

Codexを用いた Coverage-Guided pytest 自動生成システムの開発と評価

SSE09-12

1Finity株式会社

徐広勝

CoverUp Codex

開発における問題点

背景・課題

LLMでテストは生成できるが、実プロジェクトでCoverageを継続的に上げるのは難しい。

解く課題

Coverage状態から次の対象を自動選択し、改善を反復する仕組みが必要。

Coverage改善を反復する自動化が必要

手法・ツールの適用による解決

提案

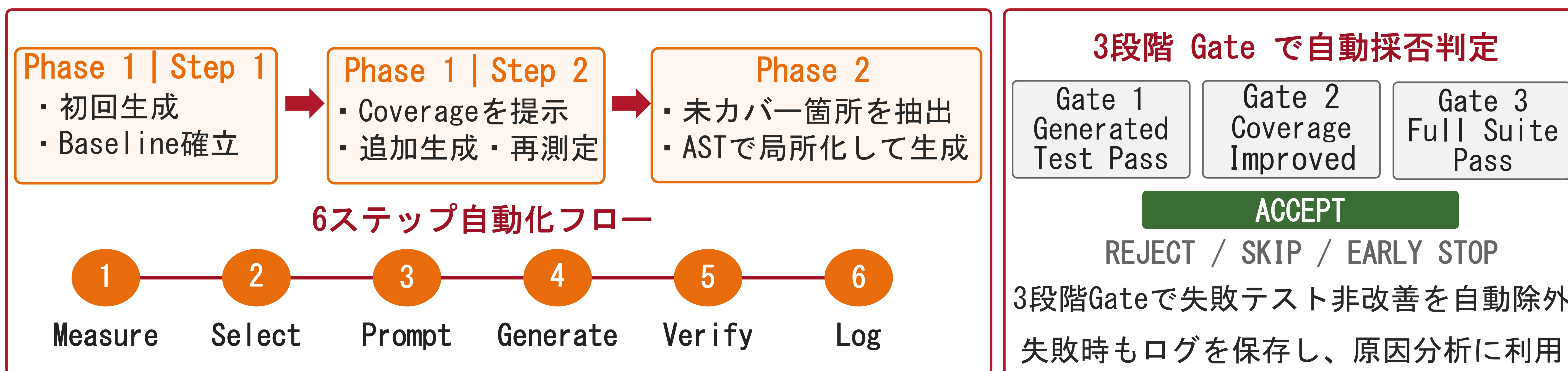
Coverageに応じて対象を選択し、生成→実行→測定→採否→記録を自動化。

主要技術

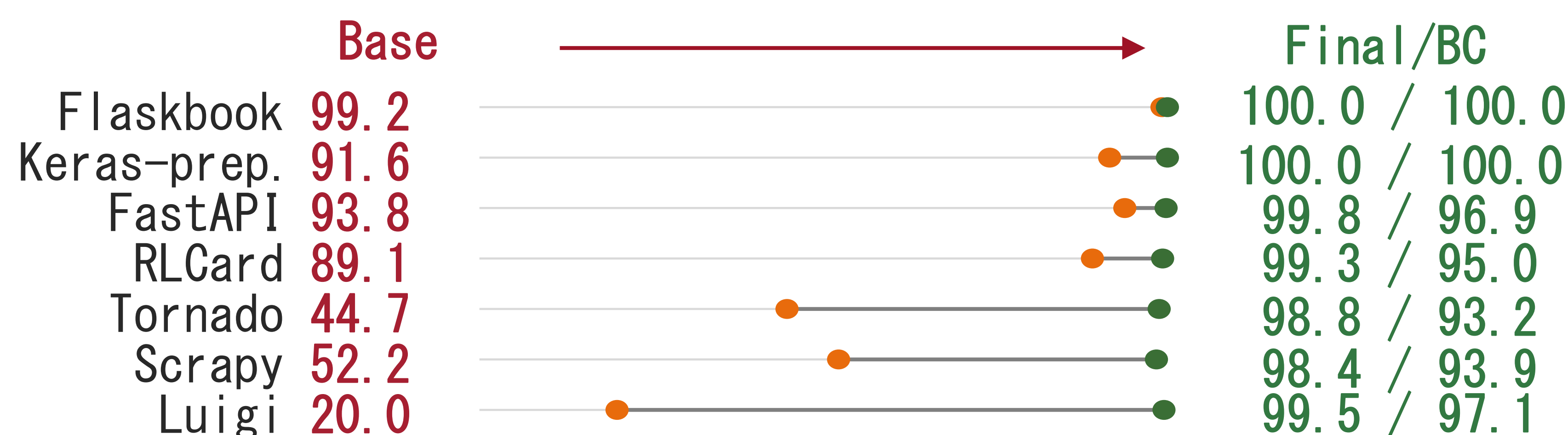
Coverage差分 / AST局所化 / 3段階Gate / 実験ログ

提案システム：Coverage改善の自動オーケストレーター

低Coverageでは広く、高Coverageでは局所化し、対象選択・生成・実行・採否・記録を1コマンドで制御



実験結果：7つのPython OSSでLine Coverage 98%以上



RQ1 | 有効性 7 OSSすべてでLC 98%以上
RQ2 | 自動化 生成/採否/記録を1コマンド化
RQ3 | 適用可能性 規模構造の異なるOSSで改善

最大の貢献：高Coverageへ至る改善プロセスを再現・追跡可能にしたこと

7/7
LC 98%以上

+46.2pt
Scrapy
52.2→98.4%

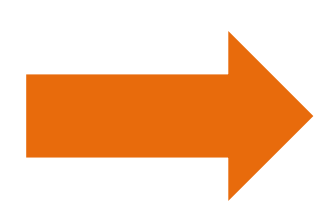
+54.1pt
Tornado
44.7→98.8%

+79.5pt
Luigi
20.0→99.5%

貢献と今後の課題

Before | 手動・半手動

対象選択・実行・採否・記録を人が反復



反復が増えるほど手動作業が増加

After | CoverUp Codex

対象選択～採否・記録を自動化
同一条件で再現・追跡可能

NEXT : Coverage × 時間 × Tokenで終了判断