

FOSE2023 第30回ソフトウェア工学の基礎ワークショップ

自動生成されたパッチが コーディング規約違反に 与える影響の分析

和歌山大学システム工学部

野口優輝 木村祐太 伊原彰紀 大平雅雄

背景

自動プログラム修正 (Automated Program Repair)

- プログラムを修正するためのパッチを自動で生成する技術
- 開発者の不具合修正における労力を減らすことができる

プログラム修正の流れ



APRパッチは開発者が意図する機能を崩すことなく不具合を修正することを期待^[1]

[1] Durieux, T., Martinez, M., Monperrus, M., Sommerard, R., and Xuan, J.: Automatic repair of real bugs: an experience report on the defects4j dataset, Empirical Software Engineering, 2017, pp1936-1964

背景 | APRパッチの品質

パッチの正しさはテストケースに通るかどうかで保証

- プログラムの振る舞いにおいての正しさは保証されるが、コードの中身（書き方）までの正しさは保証されない

パッチが生成したコードの品質

- コーディング規約に従ったコードは可読性や保守性を保ち、コードの品質を高めることができる[2]
- APRパッチはコードの品質を考慮することも重要

コーディング規約に沿ったパッチを生成する必要がある

[2]織田 泰輔, 阿萬 裕久, 佐々木 隆志, 川原 稔, オープンソース開発でのコード修正におけるコーディング規約違反の変化に関する調査, 情報処理学会研究報告会2014, pp1-8

背景 | コーディング規約

ソフトウェアにおける書き方の定義

- 各プロジェクトに独自のルールが設定されている

静的解析ツールで検査

- CheckStyleやPMDを使ってコーディング規約違反を検出[3]

関連研究

- 静的解析ツール使用による欠陥の特定は、目視より容易であり、コーディング規約遵守に有効である[4]

APRパッチがコーディング規約違反に
与える影響を調べた研究は存在しない

[3] Beller, M., Bholanath, R., McIntosh, S., and Zaidman, A.: Analyzing the state of static analysis: a large-scale evaluation in open source software, Proc. SANER, 2016, pp. 470–481.

[4] Panichella, S., Arnaoudova, V., Di Penta, M., and Antoniol, G.: Would static analysis tools help developers with code reviews?, Proc. SANER, 2015, pp. 161–170.

動機と目的

動機

- APRパッチは様々なプロジェクトの不具合情報[1]を元に生成
- ただし、**遵守すべき規約はプロジェクト毎に異なる**
- APRパッチの適用によりコーディング規約違反が混入すると、開発者が関知しない保守性の低下が起こる恐れがある

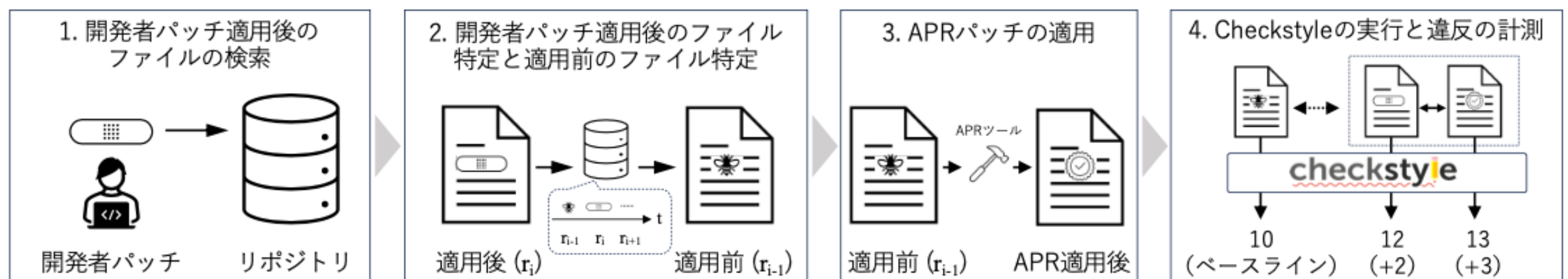
目的

- APRパッチがコーディング規約違反に与える影響を明らかにする
 - コーディング規約違反数を増加させるAPRパッチの割合はどの程度か
 - ベースラインとして開発者パッチと比較

分析手順

分析の流れ

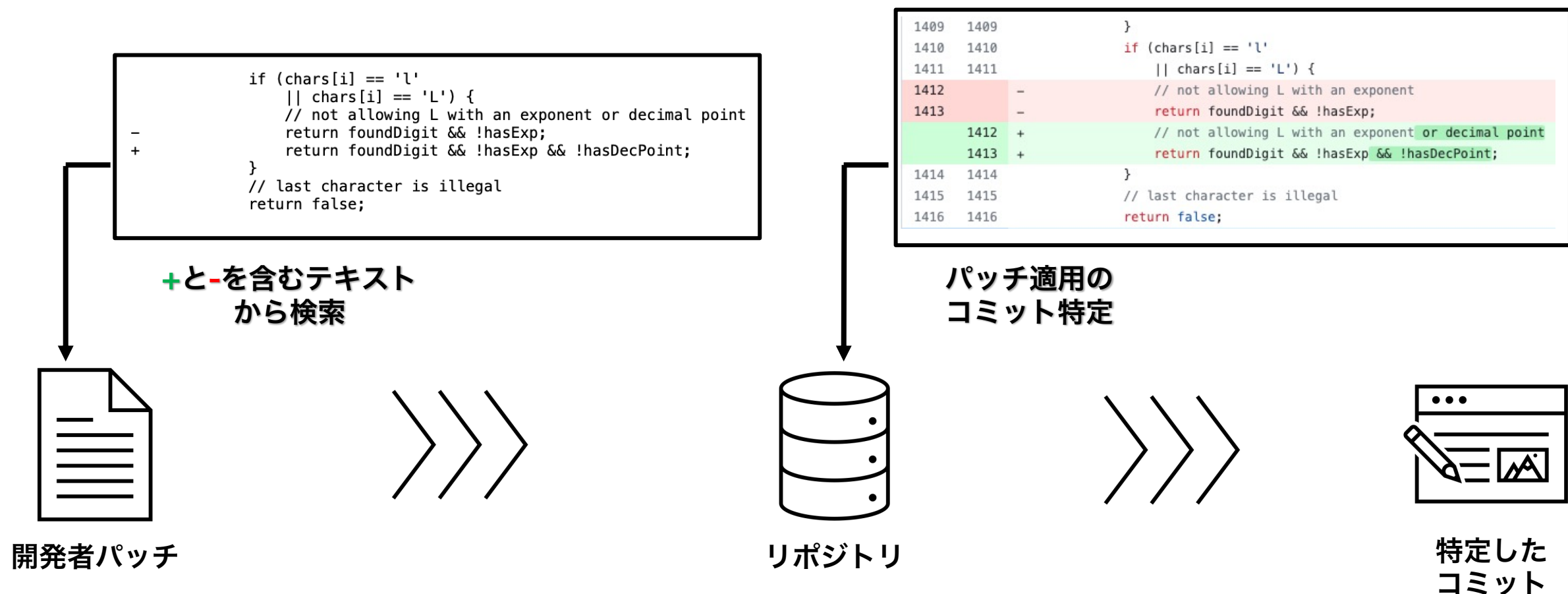
1. 開発者パッチ適用後のファイル検索しコミットを特定
2. 開発者パッチ適用前後のファイルを特定
3. APRツールごとにAPRパッチの適用
4. CheckStyle実行とコーディング規約違反の計測と比較



