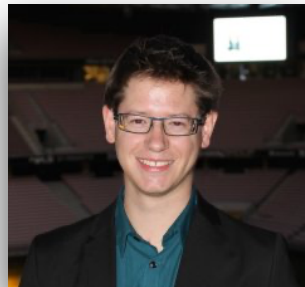
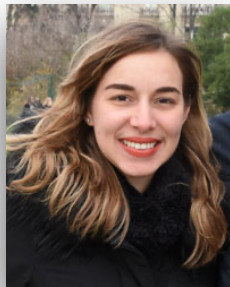


BEARS: An Extensible Java Bug Benchmark for Automatic Program Repair Studies



Fernanda Madeiral, Simon Urli, Marcelo Maia, Martin Monperrus

Proceedings of the 26th IEEE International Conference on Software
Analysis, Evolution and Reengineering (SANER '19)

発表者 吉富 楓雅

背景

- **自動バグ修正**：ソフトウェアのバグの解決策を自動的に発見
 - * 既知のバグを用いて実行することで、経験的な評価を得られる
- バグとパッチの正確性を評価するための基準(**ベンチマーク**)を適切に設計することが重要
 - * **再現が簡素化できる**
 - * **結果の一般性が確保される**
 - * **長期にわたって測定が可能となる**

問題点

- バグの収集は困難な作業である
 - * 自動バグ修正ツールを評価するためには優れたベンチマークを使用する必要がある[1]
 - ▶ 優れたベンチマーク：多様なシステムで簡単にバグを再現できる
- ベンチマークの作成には時間がかかる
 - * バグの再現は困難である

研究の目的とアプローチ

- 目的

- * 自動バグ修正の研究を支援するための、バグを収集しベンチマークとして格納するツールの提案

- ▶ 本研究のベンチマーク：再現可能な既知のバグおよびパッチ

- アプローチ(ツール名：BEARS)

- * BEARS-COLLECTOR：バグとパッチをCIツールにより自動識別

- * BEARS-BENCHMARK：自動識別されたバグおよびパッチを格納

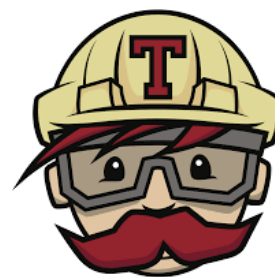
継続的インテグレーション

- コードの変更を定期的にマージ, 定期的・自動的にビルド・テストを実行する手法

- * 複数人での作業の衝突・競合を早期に発見

- 代表的なCIツール

- * Jenkins



Travis CI

- * **Travis CI**

本研究で使用

- * Circle CI



バグとパッチの収集

- ビルドbnの直後bn+1でパッチが適用されるものを収集

ビルドペア (bn, bn+1) = (bbuggy, bpatched)

- ケース 1 : failing-passing builds(テスト変更なし)

* bnの1つ以上のテストケースでfailとなり, bn+1ではpassとなる場合

バグを再現可能

- ケース 2 : passing-passing builds(テスト変更あり)

