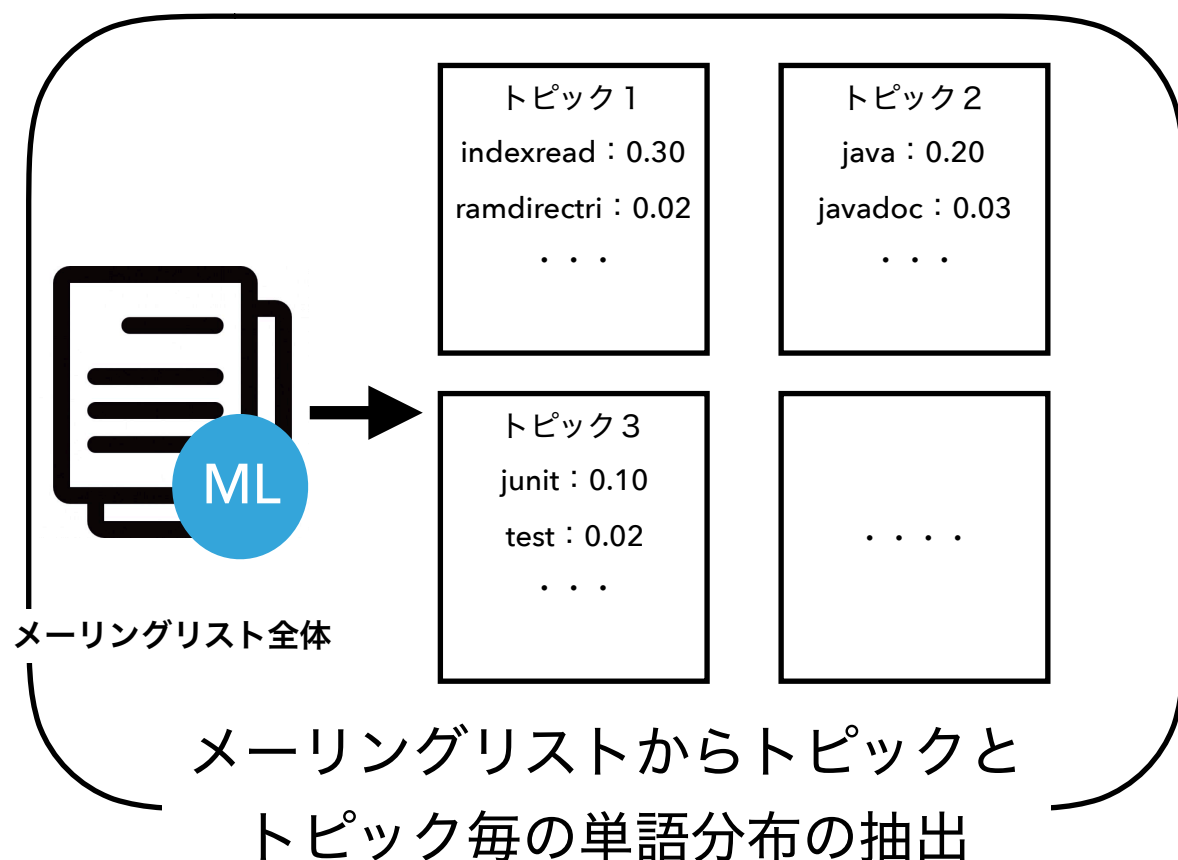


2. 技術用語や特定のプロジェクトに関する用語を入力として、開発者ごとに送信したメールと検索語との関係をスコアリング



1.メーリングリストと各投稿の内容の特定

- メーリングリストを目視で確認して内容の把握を行うことは困難
 - ▶ トピックモデルの潜在的ディリクレ分布（LDA）を用いる
- メーリングリストに含まれるトピック，各投稿の内容をトピック分布を利用して抽出