分析手順1. | パッチ適用のコミット特定

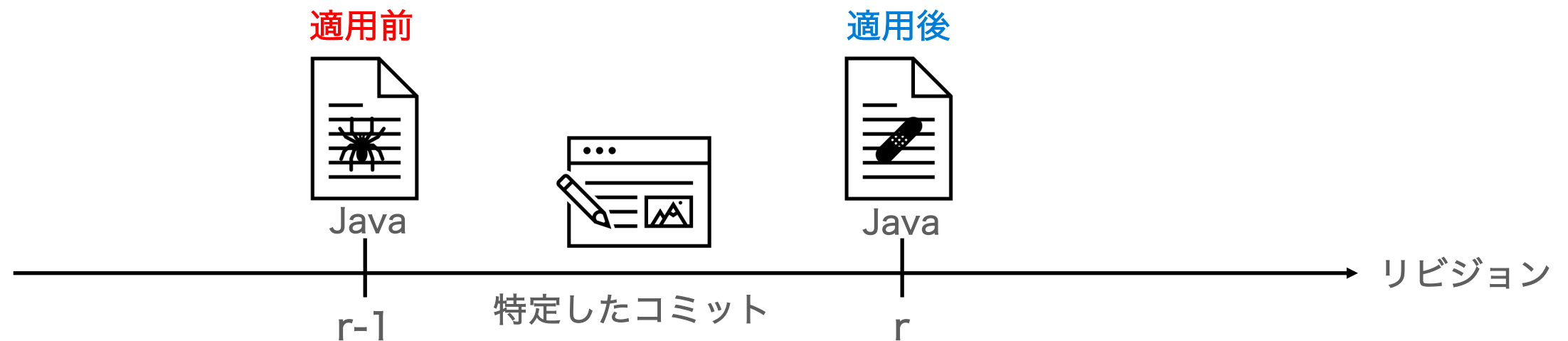
開発者パッチを利用して、実際にパッチが適用されたと思われるコミットを特定

1. パッチにある+と-行を含むテキストを検索
2. パッチが適用されたと思われるコミットを特定

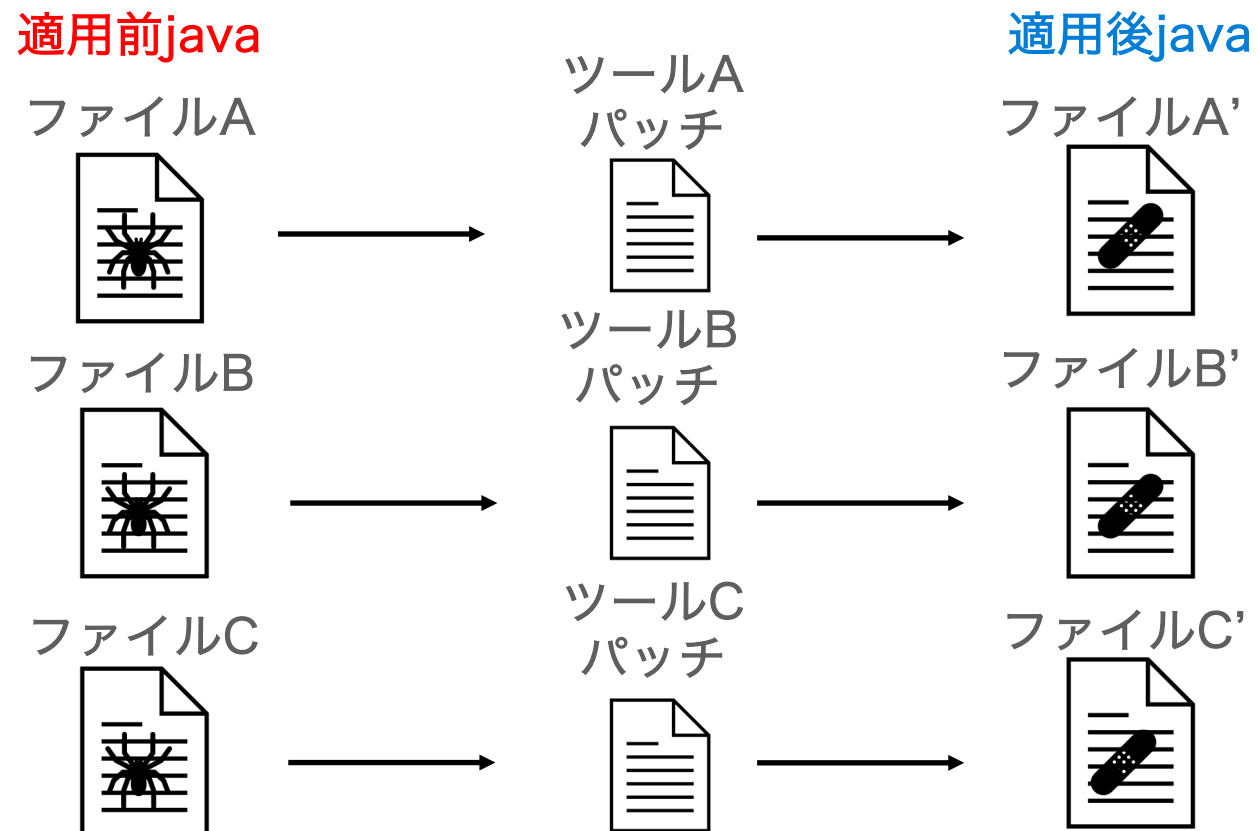


分析手順2-3. | ファイル特定, パッチ適用

開発者パッチが適用される前後のjavaファイルを特定



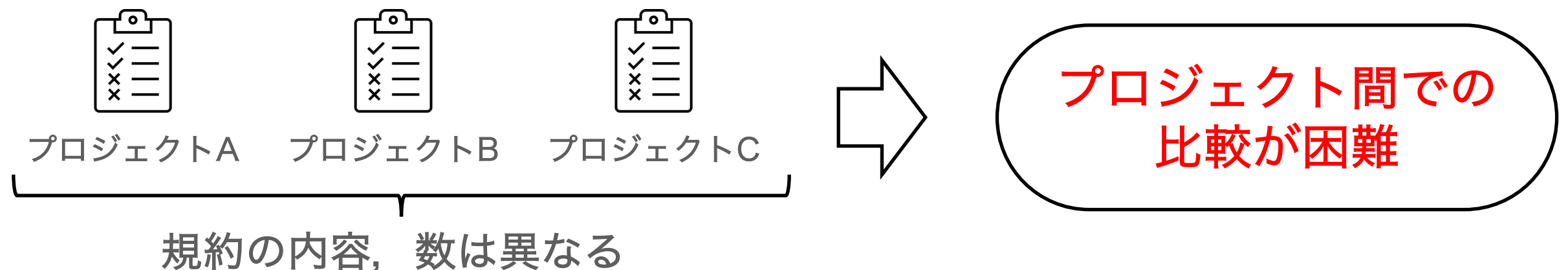
APRパッチ適用



分析手順4. | 使用する設定ファイル

projectスタイル

