令和 6 年度電子情報工学科　卒業研究中間報告

報告者	06席	氏名	**糸川 倫太朗**	研究室	**青山研究室**
論文題目	**A JavaScript compiler and TypeScript checker with a focus on static analysis and runtime performance**				
キーワード	**JavaScript, TypeScript, compiler, static analysis**				

＜背景＞AltJS として寡占的な立ち位置を獲得した TypeScript という言語がある．その根幹となる TypeScript を JavaScript に変換するコンパイラ（tsc）は，静的解析機や language server に使用されてきた．しかし，そんな tsc にはある問題がある．それは，tsc による TypeScript の解析と型推論は非常に計算コストと時間コストが高いことである．これにはいくつかの解決策があるが，どれも tsc に依存してしまうため，JavaScript ランタイムの起動から実行までのボトルネックを無視できない．そこで，本研究では tsc の代替となるパフォーマンスを意識したコンパイラを作成することにした．

＜目的＞language server や静的解析機で使用した際，tsc と比較して，高速でハイパフォーマンスなコンパイラを作成する．

＜研究の概要＞「decaf」と名付けた本研究は Rust を使用して開発する．TypeScript のパースは oxc_parser を参考に実装する．型推論は tsc を参考に実装する．

tsc のボトルネックは大きく分けて，TypeScript のパースとその型検査である．TypeScript のパースについては，Rust で一からパーサを実装することである程度の速度改善が見られると期待した．一方で，型検査については調査の過程で分かったこととして，長年の実装の積み重ねの末にアルゴリズムは複雑化しており，メンテナンス性も低くレガシーなアルゴリズムによって動いていることが分かった．そこで，decaf では tsc と異なった型検査のアーキテクチャを提案する．

decaf では，TypeScript の構文解析木（AST) を decaf の型検査機が解釈できる形（TypeID）に変換する．これにより，型検査に他の言語で実装されているモダンなアルゴリズムを適用できるようになる．この機構を便宜上「変換機」と呼称する．

decaf では変換機は，値と構造に関する情報を型として保持し，関数やブロックの振る舞いをイベントとしてエンコードする．型やイベントは，AST の走査中に生成され，型検査機に渡され使用される．

decaf の型検査機では型を「その値が取りうる可能性の空間」として捉える．decaf では条件分岐やループに差し掛かった時，そのブロック内で取りうる型の空間を広げる．AST を走査しながら型の空間を広げていくことで，実行時に起こりうるすべての可能性を型チェック時に考慮できる．

＜研究の現状＞現在， decaf は TypeScript のほぼ全ての構文をパースできる．

一方で型検査は概要で述べたアーキテクチャに基づいて実装中である．

＜今後の方針＞サポートする構文を増やし，型推論の精度を向上させる．また，プレイグラウンドを作成し，フィードバックを受けやすい環境を整える．