4L-04

JavaScript 言語における機能等価な異なるソースコード断片の利用実態調査

野口 隼杜† 伊原 彰紀‡ 大森 楓己§
和歌山大学† 和歌山大学‡ 和歌山大学§

1 はじめに

ソフトウェア開発において，ソフトウェアの品質を向上させるために，開発者は機能的に等価な異なるソースコードへの書き換えを行うことがある．このような行為はリファクタリングと呼ばれ，ソフトウェアの保守性や性能効率性などの品質を高めることが期待される．

リファクタリングにおいて，開発者はソフトウェアの特性や自身の実装経験に基づいて書き換えを検討する．従来研究[1]では，リファクタリングがソフトウェアの実行速度に与える影響を明らかにしている．また，従来研究[2]では，実行速度を向上させる自動リファクタリング手法が提案されている．しかし，書き換えられたソースコードが，ソフトウェア全体の保守性や性能効率性などの向上を，必ずしも実現させる実装方法であるとは限らない．

そこで本研究では，JavaScript における異なるソースコードの実行速度を比較するマイクロベンチマーク共有サービスである JsPerf※1 に保存されたソースコード断片のペアを利用し，頻繁に検証される機能的に等価なソースコード断片のペアを収集する．これをもとに，多くの開発者が採用している実装方法を調査し，採用されるソースコード断片の実態と選択理由について実行速度の視点で考察する．

2 機能的に等価なソースコード断片のペア

2-1 マイクロベンチマーク

マイクロベンチマークは，小さな規模のクラスやメソッド単位のソースコードの実行速度を評価するベンチマークであり，同じ入出力でありながら，異なる方法で実装される 2 つ以上のソースコードの実行速度を比較するものである．

表 1：調査対象となるソースコード断片のペア

	slow	fast
①	new Date().getTime()	Date.now()
②	Object.prototype.hasOwnProperty.call(*)	.hasOwnProperty(*)

*は任意の文字列

2-2 調査対象

本研究では，従来研究[2]によって作成された，JsPerf に保存されたソースコード断片のペアから意味的等価性および実行速度の有意差が評価されたソースコード断片のペアを利用する．評価回数の多い 2 件のペアを調査対象とし，表 1 に示す．

3 手法

3-1 データセット

GitHub において主に JavaScript を使用したリポジトリのうち，スター数上位 100 件のリポジトリを本研究の調査対象データセットとする．

3-2 採用されたソースコード断片の調査

データセット中の JavaScript ファイルについて，リポジトリごとに，表 1 に示したソースコード断片の出現回数を集計する．このとき，調査対象であるソースコード断片から正規表現を作成し，一致した回数を出現回数とする．

3-3 特徴分析

3-1 節で収集したリポジトリについて，Node.js のパッケージ管理システムである npm，ライブラリ検索エンジンである libraries.io，および GitHub からキーワードとトピックを収集し，各リポジトリのもつキーワードとして，3-2 節で調査した各リポジトリのソースコードの構成実態と合わせて考察する．収集した各リポジトリのキーワードおよびトピックの例を表 2 に示す．

An Exploratory study of different functionally equivalent source code fragments in the JavaScript

† Hayato Noguchi, Wakayama University
‡ Akinori Ihara, Wakayama University
§ Fuki Omori, Wakayama University

※1 JsPerf : https://jsperf.app/

Copyright ©2025 Information Processing Society of Japan.
All Rights Reserved.