- プロジェクト独自で定義されたCheckStyleファイル
- プロジェクトごとに規約の内容や数が異なっている

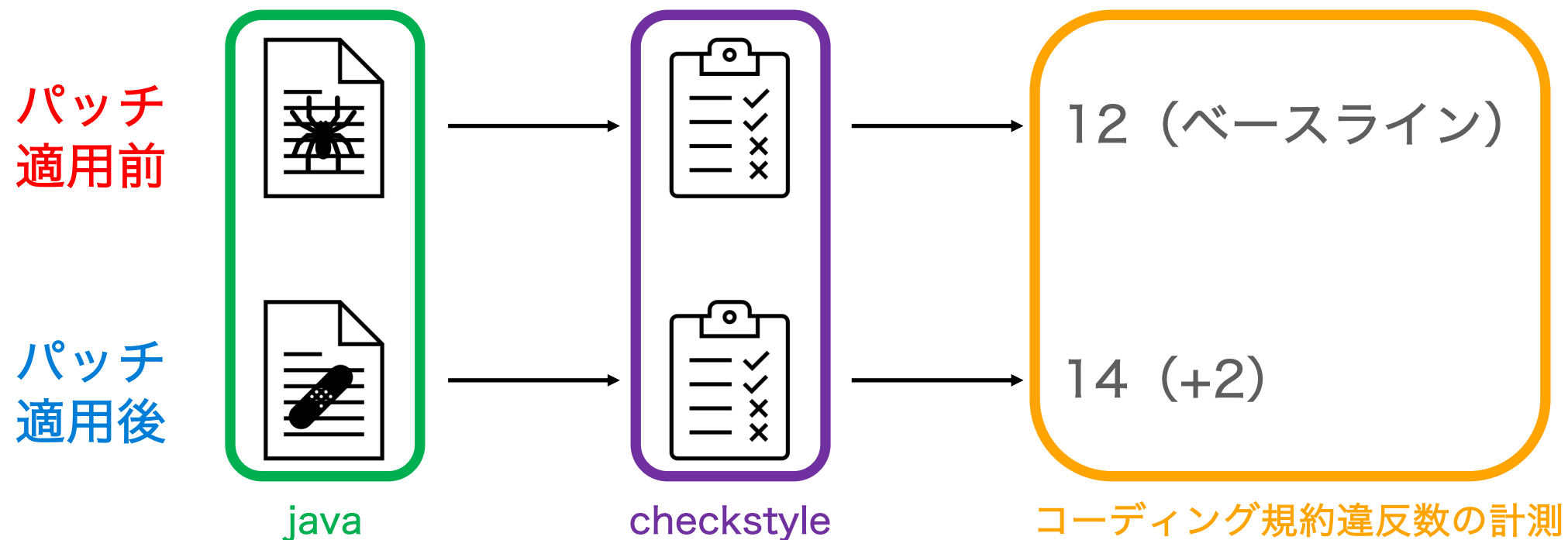


Googleスタイル, Sunスタイル

- 一般的に広く参照されているCheckStyleファイル
- projectスタイルを用いるよりも, 規約違反の変化をよく捉えることができる

分析手順4. | 規約違反数の計測

スタイルごとに規約違反数の変化を計測



パッチ適用前後における規約違反数の割合を計算

$$\text{割合(\%)} = \frac{\text{増加 | 減少 | 変化なし パッチの個数}}{\text{プロジェクトにおけるパッチの個数}} \times 100$$

(増加|減少|変化なし)