2.開発者のスコアリング

- 検索語との関連度をトピックの単語分布と各投稿のトピック分布を利用して算出する

1. 検索語に対する各投稿のスコアリング

$T_{w_i} = (t_1, t_2, t_3, t_4, t_5)$ 上位5つのトピック

$$DocumentScore_{w_i, d_j} = \sum_{k=1}^5 \theta_{t_k, w_i} \times \phi_{t_k, d_j}$$

検索語の生起確率 投稿のトピック分布

2. 検索語に対する各開発者のスコアリング

$$AuthorScore_{dev_n} = \sum_{m=1}^{NumberOfMail} DocumentScore_{w_i, d_m}$$

θ_{t_k, w_j} : トピック t_k における検索語 w_i の生起確率

ϕ_{t_k, d_j} : 投稿 d_j におけるトピック t_k のトピック分布の比率

実験

- 目的

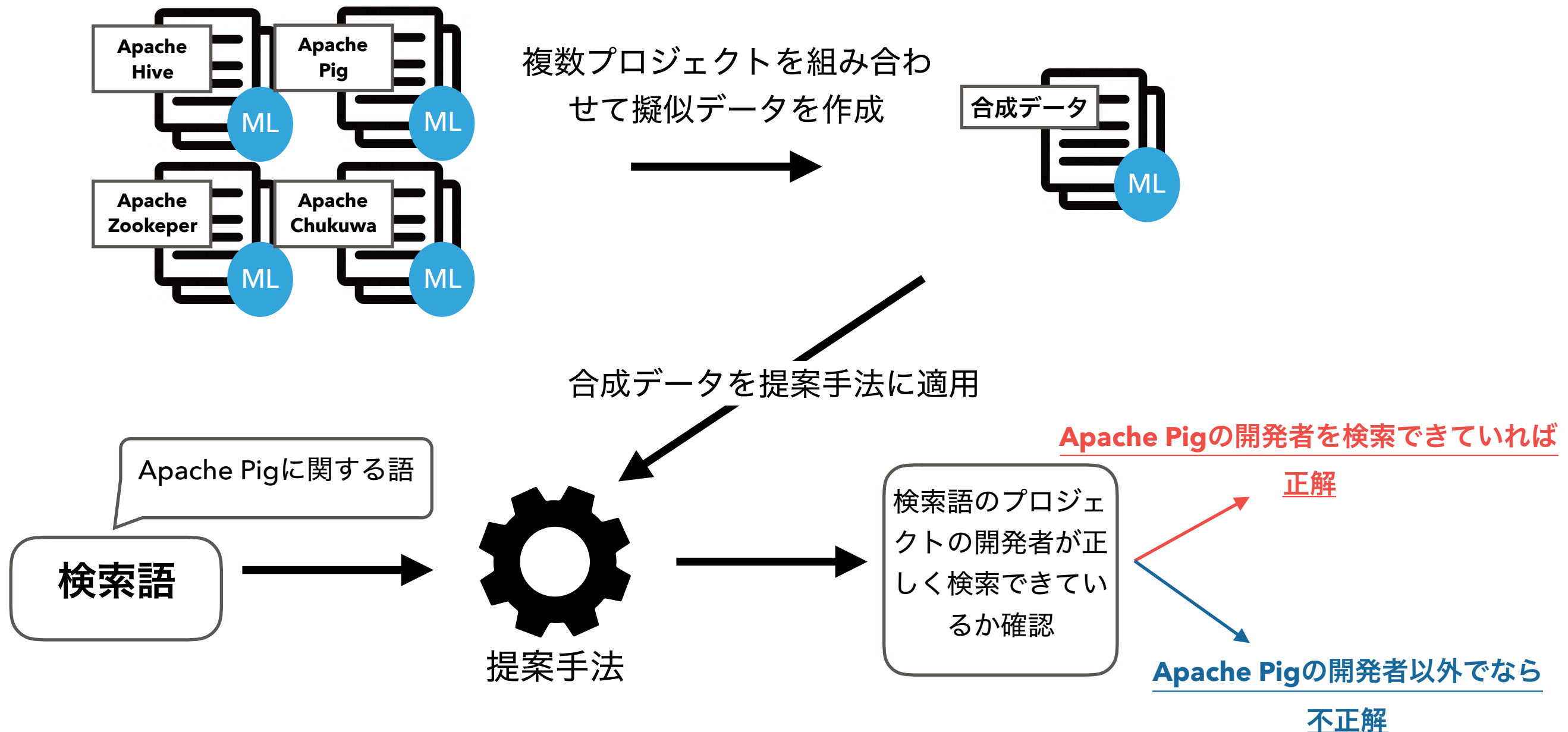
- ▶ 提案手法によるキーワード検索により，キーワードに関する開発に関わっている開発者を正しく検索できることを確認する

