可逆プログラミング言語 ROOPL のインタープリタの実装

2017SE004 青柳 祐樹　2017SE057 新美 伊織　2017SE086 竹市 翔哉
指導教員: 横山 哲郎

1　背景

可逆計算とは,計算過程においてどの状態からもその直前と直後の状態が一意に決まるという計算モデルである.したがって,可逆計算は,計算過程において完璧に情報を保つことができるため,熱放散の問題の有効な解決策として,考えられている.

現在,完全な可逆計算システムを実現するために,可逆プログラミング言語が研究されている.可逆プログラミング言語は,逆実行が可能なプログラミング言語である.

2　関連研究

可逆プログラミング言語の種類としては,命令型可逆プログラミング言語の *Janus* や,手続き型可逆プログラミング言語の *R*,関数型可逆プログラミング言語の *RFUN* などがあった.

Torben らによる研究では,形式化されていなかったオブジェクト指向型の可逆プログラミング言語 *ROOPL* [1] が示されている.

3　目的

現段階では可逆プログラミング言語 ROOPL の構文や意味論は示されているが,インタープリタは実装されていない.したがって本研究では ROOPL のインタープリタをプログラミング言語 OCaml を用いて実装することを目的とする.

4　計画

実装の手順としては,

1. データ型の定義
2. 字句解析器と構文解析器の実装
3. インタープリタの実装
4. Pretty-printer の実装
5. トップレベルの関数の定義

の順番で作成する.

1では,示された ROOPL の構文のデータ型を定義する.2では,字句解析器,構文解析器を実装するために,ocamllex,ocamlyacc というツールを用いる.ocamllex は言語の字句を定義したファイルを入力として受け取り,字句解析器を出力する.ocamlyacc は,字句解析器と構文規則と評価規則を定義したファイルを入力として受け取り,構文解析器を出力する.構文解析器を用いることで,文字列から構文木を出力することができる.また,これらのツールを用いずに、BNFC[2] という BNF 記法で構文規則を記述すると字句解析器,構文解析器を自動生成するツールを用いる方法もある。

3では,示された意味論に従い,データ型で表された構文木を評価する関数を定義する.

4では,3で定義した関数が返したデータ型を文字列としてきれいに出力するための関数を定義する.

5では,2,3,4で作成したものを組み合わせることで,ROOPL のプログラムを画面に入力すると実行結果を画面に出力する関数を定義する.

参考文献

[1] Torben Mogensen and Robert Gluck:
Design and Implementation of a Reversible Object-Oriented Programming Language (2017)

[2] 「The BNF Converter」
https://bnfc.digitalgrammars.com/