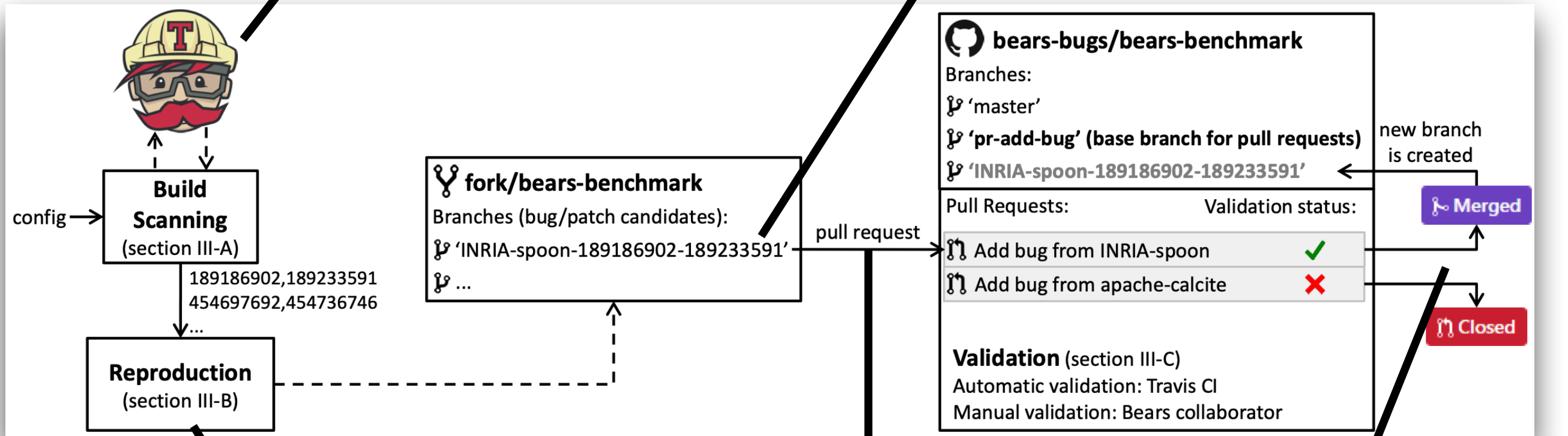
* bn, bn+1ともにpassとなる場合

パッチを再現可能

BEARS-COLLECTOR

①ビルドペアのスキャン

③ペアをブランチに格納



②ビルドペアのテスト

④プルリクエストの作成

⑤Merged or Closed

BEARS-COLLECTOR

- プロセス 1 : ビルドペアのスキャン
 - * Travis CIからビルドIDを自動的にスキャン
 - * 再現するペアを選択し, ケース 1, 2 のいずれに対応するかを確認

ケース 1 failing-passing builds	bbuggyがfailとなる ソースコードの変更はあるがテストコードの変更はない
ケース 2 passing-passing builds	bbuggyがpassとなる ソースコードおよびテストコードの変更がある

BEARS-COLLECTOR

- プロセス2：ビルドペアのテスト
 - * 対象プロジェクトをビルド，テストの実行
 - * **bbuggy**のコミットまでチェックアウトし実行
 - ▶ いずれかのテストケースがfailとなる場合，バグを再現できたとする
 - * **bpatched**のコミットまでチェックアウトし実行
 - ▶ 全てのテストケースがpassする場合，パッチを再現できたとする



BEARS-COLLECTOR

- プロセス 3 : ペアをブランチに格納

- * <project slug>-<buggy build id>-<patched build id>

- ▶ 今回の例ではブランチ 'INRIA-spoon-189186902-189233591'



fork/bears-benchmark

Branches (bug/patch candidates):



'INRIA-spoon-189186902-189233591' -



...