- 実験方法

1. 正解集合を作成するために複数のOSSプロジェクトのメーリングリストを合成する
2. 各プロジェクトに関する語を用いて開発者を検索し，各プロジェクトの開発者を検索できているか確認する

実験の概要

- 提案手法の検索結果を正答率を評価する



実験手順の詳細

1. 複数のプロジェクトが混在したメールデータを作成するため、OSSプロジェクトを組み合わせて合成データ
2. 各OSSプロジェクトの検索語を抽出
 - ▶ 検索語：各プロジェクトの開発者を検索したいため、プロジェクトに関連する語
3. 合成データに対して手法を適用
4. 検索語のプロジェクトの開発者を正解集合として、トピック数を100, 500, 1000に設定しスコアの高い上位10名の正答率で評価を行う

対象プロジェクト

.....

プロジェクト	期間	件数	開発者数
Apache Hive	2008/11/11~2010/10/07	1,881	28
Apache Pig	2007/10/31~2010/09/28	1,662	44
Apache Zookeeper	2007/10/31~2010/09/28	1,506	23
Apache Chukuwa	2009/03/10~2013/10/08	877	25
合計		4,209	118

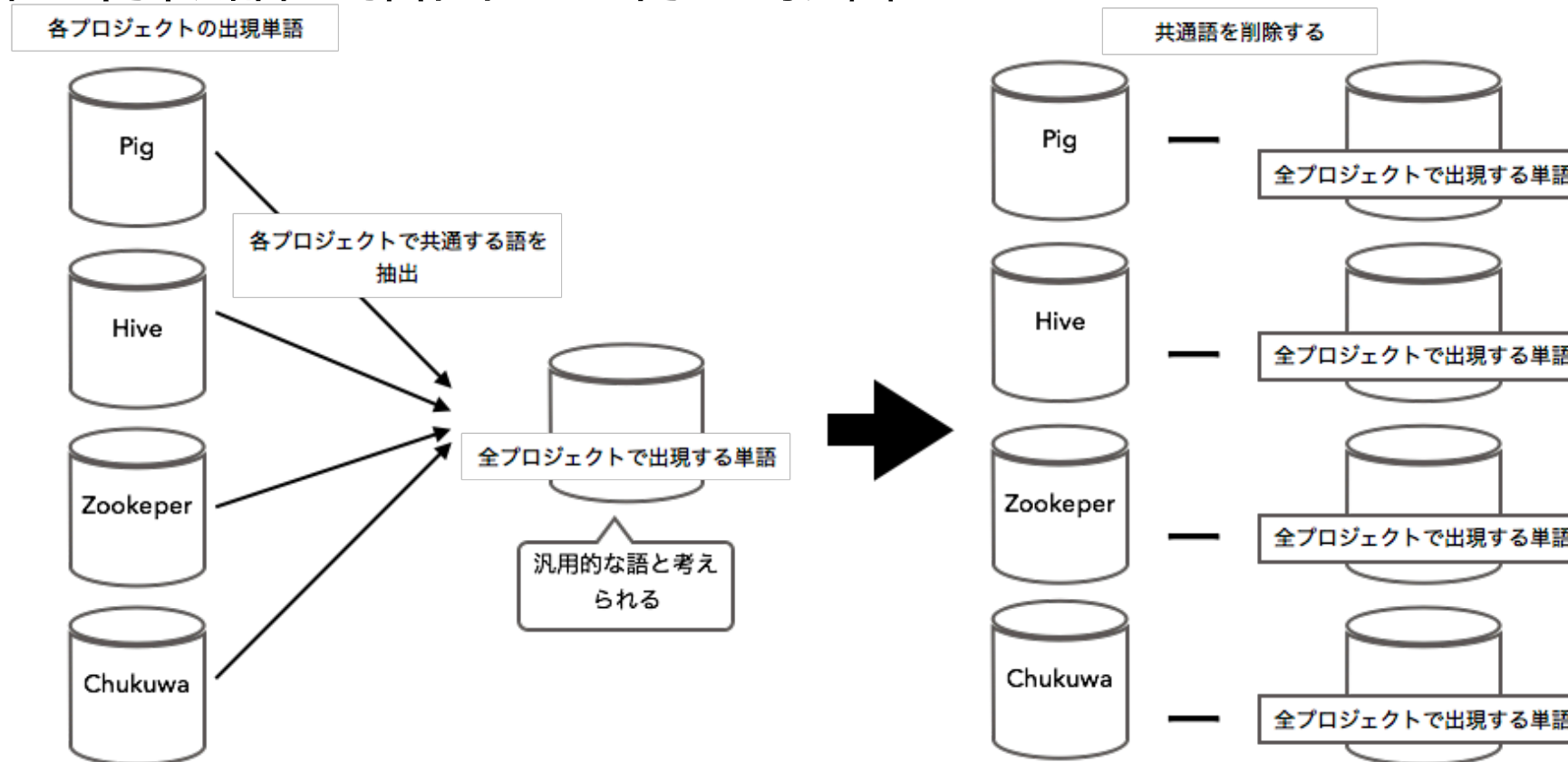
- Bugzilla JIRA Jenkinsなどの通知は削除済み

検索語の抽出手順

- 各プロジェクトの開発者を検索するための語を抽出

1. 各プロジェクトから汎用語の削除

- 頻出語と特徴語を抽出する際に影響を与えるため



- ## 2. 各プロジェクトごとに単語のtf-idf値, df値を取り, それぞれ上位10単語を特徴語と頻出語として抽出

抽出したドメイン用語一覧（特徴語）

.....

	Apache Hive	Apache Pig	Apache Zookeeper	Apache Chukuwa
1	hiveopensource	physicallay	nioservercnxn	adapter
2	srcpart	pig	zookeep	chukwa
3	metaexcept	mapreducelay	nioservercxn	hicc
4	jobconf	cocoon	todd	demux