表2：リポジトリのもつキーワード例

リポジトリ名	キーワード
react	declarative, ui frontend, react, javascript, library

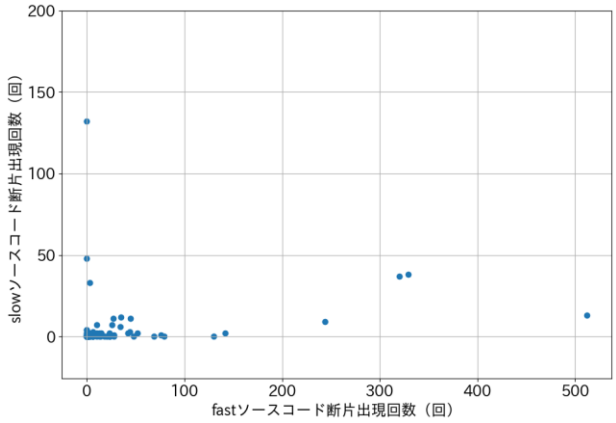

図1：リポジトリ別パターン①出現回数

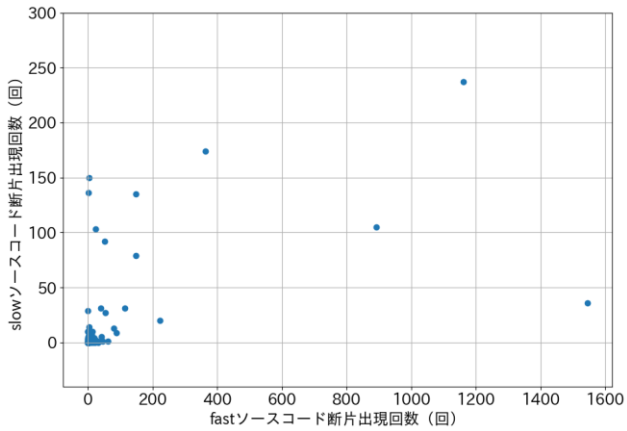

図2：リポジトリ別パターン②出現回数

4 結果と考察

図1，図2に表1の①，②のソースコード断片のリポジトリ別出現回数をそれぞれ示す．横軸が fast ソースコード断片の出現回数，縦軸が slow ソースコード断片の出現回数である．

fast ソースコード断片を多く使用するリポジトリのキーワード・トピックを目視で調査したところ，“backend”や“server-rendering”といった，実行速度が重視されるリポジトリであると考えられるキーワードが見られた．また，fast ソースコード断片を多く含むリポジトリの中でも，“frontend”をキーワードとして含む，フロントエンドに関連するリポジトリよりも，“backend”をキーワードとして含む，バックエンドに関連するリポジトリで，より多くの fast ソースコード断片を利用していることが確認された．これは，フロントエンドとバックエンドの実行速度への要求を比較すると，一般的にバックエンドの方が，数行のコードの極めて小さな実行速度を追求するため，適切に実装方法の特徴を示していると考えられる．一方で，slow ソースコード断片を多く使用するリポジトリや，fast ソースコード断片と slow ソースコード断片の出現回数に差が見られなかったリポジトリでは，“css”や“UI”のような，実行速度に大きく影響する要素に関するキーワードは含まれていなかった．

したがって，一部のリポジトリの開発者は，実行速度の観点において，その性質に合わせた実装方法を採用していることが示唆される．

5 おわりに

本研究では，マイクロベンチマーク共有サービスである JsPerf のもつソースコード断片のペアから抽出した，意味的等価性および実行速度の有意差が評価されたソースコード断片のペアの一部を利用し，GitHub において主に JavaScript を使用したリポジトリのうち，スター数上位 100 件のリポジトリ中のソースコード断片の利用実態の調査を行った．

その結果，実行速度が重視される要素をキーワードとして持つリポジトリにおいて，実行速度の大きいソースコード断片が利用されていることを確認した．また，リポジトリの特徴が，実行速度の大きいソースコード断片をどれだけ利用するかに影響する可能性が示唆された．

今後の課題として，各リポジトリにおいて実装を行った開発者の特性に依存している可能性の考慮や，ソースコード断片が利用されている箇所の，動作速度への影響度の大きさを考慮することなどが挙げられる．より詳細な分析を通じて，ソースコード断片が選択された要因の解明を目指す．

参考文献

[1] Luca Traini et al, “How Software Refactoring Impacts Execution Time” in Proceedings of the ACM Transactions on Software Engineering and Methodology (TOSEM), Volume 31, Issue 2, 2021, pp.1-32

[2] 大森楓己，伊原彰紀，柏祐太郎．“マイクロベンチマーク共有サービスを用いた実行高速化のための自動リファクタリングへの試み”，情報処理学会 ソフトウェア工学研究会 Vol.2024-SE-218 No.8，2024

Copyright ©2025 Information Processing Society of Japan.
All Rights Reserved.