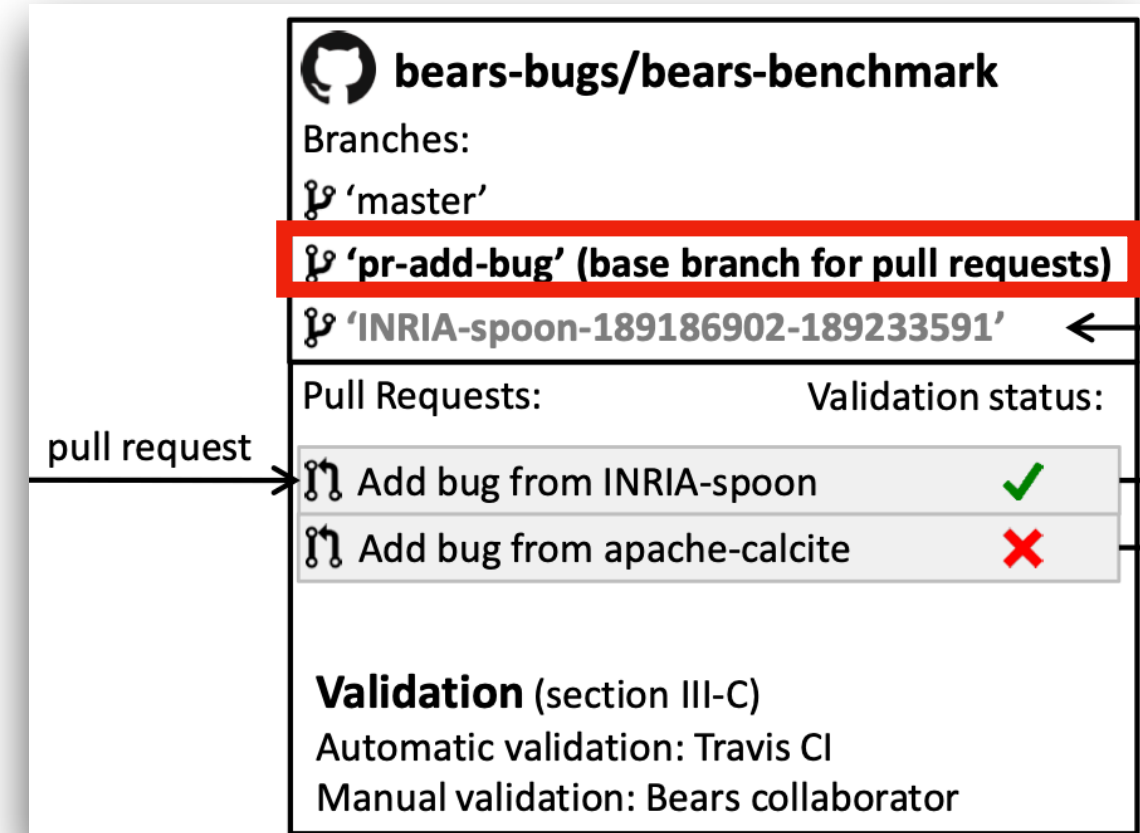
BEARS-COLLECTOR

• プロセス4：プルリクエストの作成

* ブランチをマージするために‘pr-add-bug’ ブランチをベースとしてプルリクエストを作成

▶ 著者ら：プルリクエストが有効か確認(**手動検証**)

▶ Travis CI：スクリプト実行、プルリクエストが有効か確認(**自動検証**)