5	thrift	gld	clientcnxn	collector
6	srcbucket	inactiveaccount	socket	seqfilewrit
7	protect	piggybank	peer	agent
8	numreducetask	eq	antlib	row
9	processnam	acct	hunt	depart
10	libjar	foreach	elect	scienc

抽出したドメイン用語一覧（頻出語）

.....

	Apache Hive	Apache Pig	Apache Zookeeper	Apache Chukuwa
1	hiveopensource	pig	zookeep	chukuwa
2	hive	physicallay	hunt	adaptor
3	srcpart	mapreducelay	nioservercnxn	hicc
4	thrift	schema	clientcnxn	demux
5	metaexcept	foreach	todd	collector
6	jobconf	gate	elect	scienc
7	srcbucket	cocoon	socket	depart
8	protect	piggybank	nioservercxn	jiaqi
9	numreducetask	gld	lock	agent
10	metastor	javacc	peer	seqfilewrit

Apache Hive : 特徴語

	topic:100		topic:500		topic:1000	
検索語	正答数	正答率	正答数	正答率	正答数	正答率
hiveopensource	8	80%	7	70%	6	60%
srcpart	7	70%	10	100%	6	60%
metaexcept	8	80%	6	60%	6	60%
jobconf	8	80%	3	30%	8	80%
thrift	8	80%	5	50%	5	50%
srcbucket	8	80%	5	50%	7	70%
protect	8	80%	10	100%	7	70%
numreducetas	9	90%	5	50%	6	60%
processnam	8	80%	5	50%	6	60%
libjar	9	90%	5	50%	5	50%
平均	8.1	81%	6.1	61%	6.2	62%