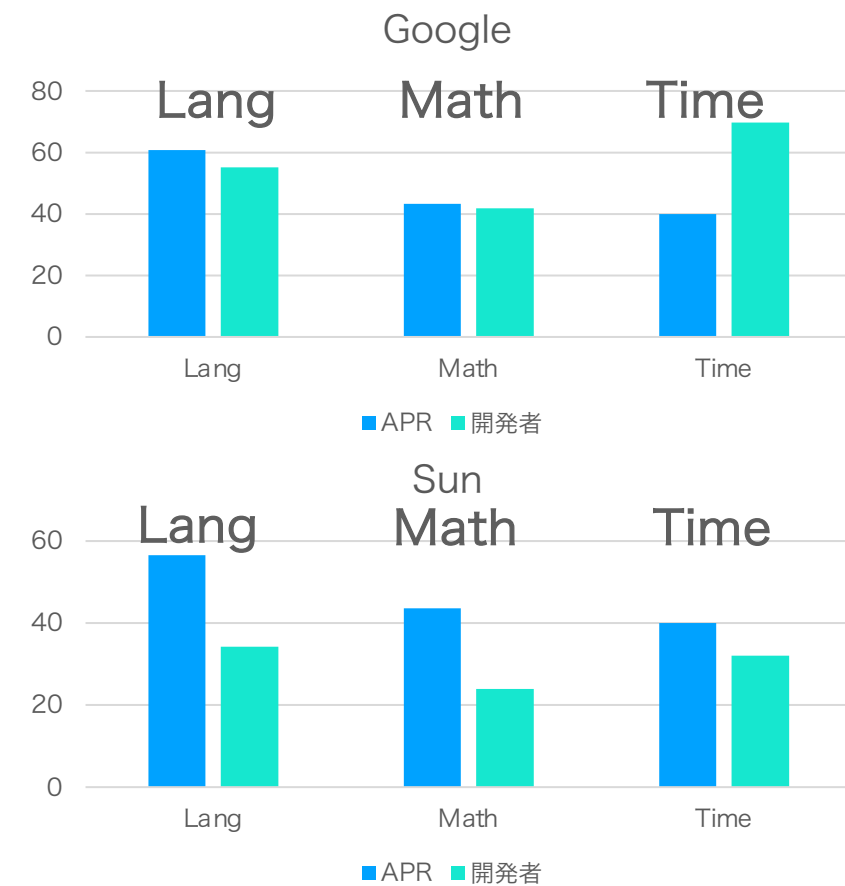
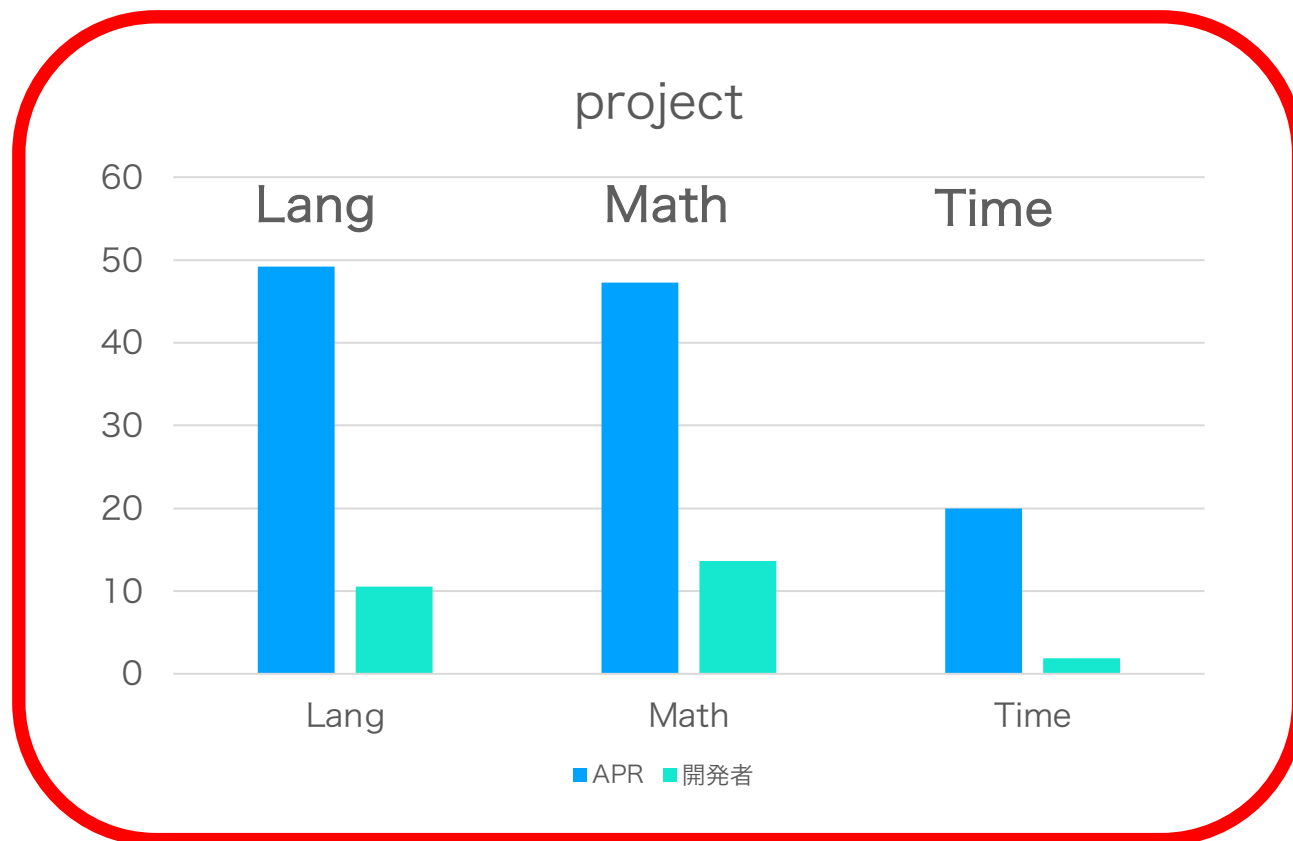
データセット

- [5]の内, CheckStyleを利用するプロジェクト
 - Apache Lang
 - Apache Math
 - Joda Time
- 32種類のAPRツールが生成したパッチ
- 開発者が作成したパッチ