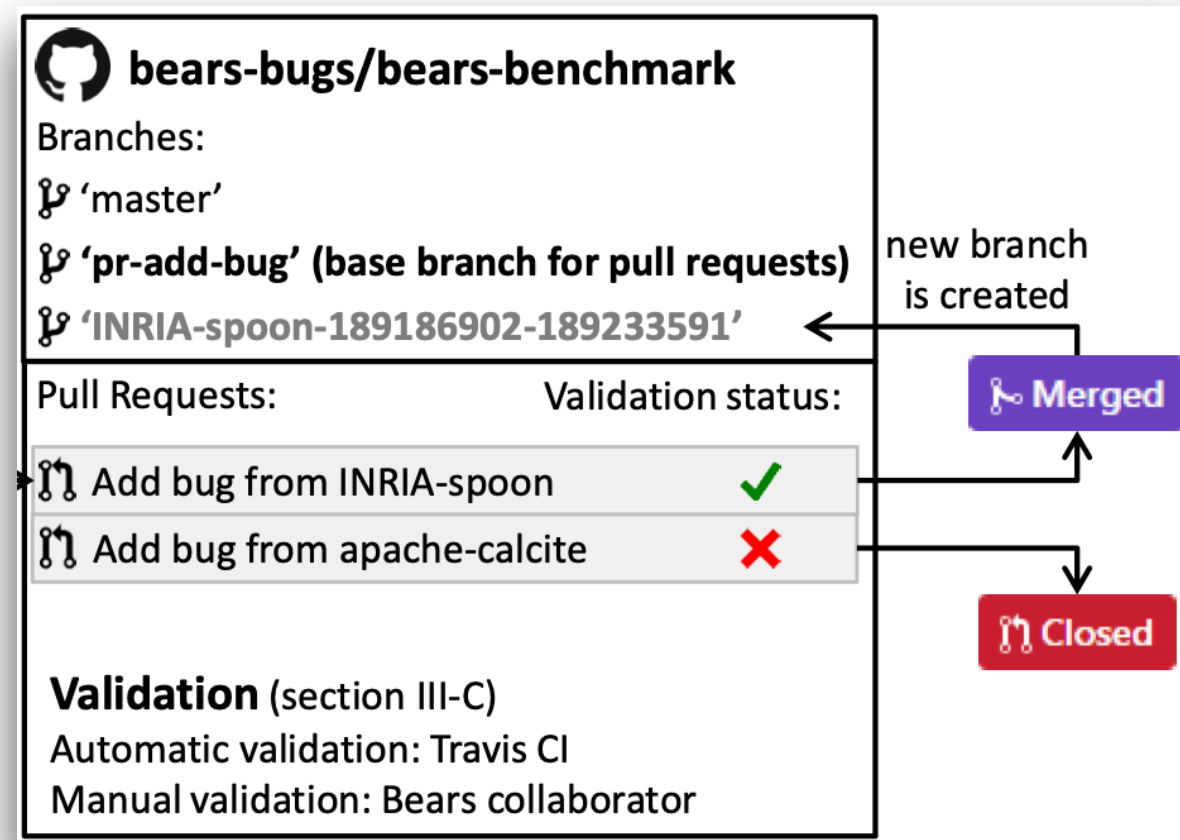
The screenshot shows a GitHub pull request page for the repository 'bears-bugs/bears-benchmark'. The 'Branches' section lists 'master' and 'pr-add-bug' (base branch for pull requests), with the latter highlighted in red. Below, the 'Pull Requests' section shows two requests: 'Add bug from INRIA-spoon' (status: green checkmark) and 'Add bug from apache-calcite' (status: red X). A 'Validation status' section at the bottom indicates 'Automatic validation: Travis CI' and 'Manual validation: Bears collaborator'. An arrow labeled 'pull request' points to the first request.

Pull Requests:	Validation status:
🔗 Add bug from INRIA-spoon	✓
🔗 Add bug from apache-calcite	✗

Validation (section III-C)
Automatic validation: Travis CI
Manual validation: Bears collaborator

BEARS-COLLECTOR

- プロセス 5 : Merged or Closed
 - * プルリクエストが検証を通過すると, Mergeされ BEARS-BENCHMARKに新しいブランチが作成される
 - * 通過しなかった場合はMergeされずClosedとなる



検証：BEARS-COLLECTORの実行

- 3 ラウンドに分割

- * ラウンド 1 (2017年01月~12月)

- ▶ GHTorrentプロジェクトから4プロジェクトを選定してスキャン

- * ラウンド 2 (2018年01月~04月)