他のプロジェクトの正答率の平均と比較すると最も低い結果

Apache Pig : 特徴語

検索語	topic:100		topic:500		topic:1000	
	正答数	正答率	正答数	正答率	正答数	正答率
physicallay	10	100%	10	100%	10	100%
pig	9	90%	10	100%	10	100%
mapreducelay	10	100%	10	100%	10	100%
cocoon	10	100%	10	100%	10	100%
gld	8	80%	10	100%	10	100%
inactiveaccount	10	100%	10	100%	10	100%
piggybank	7	70%	10	100%	9	90%
eq	9	90%	10	100%	10	100%
acct	10	100%	10	100%	10	100%
foreach	8	80%	10	100%	10	100%
平均	9.1	91%	10	100%	9.9	99%

他のプロジェクトの正答率の平均と比較すると最も高い結果

結果：まとめ

- 各プロジェクトとトピック数の設定によって正答率の差がある
 - ▶ 正答率の平均は61%~100%
- Apache Pigの特徴語を用いてトピック数を500に設定
 - ▶ 正答率の平均は100%で最も高い
- Apache Hiveの特徴語を用いてトピック数を500に設定
 - ▶ 正答率の平均は61%で最も低い

検索語の考察

正答率が半分以下を示すトピック数があった検索語

thrift	libjar	javacc	socket
antlib	row	scienc	hicc
peer	numreduceta	srcbucket	seqfilewrit
jobconf			

- ソフトウェア開発組織では頻繁に利用されるような用語, 複数のプロジェクトで使われる語が多い
 - ▶ thrift : Apache Thrift
 - ▶ javacc : Java Compiler Compiler . . .
- hicc, seqfilewrit, srcbucketは各プロジェクト固有の用語であったが正答率が低かった

トピック数の考察

Apache Chukuwaの特徴語：hicc

topic:500（正答率50%）					topic:1000（正答率90%）				
#390	#445	#97	#116	#394	#70	#832	#978	#303	#516
<u>hicc</u>	data	inject	favorit	fyi	<u>hicc</u>	agggreg	jetti	fault	lot
timestam	<u>chukwa</u>	jun	rhel	ann	process	pig	junit	correct	u
fault	hbase	machin	<u>chukwa</u>	anyon	weekend	tool	jiaqi	pig	edg
encoura	effect	tracker	releas	idea	eric	look	tanjiaqi	reus	rough
weekend	step	asrabkin	interest	told	<u>chukwa</u>	algorithm	data	process	futur
talk	use	time	period	detail	data	hbase	tan	java	hbase
<u>chukwa</u>	eric	log	commun	everyth	think	data	broken	inherit	want
asrabkin	mysql	nameno	list	news	regard	minut	right	ari	hadoop
rabkin	requir	project	develop	held	log	henc	think	eric	core
ari	storag	regard	feel	work	yang	<u>hicc</u>	<u>hicc</u>	<u>demux</u>	rest

- 正答率が高いトピック数1000の方が構成語に頻出語，特徴語が含まれていた
- seqfilewrit, srcbucketに関しても正答率の高いトピック数の方が各プロジェクトに関するトピックを特定できていた

まとめ

- 本研究では開発者のコミュニケーションに基づいた開発者の検索手法を提案した
- 正答率はプロジェクトとトピック数で差はあったが、各トピック数ごとの正答率の平均は61%~100%であった
- 特に正答率の低かった検索語は、ソフトウェア開発において頻繁に利用される語、複数のプロジェクトで使われている語が多かった
- 正答率の低い検索語は、検索したいプロジェクトに関するトピックを特定できていない場合が多いと考えられる