パッチ生成元	Lang	Math	Time	パッチ生成元	Lang	Math	Time
開発者 [12]	105	184	53	jGenProg2015 [5]	0	4	0
3sFix [4]	0	2	0	jKali [23]	2	1	0
ACS [36]	3	15	1	jMutRepair [23]	0	1	0
ARJA [39]	6	10	1	KaliA [39]	0	1	0
CapGen [33]	7	14	0	kPAR [18]	1	4	0
Cardumen [24]	0	1	0	LSRepair [20]	4	1	0
CoCoNut [22]	7	17	1	Nopol2015 [5]	3	0	0
ConFix [14]	6	9	1	PraPR [7]	5	10	3
DeepRepair [34]	4	2	0	RSRepairA [39]	0	0	0
DynaMoth [6]	0	1	0	SequenceR [3]	2	8	0
ELIXIR [27]	8	12	2	SimFix [10]	9	16	1
FixMiner [15]	0	8	0	SketchFix [8]	2	6	0
GenPat [9]	4	3	0	SOFix [21]	3	13	1
GenProgA [39]	0	0	0	ssFix [35]	5	7	0
HDRRepair [16]	1	5	1	Tbar [19]	7	8	0
Hercules [28]	13	30	8	Vfix [37]	7	0	0
JAID [2]	13	11	0	APR パッチ合計	122	220	20
				All (開発者+ APR)	231	434	75

[5] Tian, H., Li, Y., Pian, W., Kaboré, A. K., Liu, K., Habib, A., Klein, J., and Bissyandé, T. F.: Predicting patch correctness based on the similarity of failing test cases, Trans. Softw. Eng. Methodol., Vol. 31, No. 4(2022), pp. 1-30.

結果 | 規約違反数が増加したパッチの割合



全てのスタイルにおいて、APRパッチは規約違反数を増加させる割合が高い

- projectスタイルでは、開発者パッチと比較しても、特に割合が高くなっていることがわかる

考察

projectスタイルでは、APRパッチは開発者パッチよりも規約違反数を増加させるパッチの割合は高い

- 開発者はプロジェクトの規約を意識してパッチを作成している
- APRはプロジェクトの規約に準拠したパッチを生成していない

疑問点

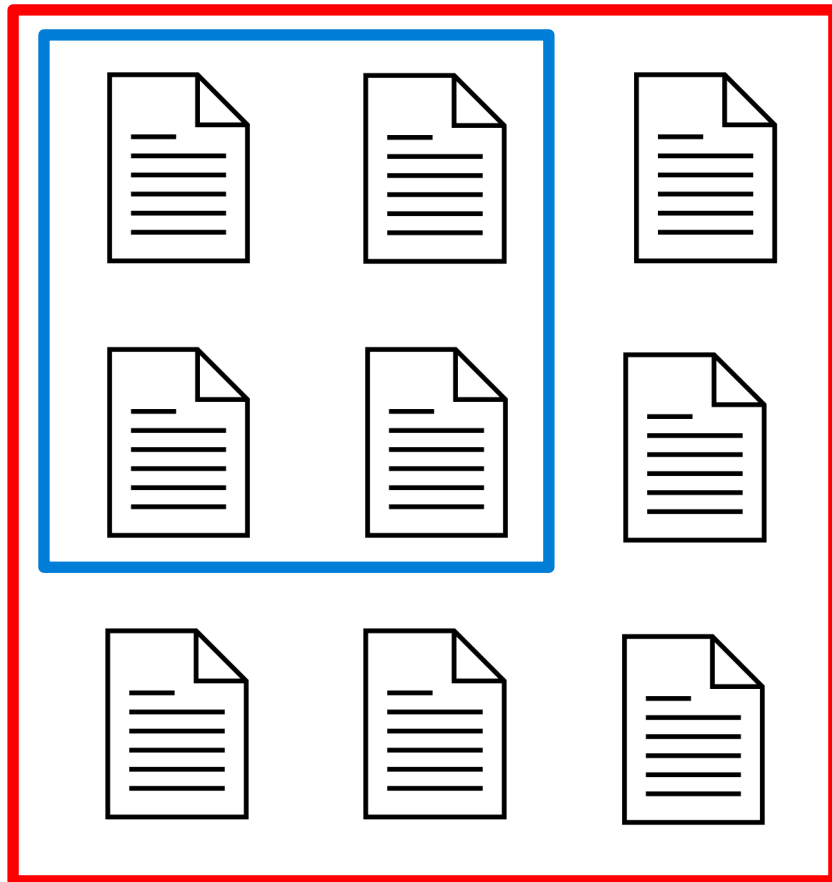
APRが発生させる規約違反の特徴は??

追加分析

- APRパッチが発生させる規約違反を分析
 - プロジェクトごとにルールが異なるため、Google, Sunスタイルを使用
 - ベースラインとして、開発者パッチと比較