- ▶ [2]のプロジェクトから100プロジェクトを選定してスキャン

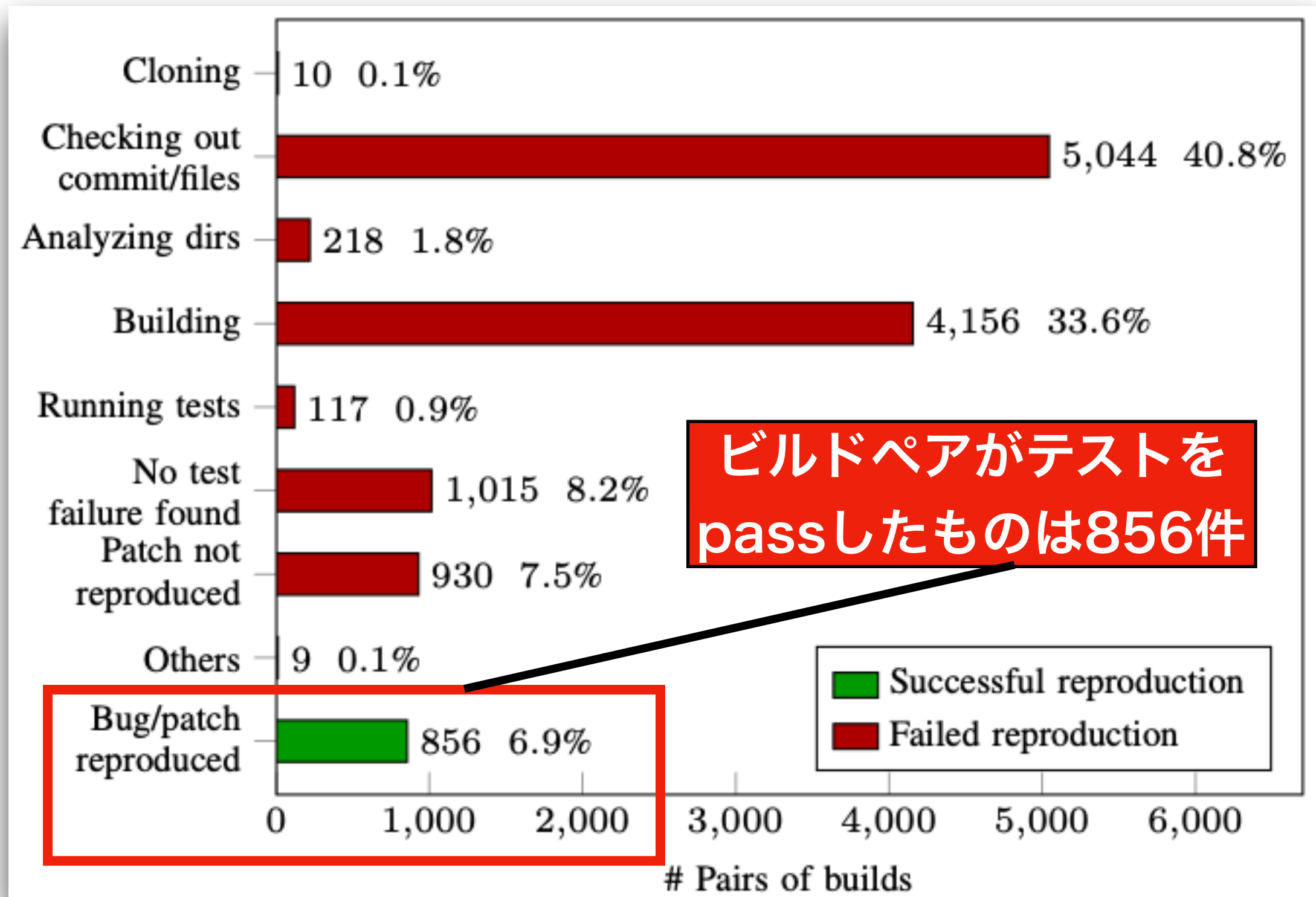
- * ラウンド 3 (2018年09月19日~09月24日, 09月30日~11月30日)

- ▶ ビルドのペアをリアルタイムにスキャン

スキャン結果

	Execution Round			Total
	#1	#2	#3	
Period	1 year	4 months	~2 months	
# Input projects	4	100	–	
# Total scanned builds	4,987	66,621	97,164	168,772
# Build pairs in case #1	17	590	134	741
# Build pairs in case #2	1,027	7,755	2,832	11,614
# Total build pairs	1,044	8,345	2,966	12,355

