静的型付けの意味における健全性に焦点を当てた TypeScript 型検査機

糸川　倫太朗　青山研究室

A TypeScript type checker with focus on soundness in the context of static typing

ITOKAWA Rintaro

1. 背景

1.1. TypeScript

TypeScript は，Microsoft によって開発された，ECMAScript 5 を拡張したプログラミング言語である．TypeScript の特徴として，型アノテーションや型エイリアス，関数オーバーロードのサポートが挙げられる．これにより，静的型検査を通じてプログラムの誤りを検出できる．一方で，`any` 型の存在により，型チェックを回避することも可能である．`any` 型は，型アノテーションを省略可能にし，JavaScript からの移行を容易にする役割を果たす．この柔軟性により，TypeScript は型検査の範囲を段階的に広げつつ利用できるデザインとなっている．

1.2. Gradual Typing

Gradual Typing 1) は，Siek と Taha によって 2006 年に提案された，静的型付けと動的型付けを融合させる手法である．この手法では，プログラム内で静的検査を適用する箇所をプログラマが選択できる．? 型が導入され，静的検査を回避するためのアノテーションとして機能する点で，TypeScript の `any` 型に類似している．

1.3. Criteria for Gradual Typing

Gradual typing という概念は Siek ら 2) の 2015 年の論文で整理され，以下の 5 つの条件を満たすべきと定義された．2006 年のオリジナル体系はこれらの条件を全て満たしている．

1. 静的型付けの内包: 型アノテーションが完全に付与されたプログラムは，通常の静的型付けシステムと同じ挙動を示す．
2. 動的型付けの内包: 型アノテーションが全て?型であるプログラムは，動的型付けシステムと同じ挙動を示す．
3. 健全性: 型エラーは必ずランタイムで検出可能である．
4. Blame-Subtyping Theorem: $T_1 <: T_2$ であれば，T_1 から T_2 へのキャストはランタイム型エラーを引き起こさない．
5. Gradual Guarantee: 静的型検査に合格したプログラムの型アノテーションを減らしても検査は成功し，動作も変わらない．

これらの条件により，静的型付けと動的型付けを柔軟に組み合わせるシステムが明確に定義されている．

1.4. TypeScript と Criteria for Gradual Typing

TypeScript の型システムを Siek らの基準に照らして検討したところ，TypeScript は gradual typing の意味での健全性を失っており，ランタイム型チェックを行わない言語設計がその要因であることが明らかとなった．

1.5. Safe TypeScript とその課題

Safe TypeScript は，TypeScript の gradual typing における健全性の欠如に対応するため Microsoft Research によって開発された．Safe TypeScript ではランタイム型チェックのオーバーヘッド削減が試みられ，約 15% に抑えられたが，それでも実用性に問題がある．特に現代の Web 開発では，ランタイムの負荷がユーザビリティに大きな影響を与えるため，Safe TypeScript は現実的な選択肢とは言えない．

2. 目的

TypeScript の柔軟性と静的型検査による安全性を両立させることを目的とし，新しい型検査ツール「decaf」を提案する．これは，開発者が求める「省略された型アノテーションを補完した上での静的型検査」という柔軟性を実現し，ランタイムのオーバーヘッドを削減しつつ，健全性を保つことを目指している．

3. decaf の特徴

4. 実験

5. 結果と考察

6. まとめ

7. 参考文献

1) Robin Milner. A theory of type polymorphism in programming. Journal of Computer and System Sciences, Vol. 17, No. 3, pp. 348-375, 19

2) Jeremy G. Siek, Michael M. Vitousek, Matteo Cimini, and John Tang Boyland. Re-fined Criteria for Gradual Typing. In Thomas Ball, Rastislav Bodík, Shriram Krishna-murthi, Benjamin S. Lerner, and Greg Morriset, editors, 1st Summit on Advances in Programming Languages (SNAPL 2015), Vol. 32 of Leibniz International Proceedings in Informatics (LIPIcs), pp. 274-293, Dagstuhl, Germany, 2015. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.