追加分析 | 分析手順

1. パッチ適用で発生した規約違反を集計
2. 規約違反ごとにパッチの割合を計算



違反Aが発生

計算方法

$$\text{割合} = \frac{\text{ある違反が発生したパッチの総数}}{\text{違反数が増加したパッチの総数}} \times 100$$

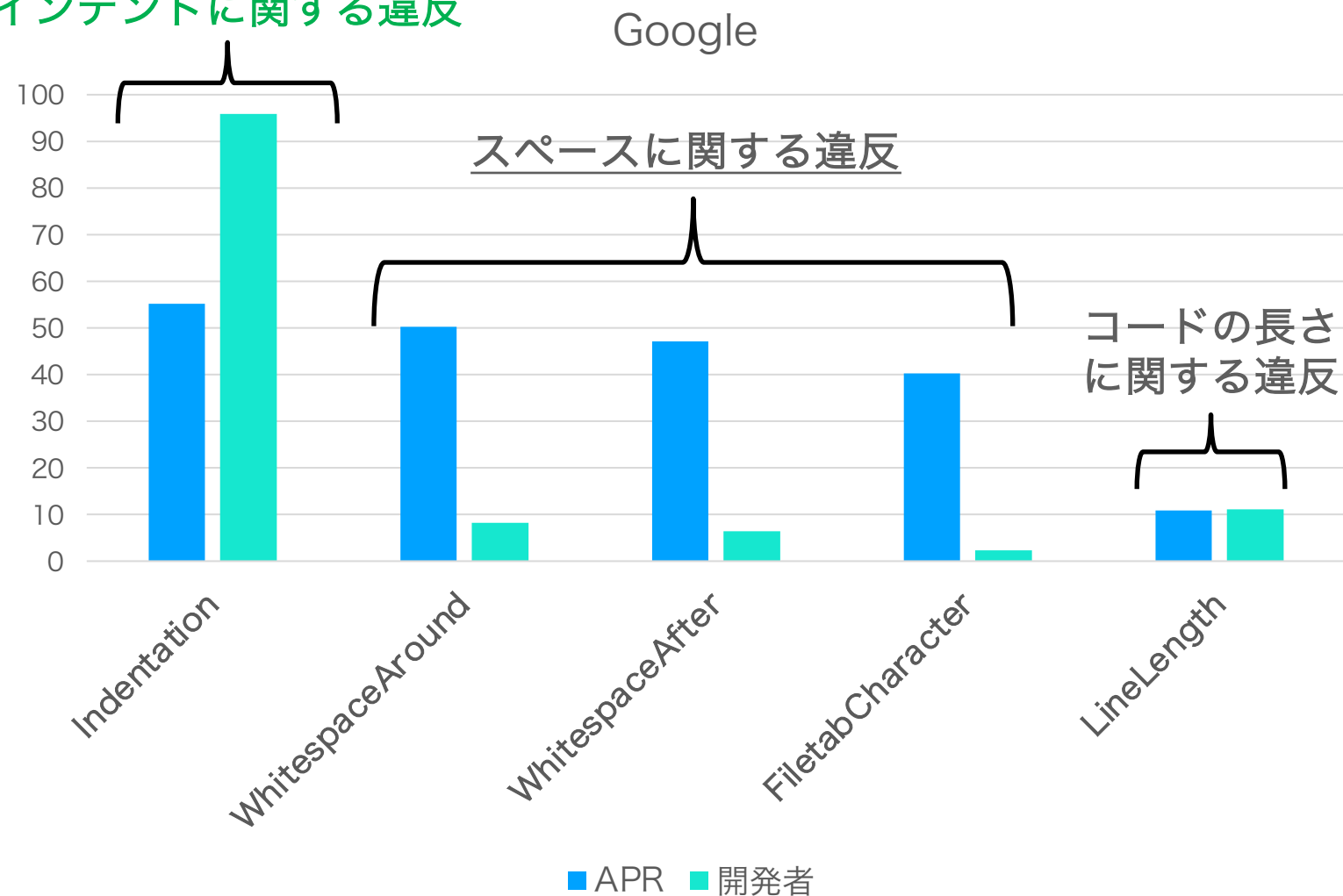
例)

$$\frac{4}{9} \times 100 = 44.44\ldots$$

約44%, 違反Aが発生している

結果 | Googleスタイル (APR vs. 開発者)

インデントに関する違反

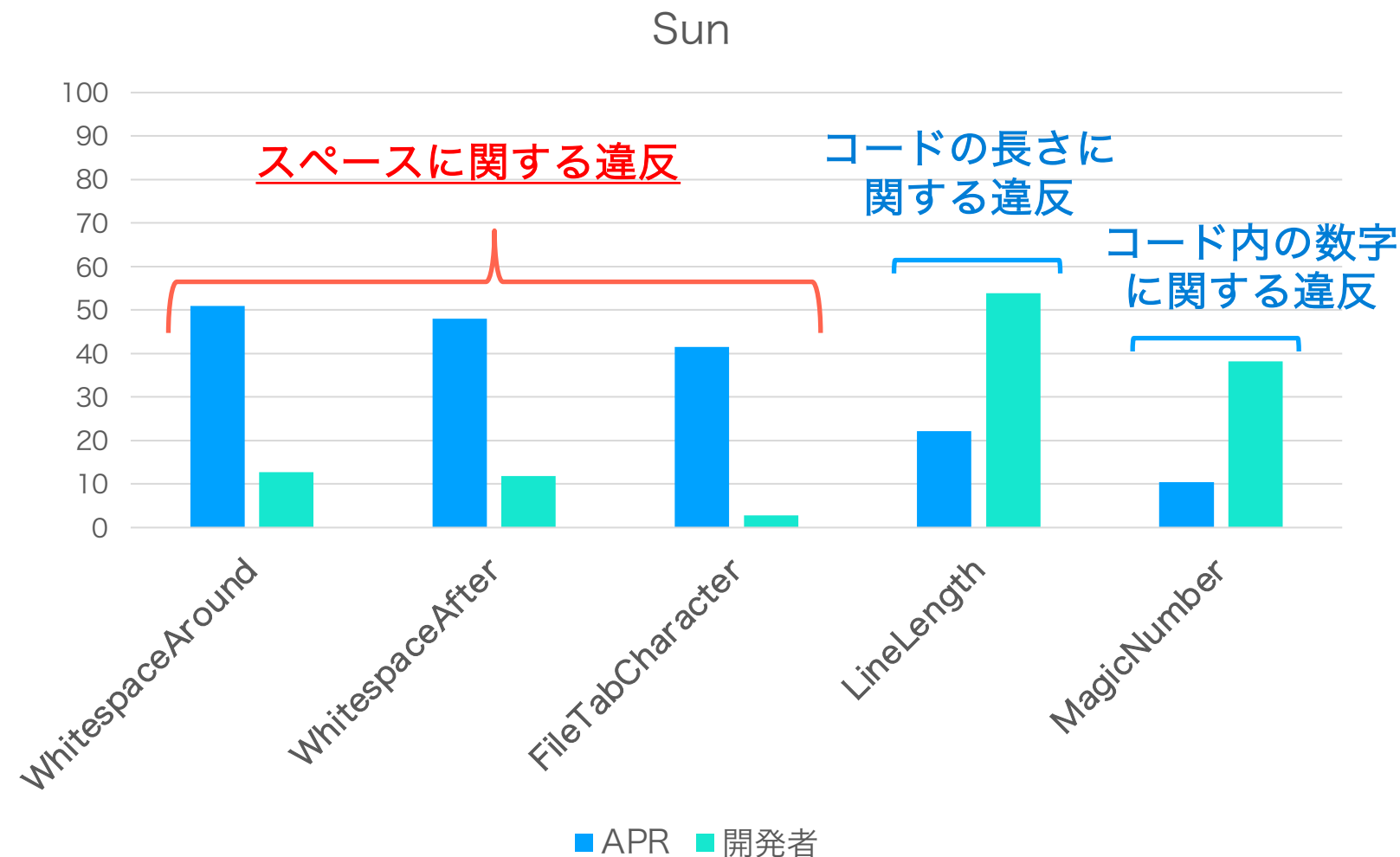


違反名	説明
Indentation	インデントがルールに合致
Whitespace Around	トークンの前後に空白があるか
Whitespace After	特定の文字の後に空白があるか
FileTab Character	タブ文字ではなく空白を使ってるか
LineLength	1行あたりにある文字数