検証：BEARS-COLLECTORの実行



検証：BEARS-COLLECTORの実行

- 856件(72プロジェクト)の自動検証と手動検証の結果
 - * 手動検証：295件のブランチが有効とみなされた
 - * 自動検証：295件のブランチのうち251件が有効とみなされた
 - ▶ ケース1のブランチ：19件
 - ▶ ケース2のブランチ：232件

BEARS-BENCHMARKに格納されたペア

- BEARS-BENCHMARKに格納されたビルドペアが多かったプロジェクトTop10
 - * プロジェクト期間, システムサイズ(LOC), テスト数全てで違いが見られる
- ▶ **BEARS-BENCHMARKのバグが多様なプロジェクトから収集できたことを示す**

Name	Project		Creation	GitHub info ^a		BEARS info ^b		
	Type	Domain/Topics		#Contrib.	#Commits	LOC	#Tests	#Bugs
INRIA/spoon	Lib	Java code analysis and transformation	2013/11	47	2,804	76,295	1,114	62
traccar/traccar	App	GPS tracking system	2012/04	87	5,416	43,397	255	42
FasterXML/jackson-databind	Lib	general data binding (e.g. JSON)	2011/12	141	5,038	99,151	1,711	26
spring-projects/spring-data-commons	Lib	spring data, data access	2010/11	62	1,793	36,479	2,029	15
debezium/debezium	App	platform for change data capture	2016/01	69	1,439	53,318	508	7
raphw/byte-buddy	Lib	runtime code generation for the JVM	2013/11	39	4,640	140,087	8,066	5
SzFMV2018-Tavasz/AutomatedCar	App	passenger vehicle behavior simulator	2018/01	33	862	2,197	48	2
rafonsecad/cash-count	App	accounting software back-end	2018/09	1	29	759	16	2
Activiti/Activiti	App	business process management	2012/09	160	8,041	205,097	1,952	1
pippo-java/pippo	Lib	micro Java web framework	2014/10	22	1,341	19,738	131	1

^a The number of contributors and the number of commits were collected in December 21, 2018.

^b The metrics LOC and number of tests are averaged over the collected bugs.

関連研究

- Defect4J[3], Bugs.jar[4]
 - * Defect4J : 6 プロジェクト, 395件の再現可能なバグを格納
 - * Bugs.jar : 8 プロジェクト, 1,158件の再現可能なバグを格納
- BEARSの方が優れている点

大規模プロジェクト以外の多様なプロジェクトからバグ収集するように設計されている

Defect4J, Bugs.jarはバグ管理システムJIRAからデータ収集しているが, 実際にプロジェクトをビルドおよびテストする方が再現可能なバグを特定できる

[3]R. Just, D. Jalali, and M. D. Ernst, "Defects4J: A Database of Existing Faults to Enable Controlled Testing Studies for Java Programs," in Proceedings of the 23rd International Symposium on Software Testing and Analysis (ISSTA '14). New York, NY, USA: ACM, 2014, pp. 437–440.

[4]R. K. Saha, Y. Lyu, W. Lam, H. Yoshida, and M. R. Prasad, "Bugs.jar: A Large-scale, Diverse Dataset of Real-world Java Bugs," in Proceedings of the 15th International Conference on Mining Software Repositories (MSR '18). New York, NY, USA: ACM, 2018, pp. 10–13.

まとめと所感

- 自動バグ修正の研究を支援するために、バグを収集しベンチマークに格納するプロジェクトを提案

* 最終的に72のプロジェクトから251件のバグを収集できた

▶ バグデータの多様性が示された

- 所感

BEARS-BENCHMARK <https://github.com/bears-bugs/bears-benchmark>
BEARS-COLLECTOR <https://github.com/bears-bugs/bears-collector>
Bug/patch browser <https://bears-bugs.github.io/bears-benchmark>
Data from Section IV <https://github.com/bears-bugs/saner2019-data>



* 手動で検証するフェーズが大変そう….

<各データのリンク>