7L-08

Idrisプログラム中の任意の部分式の型を表示する支援ツールの設計と実装

南山陸† 川端英之‡ 弘中哲夫‡
広島市立大学情報科学部† 広島市立大学大学院情報科学研究科‡

1 はじめに

依存型を導入した言語では，高階論理を用いた型記述によりプログラムの仕様の詳細を型で表現できるため，簡潔ながら安全なプログラムを記述しやすい [1]．しかしながら，依存型を用いた言語では，計算による値の導出や値どうしの関係を評価することに終始する一般的なプログラミング言語とは異なり，型で表される命題の証明に相当する記述が求められる．そのため，プログラムの記述並びに読解がいずれも容易とは言い難い．本研究では，依存型を持つ関数型プログラミング言語 Idris [2] を対象とし，プログラム中の任意の部分式に対する型を簡潔に表示し，各部分式がどのような証明に相当するかを明確に読み取れるようにするユーザ支援ツールを設計する．

2 現状の環境と支援ツールの必要性

VSCode には Idris-lsp [3] を有効にする拡張機能があり，Idris プログラムを読み書きする際，図 1 に示すように，関数や引数それぞれの項の型を提示してくれる．しかし，関数適用が複数行われる複雑な式では，その

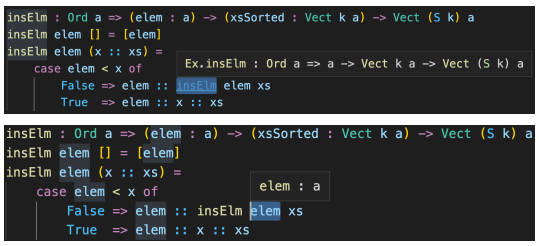

図 1: 現状の VSCode で出来ること

式の一部である部分式の型を捉えたい状況に置かれる．部分式の型を知るには，各項の型をプログラマが頭の中で組み立てて作り上げる必要がある．例えば，以下のように部分式とその各項に対する型情報が与えられたとき，

部分式：`insElm elem`
各項の型情報：

```
insElm : Ord a => a -> Vect k a
                   -> Vect (S k) a
elem : a
```

プログラマは 2 つの型情報を見比べ，`insELm` が `elem` に適用されることから，合成された式すなわち部分式の型情報を以下のように組み立てる．

出力 (部分式の型情報)：

```
insElm elem : Vect k a -> Vect (S k) a
```

支援ツールによってこのような作業を自動化し，頭の中で行っていた作業を可視化出来れば，プログラマの負担を減らして，プログラムを読み取る上で大いに助けになると考える．支援ツールの実現は，初学者が Idris を学ぶ敷居を下げ，中上級者には他の本質的な内容に注力するための助けとなるだろう．

3 支援ツールの設計

3.1 支援ツールの全体像

我々は VSCode プラグインを活用した部分式の型を表示するツールの設計をする．この支援ツールでは既存の VSCode での LSP クライアントである idris-lsp-vscode を拡張し，部分式の型をホバー表示する機能を埋め込む．図 2 にその構造図を示す．

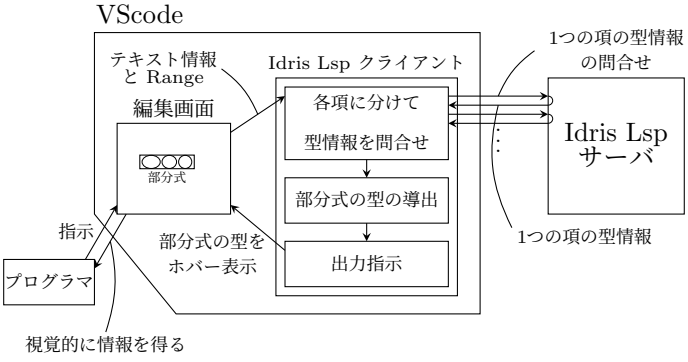

図 2: 機能の相互関係

3.2，3.3 節では，全体を通してこの図 2 を参照している．

A tool for visualizing types of arbitrary subexpressions in a program for a dependentry-typed functional programing language Idris
Minamiyama Riku† Kawabata Hideyuki‡ Hironaka Tetsuo‡
†Department of Computer and Network Engineering, Hiroshima City University
‡Graduate School of Information Sciences, Hiroshima CityUniversity