APRは開発者と同様にインデントの違反が多い

APRはスペースに関する違反が多い

結果 | Sunスタイル (APR vs. 開発者)



違反名	説明
Whitespace Around	トークンの前後に空白があるか
Whitespace After	特定の文字の後に空白があるか
FileTab Character	タブ文字ではなく空白を使ってるか
LineLength	1行あたりにある文字数
MagicNumber	マジックナンバーを使用しているか

Google同様, APRはスペースに関する違反が多い

APRは開発者と比較して, コードの長さ, マジックナンバーに関する違反は少ない

WhitespaceAfterの規約違反の例

APRパッチはスペースに関する違反が多い

開発者パッチ適用後

```
public static float max(final float a, final float b) {  
    return (a <= b) ? b : (Float.isNaN(a + b) ? Float.NaN : a);  
}
```

↑
スペースあり

APRパッチ適用後

```
public static float max(final float a, final float b) {  
    return( a <= b ) ? b :( Float.isNaN( a + b ) ? Float.NaN : a ) ;  
}
```

↑
スペースなし

whitespaceAfterの違反発生

APRツールにコード規約遵守の観点を取り入れる必要がある

まとめ・今後の展望

自動生成されたパッチがコーディング規約違反に与える影響を分析

- APRはprojectに準拠したパッチを生成しにくい
- APRパッチは主に、スペースに関する違反を多く発生させる
 - APR技術はコードのトークン化や抽象構文木化を利用していることで、コードのスタイルが失われやすい

今後の展望

- APRツールごとに、コーディング規約違反数の変化を比較

査読者様から頂いた コメントに基づく追加調査

査読者様からのコメント

IDEで違反は自動修正可能では？

- 追加調査を実施

APRそれぞれの実装ごとに比較

CheckStyle以外のツールは？

- PMDも用いて検証

開発者パッチの情報は？

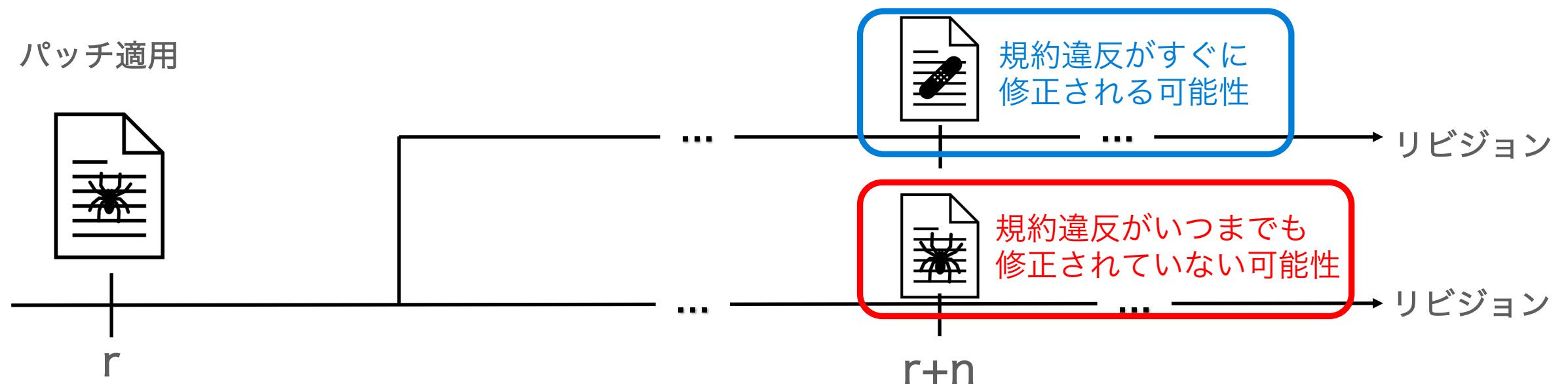
- 開発者経験をもとに検証

今後の研究課題

規約違反はIDEで自動修正可能？

スペースに関する違反, 括弧の位置に関する違反は,
IDEで自動修正することができる

- コーディング規約を遵守したパッチ生成は不要では??



