



Title	Analyzing Code Compatibility, Equivalence, and Similarity to Support Software Maintenance and Evolution
Author(s)	楊, 詩語
Citation	大阪大学, 2026, 博士論文
Version Type	VoR
URL	https://doi.org/10.18910/105497
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

<https://ir.library.osaka-u.ac.jp/>

The University of Osaka

論文内容の要旨

氏 名 （ 楊 詩語 ）	
論文題名	Analyzing Code Compatibility, Equivalence, and Similarity to Support Software Maintenance and Evolution (ソフトウェア保守および進化を支援するためのコード互換性・等価性・類似性の分析)
論文内容の要旨 Software has become a fundamental component of modern society, supporting critical infrastructure, industries, and everyday life. As software systems continue to expand in scale and complexity, maintaining and evolving them over time while ensuring reliability and adaptability has become a central challenge. In this context, developers increasingly rely on reusing existing code from open repositories and online platforms such as Stack Overflow. While such reuse accelerates development, it also introduces new risks, particularly when the reused code is incompatible with the target environment or semantically misunderstood. Addressing these challenges requires a deeper understanding of how compatibility, functional equivalence, and structural similarity interact during software evolution. This dissertation systematically investigates how these three dimensions, compatibility, equivalence, and similarity, can be jointly analyzed to support sustainable software maintenance and evolution. The research first examines the reliability of reused online code, focusing on Python: a widely used programming language whose major versions are intentionally not backward compatible. An empirical study of Python code snippets collected from Stack Overflow demonstrates that a substantial portion of widely referenced examples fail to execute due to version incompatibilities between Python 2 and Python 3. This finding exposes hidden maintenance risks in online code reuse and underscores the critical need for automated version-aware analysis. Building upon this empirical necessity, the dissertation introduces the browser-based tool PyVerDetector. It automatically detects the Python version compatibility of reused Stack Overflow code snippets in real time. By leveraging multiple Abstract Syntax Tree (AST)-based grammars, the tool identifies version-specific syntax issues and visualizes the compatible Python versions directly within the browser. Evaluation results indicate that PyVerDetector achieves accuracy comparable to existing offline analyzers while still providing lightweight, immediate feedback to developers. This demonstrates that automated compatibility checking can effectively prevent subtle version-related faults during code reuse. Building on the insights from compatibility analysis, the dissertation extends its focus to code equivalence and similarity, exploring whether syntactically different implementations can exhibit identical functional behavior. To this end, the PyFuncEquivDataset, a representative dataset of Python functions, was systematically constructed from GitHub, encompassing diverse projects and implementation styles. Through automated test generation, mutual execution, and manual validation, functionally equivalent pairs were identified and analyzed. The results reveal that equivalent functions often differ substantially in structure, performance, and readability, suggesting that equivalence detection can facilitate refactoring, redundancy reduction, and performance optimization. Furthermore, by evaluating large language models (LLMs) on code equivalence and semantic similarity recognition, this study provides new insights into the emerging role of artificial intelligence in understanding software semantics. Taken together, this research establishes a unified research framework that connects code compatibility, equivalence, and similarity as interdependent dimensions of software evolution. The dissertation contributes (1) empirical evidence of real-world compatibility issues in reused code, (2) a practical browser-based tool for real-time version-aware maintenance, and (3) a representative dataset and analytical methods for identifying functionally equivalent and similar code fragments. Collectively, these outcomes advance the understanding of software reuse and lay the foundation for intelligent, automated approaches that support reliable maintenance and sustainable evolution of large-scale software systems.	

論文審査の結果の要旨及び担当者

氏 名 (楊 詩 語)			
論文審査担当者	(職)		氏 名
	主 査	教授	肥後 芳樹
	副 査	教授	楠本 真二
	副 査	教授	土屋 達弘

論文審査の結果の要旨

ソフトウェアは現代社会における基盤的な構成要素であり、重要インフラ、産業活動、および日常生活を支えている。ソフトウェアシステムが規模・複雑性の両面で拡大を続ける中、信頼性と環境適応性を確保しつつ、長期にわたって保守・進化させることが中心的な課題となっている。このような状況において、開発者はオープンリポジトリや Stack Overflow などのオンラインプラットフォームから既存コードを再利用することにますます依存している。この再利用は開発を加速させる一方で、再利用したコードが対象環境と互換性を持たない場合や、意味的に誤解された場合には、新たなリスクをもたらす。これらの課題に対処するためには、ソフトウェア進化の過程において、互換性、機能的等価性、および構造的類似性がどのように相互作用するのかを、深く理解する必要がある。

本博士論文では、これら三つの要素（互換性、等価性、および類似性）を統合的に分析することで、持続可能なソフトウェア保守・進化を支援する方法を体系的に検討する。まず、オンラインで再利用されるコードの信頼性に着目し、主要バージョン間で後方互換性を意図的に持たない、広く利用されているプログラミング言語である Python を対象として分析を行った。Stack Overflow から収集した Python コードスニペットに対する実証的調査により、広く参照されているコード例の相当部分が、Python2 と Python3 のバージョン非互換性に起因して実行に失敗することが明らかになった。この結果は、オンラインコード再利用に潜む保守上のリスクを顕在化させるとともに、バージョンを考慮した自動解析の重要性を強く示している。この実証的知見を踏まえ、本論文ではブラウザベースのツール PyVerDetector を提案した。このツールは、再利用される Stack Overflow の Python コードスニペットに対して、Python のバージョン互換性をリアルタイムに自動検出する。評価結果から、PyVerDetector は既存のオフライン解析ツールと同等の精度を達成しつつ、軽量で即時的なフィードバックを開発者に提供できることが示された。これは、自動的な互換性検査が、コード再利用時に生じる微妙なバージョン関連の欠陥を効果的に防止できることを示した。

本論文では、互換性解析から得られた知見を基に、コードの等価性および類似性についても取り組んだ。具体的には、自動テスト生成、相互実行、および手動検証を通じて、機能等価な関数の組を特定・分析した。その結果、等価な関数であっても、構造、性能、可読性において大きく異なる場合が多いことが明らかとなり、等価性検出がリファクタリング、冗長性削減、および性能最適化に寄与し得ることが示唆された。また、特定した関数の組からデータセットを作成した。さらに、大規模言語モデル（LLM）を対象として、コードの等価性および意味的類似性の認識能力を評価することで、ソフトウェア意味理解における人工知能の新たな役割についての知見を提供した。

以上を総合すると、本論文は、コードの互換性、等価性、および類似性を、ソフトウェア進化における相互依存的な次元として結び付ける統一的な研究枠組みを確立した。本博士論文の貢献は、（1）再利用コードにおける実世界の互換性問題に関する実証的証拠の提示、（2）リアルタイムなバージョン考慮型保守を実現する実用的なブラウザベースツールの提案、（3）機能的に等価または類似したコード断片を同定するための代表的データセットと分析手法の提供、の三点にまとめられる。これらの成果は、ソフトウェア再利用に対する理解を深化させ、大規模ソフトウェアシステムの信頼性の高い保守と持続的な進化を支援する知的かつ自動化されたアプローチの基盤を築くものである。よって本論文は、博士（情報科学）の学位論文として価値のあるものと認める。