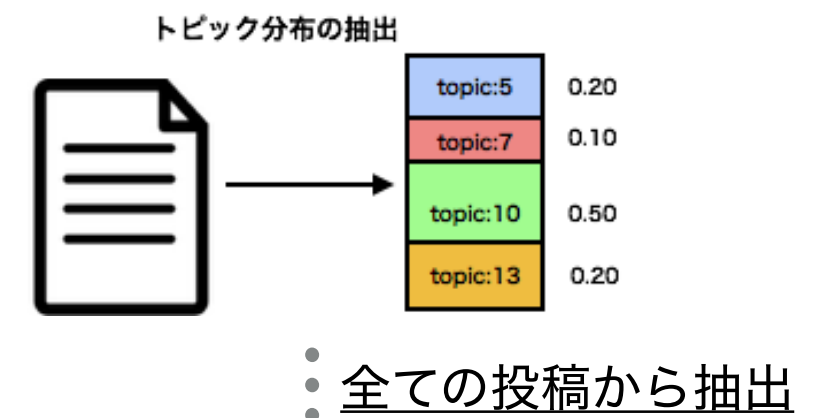
予備スライド

スコアリング例

1. 検索語の生起確率の 高いトピックを5つ特定

トピック	生起確率
#5	0.01
#7	0.20
#13	0.13
#15	0.01
#20	0.30

2. 各投稿のトピック分布 を抽出



3. 各投稿のDocumentScore（例：投稿A）の算出

トピック	比率
#5	0.20
#7	0.10
#10	0.50
#13	0.20

投稿Aのトピック分布

$$\begin{aligned} DocumentScore_{d_A} &= \theta_{t_5, w} \times \phi_{t_5, d_A} + \theta_{t_7, w} \times \phi_{t_7, d_A} + \theta_{t_{13}, w} \times \phi_{t_{13}, d_A} \\ &= 0.01 \times 0.20 + 0.10 \times 0.20 + 0.13 \times 0.20 \\ &= 0.048 \end{aligned}$$

⋮ 同様に全ての投稿で計算

4. 各開発者のAuthor（例：開発者A） スコアを算出

例：開発者Aが投稿A, B, Cを送信していたとすると

$$AuthorScore_{dev_A} = DocumentScore_{d_A} + DocumentScore_{d_B} + DocumentScore_{d_C}$$

⋮ 同様に全ての開発者で計算



スコア上位の開発者を検索結果とする

Apache Hive : 特徴語

	topic:100		topic:500		topic:1000	
検索語	正答数	正答率	正答数	正答率	正答数	正答率
hiveopensourc	8	80%	7	70%	6	60%
srcpart	7	70%	10	100%	6	60%
metaexcept	8	80%	6	60%	6	60%
jobconf	8	80%	3	30%	8	80%
thrift	8	80%	5	50%	5	50%
srcbucket	8	80%	5	50%	7	70%
protect	8	80%	10	100%	7	70%
numreducetask	9	90%	5	50%	6	60%
processnam	8	80%	5	50%	6	60%
libjar	9	90%	5	50%	5	50%
平均	8.1	81%	6.1	61%	6.2	62%

Apache Hive : 頻出語

	topic:100		topic:500		topic:1000	
検索語	正答数	正答率	正答数	正答率	正答数	正答率
hiveopensource	8	80%	7	70%	6	60%
hive	7	70%	7	70%	7	70%
srcpart	7	70%	10	100%	6	60%
thrift	8	80%	5	50%	5	50%
metaexcept	8	80%	6	60%	6	60%
jobconf	8	80%	3	30%	8	80%
srcbucket	8	80%	5	50%	7	70%
protect	8	80%	10	100%	7	70%
numreducetask	9	90%	5	50%	6	60%
metastor	7	70%	7	70%	8	80%
平均	7.8	78%	6.5	65%	6.6	66%