調査目的

パッチ適用後のリビジョンで,
規約違反が修正されているかを明らかにする

調査対象

projectスタイルで違反が発生した開発者パッチ

プロジェクト	パッチ(個)	規約違反(個)
Lang	13	13
Math	41	51

APRパッチは対象にできない

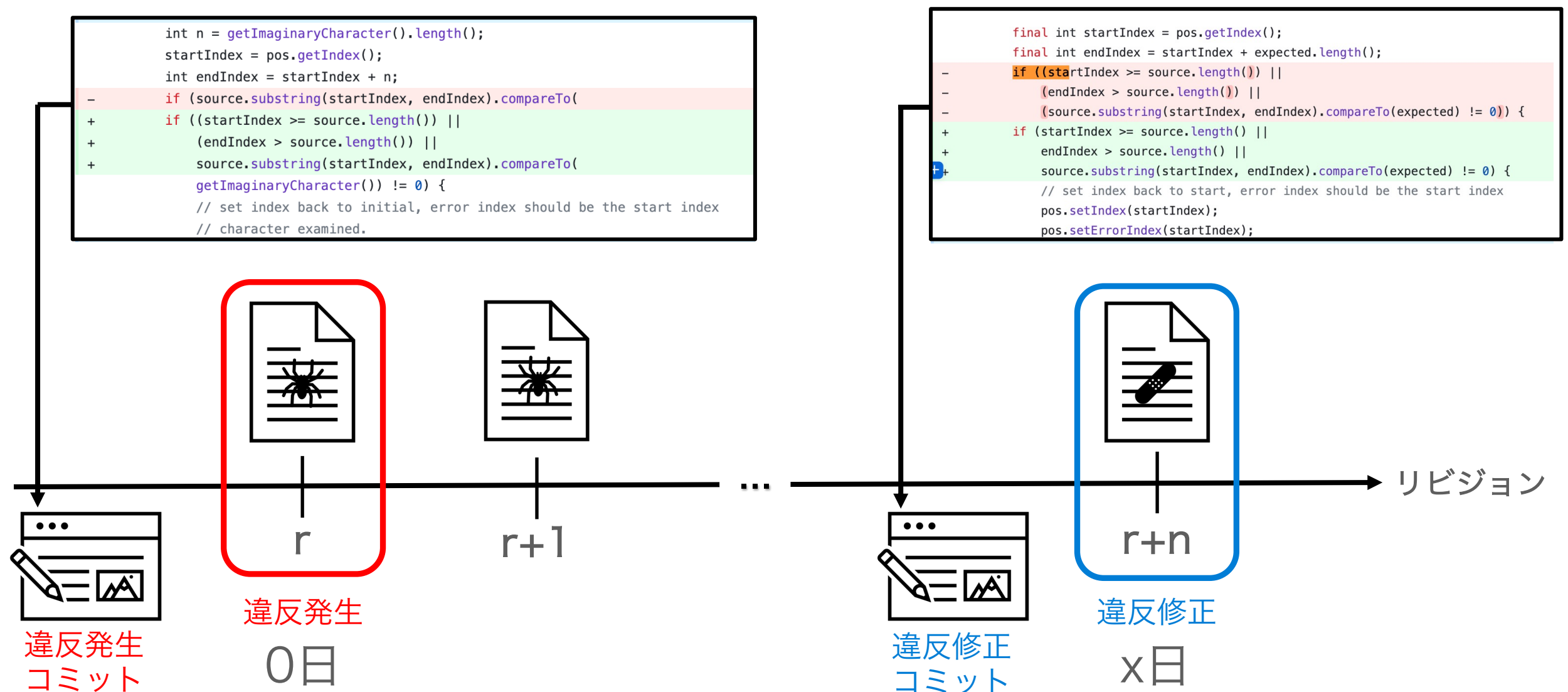
- APRパッチは実際のプロジェクトでは使用されていないため、実際に適用された開発者パッチのみを使用する

開発者パッチでの違反が修正されない場合、
APRパッチを用いた場合でも修正されない可能性

調査方法

パッチ適用部分が変更されているコミットを探索

1. 開発者パッチ適用で発生した違反行を含むコミットを探索
2. 違反修正コミットがある場合，修正までにかかった日数を計測

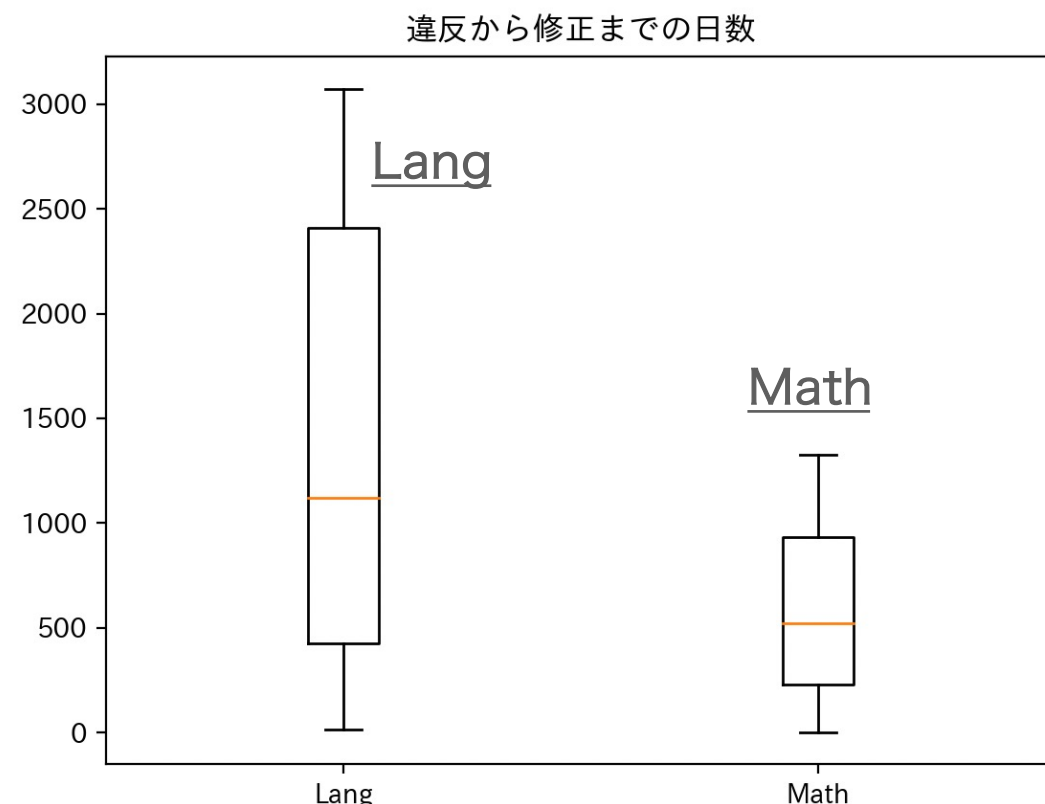


調査結果

Lang, Mathともに大部分の違反が修正されていた。

- Langでは, 11個/13個が修正・変更・削除
- Mathでは, 30個/51個が修正・変更・削除

修正までには相当な日数を要していた。



プロジェクト	平均値(日)	中央値(日)
Lang	1430	1120
Math	978	522

違反から修正までの日数

査読者様への回答

規約違反はIDEで自動修正可能？

- 規約違反は自動修正可能だが、パッチ適用後、実際に修正されているのかは不明
- パッチ適用後のレビューで、規約違反が修正されているか

多くの違反は修正されていたが、修正までにはかなりの日数を要していた

- Lang, Mathでそれぞれ1120日, 522日かかった

ソフトウェア保守性の低下を防ぐためにも、
パッチ適用での規約違反は早めに対処すべき