# 実例の分析結果に基づくコンパイルエラーに対する改善案の提示手法

情報科学部情報科学科　小林　健太　　　指導教員：山本　晋一郎

## 1　はじめに

近年, ソフトウェアの大規模化, 複雑化が進んでいる. それに伴い参照透明性, 高階関数や型推論などによりコードを比較的短く記述することが可能であり, 関数合成により関数の拡張が比較的容易である関数型言語が注目されつつある. そのため関数型言語を学ぶ人が増加している. そこで関数型言語 Haskell に対する学習支援を考える. 初学者の Haskell 学習を難しくする要因には, 厳格なインデント規則, 厳密な型システム, 主流な Haskell コンパイラである GHC によるエラーメッセージの難解さ, が挙げられる. 本論文では, これらのうち, GHC によるエラーメッセージの難解さに着目し, その緩和方法を提案する.

本論文のアプローチは, 本学で開講している「プログラミング入門」での学生の演習問題に対する解答とそれに対する GHC のエラーメッセージを検証し, 初学者にとってより理解しやすい日本語の解説文をエラーメッセージの種類に対応させて作成ことである。

次にその結果を用いて, エラーメッセージを入力し解説文を提示するシステムを Web アプリとして作成する. このシステムにより初学者は難解な GHC のエラーメッセージでなく, より適切な日本語の解説文を得られるため, 言語学習の支援が可能となる.

## 2　手法

本手法ではエラーメッセージをエラー種類ごとに分類し, 各エラーメッセージに対する解説文を作成, それを提示することで支援を行う.

### 2.1　エラーメッセージ分類及び解析

エラーの種類の分類は, 学生のデータからエラーの種類を示すキーワードを決定し, そのキーワードを基に行う. 次にエラーメッセージをエラーの種類に対応する構文解析をすることで解説文の作成に必要な変数名, 関数名などの情報をエラーメッセージより抽出し, 解説文を生成した.

GHC が出力するエラーメッセージは, ファイル名, エラーの行位置と列位置, エラーの内容という形式を持つ. その例を以下に示す.

リスト 1　GHC エラーメッセージ

```
test.hs:1:21:
  Syntax error on ’div’
  Perhaps you intended to use -XTemplateHaskell
```

エラーの内容部において, メッセージの序盤はエラーの大まかな原因が表示される. 例えば上記のメッセージでは Syntax error を分類のキーワードとする. 今回, 集めた学生の解答データは 37 種類に分類することができた. GHC によって出力されるメッセージはエラーの種類ごとで似通った構文規則を持つため, それにあった解析器をエラーの種類ごとに作成することで必要な情報を取り出すことができる. 現在, 対応している 37 種類のエラー出力のうち, メッセージ内にプログラム改善に使用可能な情報を含むものに関しては対応が可能である.

### 2.2　メッセージ作成

解説文は学生の解答での実際の誤り原因と GHC のエラーメッセージを比較した分析結果を基に, より初学者が分かりやすいように作成した.

実際に本手法を用いた例として, 提案ツールによりリスト 1 のエラーメッセージを変換すると以下の文章が出力される.

エラーメッセージ変換後

**test.hs,1 行,21 文字目**
**div を中置関数として使用するには**
**バッククォートで囲んで記述してください。**

リスト 1 の種類のエラーメッセージが出力されたときの学生の解答データを検証した結果, エラーの原因が上記メッセージの内容に限られるという事例に基づき説明文を作成した. div というキーワードはエラーメッセージから抽出した. 上記の例のように、GHC が出力する内容とは大きく異なるが, より原因となるものを過去の事例を参考に示した. また, エラーメッセージより抽出した情報を用い, 実際のプログラム改善例を示した.

## 3　エラーメッセージ変換ツールの作成

本手法を用い, ユーザから GHC のエラーメッセージを入力として受け取り, 変換結果を表示する Web ページの作成を行った. Web ページ上での動作に関しては, 作成したツールの実装には Ruby による簡易 Web アプリケーション作成 DSL である Sinatra を用いた.

## 4　評価

本システムにより GHC のエラーメッセージをより初学者に分かりやすい日本語文へと変換する. これにより初学者のエラーメッセージの内容理解を助けることは可能となる. しかし, リスト 2 のようなエラーに対してはプログラムソースの情報なしでは明確な原因が判断できないため, エラーメッセージを意訳したのみで, 改善方法の提案ができない. また説明文の作成が過去のデータに依存しているため, データの存在しない GHC のエラーメッセージに関しては対応することができない.

リスト 2　エラーメッセージ

```
test.hs:1:36: parse error on input ‘=’
```

## 5　終わりに

GHC によるエラーメッセージを分類, 解析し, 過去の学生の誤答例に基づいたより分かりやすい日本語のメッセージを作成し, 提示することによって初学者の言語理解の支援を行った.

今後の課題として以下に示す. 1 つ目の課題を対応可能なエラーメッセージの追加があげられる. また 2 つ目の課題として, エラーメッセージからは得られないプログラムソースの情報を加味することにより, より詳細な改善案の提示を行える高品質な説明文の作成を行うことである.