Apache Pig : 特徴語

	topic:100		topic:500		topic:1000	
検索語	正答数	正答率	正答数	正答率	正答数	正答率
physicallay	10	100%	10	100%	10	100%
pig	9	90%	10	100%	10	100%
mapreducelay	10	100%	10	100%	10	100%
cocoon	10	100%	10	100%	10	100%
gld	8	80%	10	100%	10	100%
inactiveaccount	10	100%	10	100%	10	100%
piggybank	7	70%	10	100%	9	90%
eq	9	90%	10	100%	10	100%
acct	10	100%	10	100%	10	100%
foreach	8	80%	10	100%	10	100%
平均	9.1	91%	10	100%	9.9	99%

Apache Pig : 頻出語

	topic:100		topic:500		topic:1000	
検索語	正答数	正答率	正答数	正答率	正答数	正答率
pig	9	90%	10	100%	10	100%
physicallay	10	100%	10	100%	10	100%
mapreducelay	10	100%	10	100%	10	100%
schema	9	90%	10	100%	9	90%
foreach	8	80%	10	100%	10	100%
gate	6	60%	9	90%	10	100%
cocoon	10	100%	10	100%	10	100%
piggybank	7	70%	10	100%	9	90%
gld	8	80%	10	100%	10	100%
javacc	9	90%	10	100%	3	30%
平均	8.6	86%	9.9	99%	9.1	91%

Apache Zookeeper : 特徴語

	topic:100		topic:500		topic:1000	
検索語	正答数	正答率	正答数	正答率	正答数	正答率
nioservercnxn	6	60%	8	80%	10	100%
zookeep	7	70%	7	70%	9	90%
nioservercxn	7	70%	8	80%	7	70%
todd	6	60%	7	70%	6	60%
clientcnxn	7	70%	8	80%	8	80%
socket	5	50%	6	60%	5	50%
peer	5	50%	7	70%	7	70%
antlib	4	40%	4	40%	4	40%
hunt	6	60%	7	70%	8	80%
elect	8	80%	8	80%	9	90%
平均	6.1	61%	7	70%	7.3	73%

Apache Zookeeper：頻出語

	topic:100		topic:500		topic:1000	
検索語	正答数	正答率	正答数	正答率	正答数	正答率
zookeep	7	70%	7	70%	9	90%
hunt	6	60%	7	70%	8	80%
nioservercnxn	6	60%	8	80%	10	100%
clientcnxn	7	70%	8	80%	8	80%
todd	6	60%	7	70%	6	60%
elect	8	80%	8	80%	9	90%
socket	5	50%	6	60%	5	50%
nioservercxn	7	70%	8	80%	7	70%
lock	9	90%	9	90%	6	60%
peer	5	50%	7	70%	7	70%
平均	6.6	66%	7.5	75%	7.5	75%

Apache Chukuwa : 特徴語

検索語	topic:100		topic:500		topic:1000	
	正答数	正答率	正答数	正答率	正答数	正答率
adapter	9	90%	8	80%	9	90%
chukuwa	8	80%	7	70%	10	100%
hicc	7	70%	5	50%	9	90%
demux	10	100%	7	70%	8	80%
collector	6	60%	7	70%	7	70%
seqfilewrit	7	70%	5	50%	5	50%
agent	6	60%	8	80%	8	80%
row	7	70%	4	40%	3	30%
depart	6	60%	8	80%	10	100%
scienc	5	50%	8	80%	9	90%
平均	7.1	71%	6.7	67%	7.8	78%

Apache Chukuwa : 頻出語

	topic:100		topic:500		topic:1000	
検索語	正答数	正答率	正答数	正答率	正答数	正答率
chukuwa	8	80%	7	70%	10	100%
adaptor	9	90%	8	80%	9	90%
hicc	7	70%	5	50%	9	90%
demux	10	100%	7	70%	8	80%
collector	6	60%	7	70%	7	70%
scienc	5	50%	8	80%	9	90%
depart	6	60%	8	80%	10	100%
jiaqi	7	70%	6	60%	7	70%
agent	6	60%	8	80%	8	80%
seqfilewrit	7	70%	7	70%	5	50%
平均	7.1	71%	7.1	71%	8.2	82%