3.2 部分式の型情報の生成

部分式の各項の型情報を元に部分式の型情報を生成する．それぞれの型情報は Idris Lsp サーバから文字列として得られる．したがって，それぞれを構文解析する必要があり，抽象構文木を作成した後に，単一化して部分式の型情報を生成する．その流れを図 3 に示す．なお，置き換えが可能な型名や変数名は制約条件としてタプルで保持しておき，単一化するときに置き換えを行う．

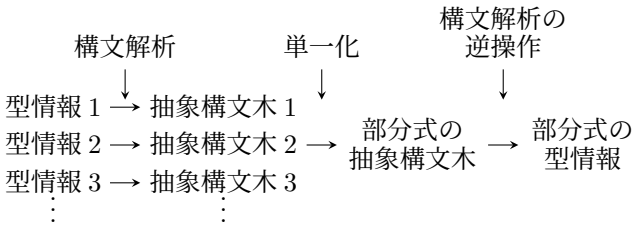

図 3: 部分式の型情報を取得する流れ

また，項の型情報の取得方法は，プログラムファイルにおけるその項の先頭文字の行と列が分かれば，Idris LSP サーバに問合せることで取得することが出来る．行と列の情報は，プログラマが部分式を選択し特定のコマンドを実行すると，その部分式の

- 文字列
- 先頭文字のプログラムファイルにおける行と列
- 終端文字のプログラムファイルにおける行と列

が Idris Lsp クライアントに渡されるので，それらを元に求めることが出来る．

3.3 部分式の型情報をユーザに提示

部分式の型を導出する部分では，各項の型情報と制約条件，生成された部分式の型情報を出力形式に整えて 1 つの文字列とする．そして，それをホバー表示する際に，マークダウン形式にしてホバーメッセージとすることで，ただの文字列ではなく色を付けたり書式を整えたりすることが出来る [4] .

2 節と同様の例で実装した支援ツールの出力結果を以下の図 4 に示す．

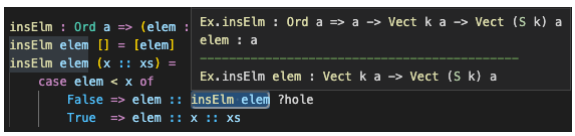

図 4: ホバー表示の様子

また，制約条件を持つ一例とその場合の出力形式を以下に示す．

```
quicksort : Ord a => Vect k a -> Vect k a
list : Vect n a
("k","n"), ("a", "Nat")
---------------------------------------

quicksort list : Vect n Nat
```

4 支援ツールの現状と課題

現状では，図 4 で示したように，選択した範囲の部分式の型を表示する支援ツールを VScode の拡張機能に組み込むことが出来た．しかし，例えば `f (g x y) z` のように引数に関数適用が含まれている，関数の部分適用には対応出来ていない．また，中置関数を含むものも同様に型情報を組み合わせる順序が単純でないので未対応である．つまり，型を導出することが出来る部分式の幅が限られている現状である．

これらの課題を解決する方針として，引数に関数適用が含まれている場合は，図 4 で組み立てた木構造が，複数重なるようなものに対応づけることことが必要となる．中置関数を含む場合は，各項の抽象構文木の構造からそれが中置関数を含む部分式であることを判断し，中置関数を前置関数に変換する処理を行う必要があると考える．

5 まとめ

本研究では，プログラマが選択した部分式の型をホバー表示することで，型の持つ情報量が多い Idris のプログラムを読む負担を軽減する支援ツールの設計と実装を行なった．あらゆる部分式に対応したシステムを完成させて詳細に評価することが今後の課題である．

参考文献

[1] Benjamin C. Pierce. *Types and Programming Languages*. The MIT Press, 2002.

[2] Edwin Brady. *Type-Driven Development with Idris*. Manning, 2017.

[3] Giuseppe Lomurno. idris-community/idris2-lsp: Language Server for Idris2 - GitHub, 2021. https://github.com/idris-community/idris2-lsp.

[4] Microsoft. Language Server Protocol - Microsoft Open Source, 2022. https://microsoft.github.io/language-server-protocol/specifications/.