SRL の構文拡張と SRL から RL への翻訳

2018SE034 児玉 春司　2018SE098 若浜 大揮
指導教員：横山 哲郎

1　現状の把握

横山ら [1] が提案した構造化可逆プログラミング言語である SRL と非構造化可逆プログラミング言語である RL が存在する。これらのプログラミング言語は可逆なフローチャートを表現できる。

SRL,RL の構文

SRL,RL の構文規則、構文領域、共通の演算と式を示す。

$$p ::= b \qquad b ::= a \mid \mathrm{if}\ e\ \mathrm{then}\ b\ \mathrm{else}\ b\ \mathrm{fi}\ e \mid b\ b \mid \mathrm{from}\ e\ \mathrm{do}\ b\ \mathrm{loop}\ b\ \mathrm{until}\ e$$

図 1　SRL の構文規則

$$q ::= d^{+} \qquad k ::= \mathrm{from}\ l \mid \mathrm{fi}\ e\ \mathrm{from}\ l\ \mathrm{else}\ l \mid \mathrm{entry} \qquad j ::= \mathrm{goto}\ l \mid \mathrm{if}\ e\ \mathrm{goto}\ l\ \mathrm{else}\ l \mid \mathrm{exit}$$
$$d ::= l : k\ a^{*}\ j$$

図 2　RL の構文規則

$$a ::= x \oplus= e \mid x[e] \oplus= e \mid \mathrm{push}\ x\ x \mid \mathrm{pop}\ x\ x \mid \mathrm{skip} \qquad e ::= c \mid x \mid x[e] \mid e \otimes e \mid \mathrm{top}\ x \mid \mathrm{empty}\ x$$
$$c ::= 0 \mid 1 \mid \cdots \mid 4294967295 \qquad \otimes ::= \oplus \mid * \mid / \mid \cdots \qquad \oplus ::= + \mid - \mid \hat{}$$

図 3　SRL,RL の演算と式

$$\begin{array}{lllllll} SRL: & p \in \mathrm{SRL} & b \in \mathrm{Blk} & & & & \\ RL: & q \in \mathrm{RL} & d \in \mathrm{RLblk} & j \in \mathrm{Jump} & k \in \mathrm{From} & & l \in \mathrm{Label} \\ SRL, RL: & a \in \mathrm{Step} & e \in \mathrm{Exp} & c \in \mathrm{Const} & x \in \mathrm{Var} & \oplus, \otimes \in \mathrm{Op} & \end{array}$$

図 4　SRL,RL の構文領域

2　問題の発見

SRL は高水準言語ではあるが構文が最低限しかない。そこで新たな構文を追加することによって、より使いやすい言語になると考えた。

3　課題の設定

SRL 構文の拡張と拡張した SRL 構文を RL へ翻訳する翻訳規則を考える。今回は新たに for 文を追加することを試みた。

4　解決策の立案

github 上の srl2rl のパッケージ [2] をインストールして、構文の拡張を試みる。SRL.cf ファイルに for 文の構文規則を追加し、抽象構文木が正しく作成できているか TestSRL で確認する。SRL に追加した for 文を RL に翻訳する翻訳規則を考え、Haskell で実装する。
for 文の構文規則を次のように考えた。
Bfor.　Blk ::= ”for” Exp ”ttimes” [Blk] ”do” [Blk] ”loop” [Blk] ”ftimes” [Blk] ”until” Exp;
ttimes は、順実行時にカウンタ変数に何回足すまたは引くかを記述する。ftimes は、逆変換したときに ttimes と同様の演算をする。また for 文を追加するにあたって、インクリメント、デクリメントする Step を追加した。
Sinc.　Step ::= Ident ”++”;
Sdec.　Step ::= Ident ”−”; この for 文のテストを行った結果が以下のとおりである。

```
nanzan@nanzan-VirtualBox:~/srl2rl-master$ ./TestSRL < test.srl

Parse Successful!

[Abstract Syntax]

ESRL [Bfor (Eeq (Evar (Ident "i")) (Econ 0)) [Bstep (Sinc (Ident "i"))] [Bstep S
skip] [Bstep (SasnAdd (Ident "x") (Econ 1))] [Bstep Sskip] (Eeq (Evar (Ident "i"
)) (Econ 5))]

[Linearized tree]

for i = 0 ttimes i ++ do skip loop x += 1 ftimes skip until i = 5
```

図 5　テスト結果

この抽象構文木から RL への翻訳規則を考え、実装することで翻訳できると考えられる。

5　解決策（代替案）の立案

for 文であるためには何が必要か検討した。ループに入る前にカウンタ変数を初期化すべきと考えたため以下のような構文を考えた。
Bfor.　Blk ::= ”tinit” [Blk] ”for” Exp ”do” [Blk] ”loop” [Blk] ”until” Exp ”finit” [Blk];

6　おわりに

今回は for 文の構文規則を考え、抽象構文木を作成するまでしかできなかった。for 文が可逆であり、逆変換・評価ができるのか検証する必要があると考えられる。そのためには、Haskell で書かれた srl2rl のパッケージを理解しなければならない。
また、SRL,RL についても完全には理解できていないため、それぞれの文の操作的意味論などを見ながら理解する必要がある。

役割分担

BNFC とレポート作成　2018SE034 児玉 春司
翻訳規則の検討　2018SE098 若浜 大揮

参考文献

[1] Yokoyama, T., Axelen, H.B. and Glück, R.: Fundamentals of reversible flowchart languages, Theoretical Computer Science, Vol.611, pp.87–115 (2016).

[2] srl2rl パッケージ,https://github.com/yokoyama-lab/srl2rl

[3] 宮本 昌武, 水野 幹大, 野端 祐人, 横山 哲郎: 構造化可逆言語から非構造化可逆言語へのトランスレータの試作,(2020/12/23)