RustOwl: Rustにおける所有権とライフタイムの可視化

岡本 祐希　篠埜 功

芝浦工業大学 プログラミング言語研究室

研究背景

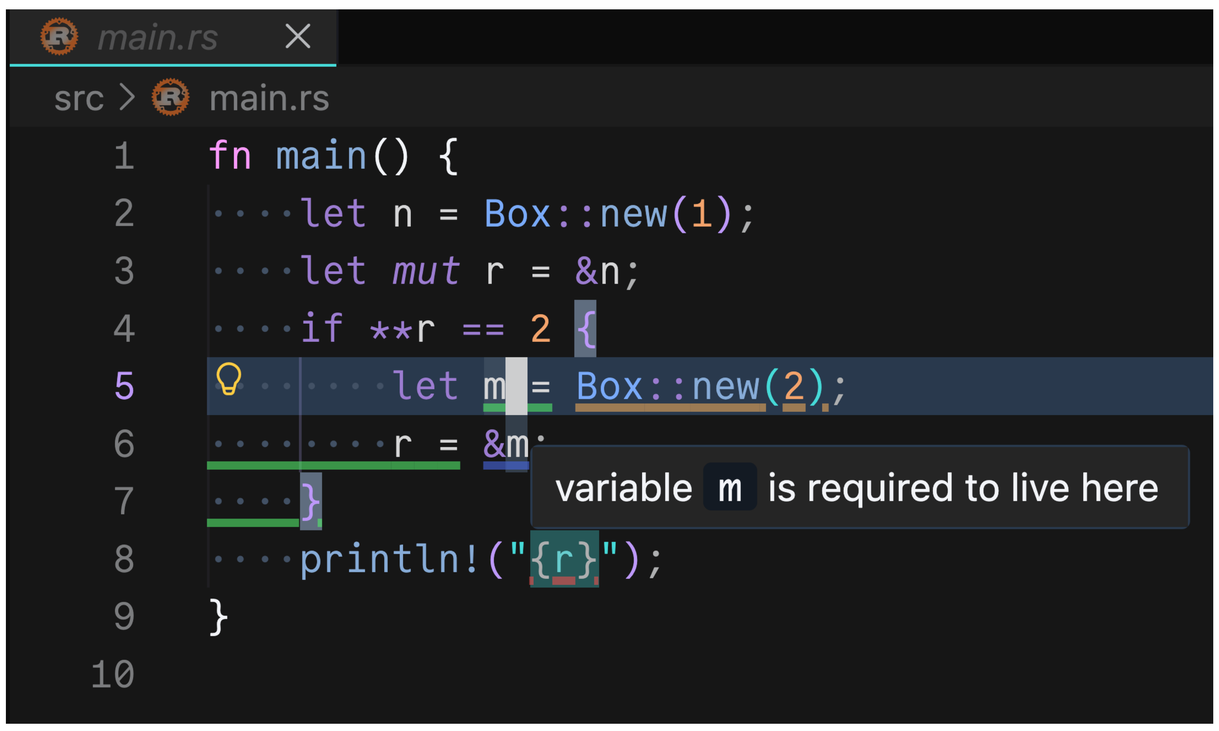
Rustでは所有権とライフタイムを導入
→ 効率良く、メモリ安全でデータ競合が起きない。

これらは誤った理解をされることが多い
→ コンパイルエラーや効率の悪いプログラムが記述される。

変数のライフタイムはデッドロックといったバグやリソースの解放にも関連。

提案および実装

所有権とライフタイムを色付き下線で可視化するツールRustOwlを実装。



RustOwlのVS Code拡張を用いている例

色の意味は、ローカル変数についてそれぞれ

- 緑色：有効範囲
- 青色：不変借用する式の範囲
- 紫色：可変借用する式の範囲
- 橙色：オブジェクトがムーブしている式の範囲
- 赤色：有効であるべき範囲と有効範囲との差分

また、色は設定で変更できる。

コンパイラ拡張、LSPサーバをRustで、VS Code拡張をTypeScriptで記述。Neovim、Emacsにも対応。

`https://github.com/cordx56/rustowl`

アルゴリズム

借用検査を行うライブラリPoloniusの出力を用いる。中間言語MIR中の各ローカル変数について、

- 有効範囲：変数に代入されてからムーブ（ドロップを含む）されるまでの範囲
- 借用、ムーブ：変数の宣言を集め、それらがMIRでそれぞれの操作をされる範囲
- エラー：変数の有効範囲と変数が有効であるべき範囲の差分

を計算して求める。

変数mへの直接、または間接的な参照について、それぞれの型に含まれるライフタイム変数が表す範囲を「変数mが有効であるべき範囲」とし、ライフタイム変数間の制約からこれを計算する。

評価

72名のプログラミング経験者に理解度を問う調査を実施。既存の可視化ツールAquascopeに比べ、有意水準5%の両側検定で、正答率が有意に向上。Rust習熟度別では中級者において有意に向上。学習効果が初学者に限らないことを示した。

ツールごとの正答率 (%)、Welchのt検定のp値

習熟度	Aquascope	RustOwl	p
First time	50.0	79.2	0.126
Beginner	77.5	80.8	0.778
Intermediate	67.3	93.2	0.037
Proficient	75.0	93.8	0.400
Expert	–	87.5	–
全体	67.4	86.1	0.010

主観評価では、Aquascopeに比べてRustOwlがメモリ最適化に有効であると評価された。

将来課題

- アルゴリズムの妥当性
- 実用的なRustコードでの評価
- RustRoverの可視化と比較