神奈川大学理学部 情報科学科 馬谷研究室

数学的記法による行列表現から S 式表現への変換ライブラリ

本田 統哉 201902988

1 はじめに

本研究では, 人間が普段利用する数学的記法である m*n 行列表現を S 式表現に変換し, 行列計算するプログラムコードに組み込むことを目標とする.Racket 言語はリスト処理に優れた関数型プログラミング言語であるが前もって S 式に変換することにより, プログラムコードを作り上げる上で見やすくなる等のメリットが生まれ, 行列演算やデータの扱いをより柔軟に行うことできる.

2 背景

数学的記法である行列表現は数値計算等においては人間が特に扱いやすいものといえる. しかし, プログラミングにおいてはリストを使ったデータ構造が一般的である. リストはその扱いやすさから様々な構造の表現に適していて,S 式はリストに用いることでプログラム自体もデータとして扱うことができ, データとプログラムコードを同じ構文形式に統一することでシンプルに数学的要素をプログラムに落とし込みやすくなり, 作り手としても理解が容易になることが考えられる. 今回は直感的に行列のリストから S 式としてのリストに変換する手段を作るのが目的である.

3 設計

本プログラムの設計にあたり,3 つの要素に重きを置いて考える. 第 1 に入力として与えられる行列データである文字列から不要な記号を取り除いて後々の処理が容易になるようにする。第 2 に不要な記号が取り除かれた後のデータの各行をリストとして作り、それら結合させて行列全体を表すリストとする。最後に変換された S 式を人間が読みやすい形式となるように各要素を縦に出力することで見やすくする。

4 実装

```
(define (parse-matrix-row row-str)
```

parse-matrix-row という関数を定義する. この関数は行の文字列 row-str を引数に取り, その中の数値をリストに変換する.

```
(let ((numbers-str (filter (lambda (c)
(or (char-numeric? c) (char-whitespace?
    c)))
                (string->list row-str))))
```

文字列 row-str を文字のリストに変換し,filter 関数を用いて数値か空白のみを含む新しいリスト numbers-str を作成する.

```
(map string->number (filter (lambda (s)
    (not (equal? s "")))
    (string-split (list->string
    numbers-str)))))
```

numbers-str リストを文字列に戻し,string-split で空白を区切りとして分割する. 空の文字列を除外し, 残った数値の文字列を string-¿number で数値に変換する. 最終的な数値のリストを返す.

```
(define (matrix-to-s-exp matrix-str)
```

matrix-to-s-exp という関数を定義する. この関数は行列の文字列リスト matrix-str を引数に取り,S 式へと変換する.

```
(let* ((rows (filter (lambda (line)
(equal?
    (string-ref line 0) #\|))
                    matrix-str))
       (parsed-rows (map parse-matrix-row
       rows)))
  (apply list parsed-rows)))
```

filter を使用して,matrix-str から「—」で始まる行のみを取り出す. 取り出した各行に対して parse-matrix-row を適用し, それをリストにまとめて S 式を形成する.

```
(define (print-s-exp-matrix s-exp-matrix)
```

print-s-exp-matrix という関数を定義する. これは S 式の行列 s-exp-matrix を引数に取り, それを出力する.

```
(for-each (lambda (row)
            (println row))
          s-exp-matrix)
```

for-each を使用して s-exp-matrix の各行に対して出力処理を行う. その後,println 関数で各行を出力する.

```
(define input-matrix
  '("+-----+"
    "|1 3 4|"
    "|4 0 4|"
    "|4 5 6|"
    "+-----+"))
```

input-matrix という変数を定義し, サンプルの行列の文字列リストを代入する.

```
(define s-exp-matrix (matrix-to-s-exp
    input-matrix))
```

matrix-to-s-exp 関数を使用して input-matrix を S 式に変換し, その結果を s-exp-matrix 変数に格納する.

```
(print-s-exp-matrix s-exp-matrix)
```

最後に print-s-exp-matrix 関数を呼び出して,S 式に変換された行列を出力する.

結果として, 以下のような結果が出力される。

```
'(1 3 4)
'(4 0 4)
'(4 5 6)
```

実装では,Racket 言語の string-¿list、filter、map といった関数を利用して, 行列の文字列から数値を取り出し, それらを S 式のリストに変換する一連の関数を定義した. 具体的には, プログラムの実行が開始される matrix-to-s-exp 関数が, 与えられた行列の文字列リストから「—」で始まる行を取り出し、それぞれの行に対して parse-matrix-row 関数を適用する. この関数は, 文字列を個々の文字へと分解し, 数値のみを取り出してリストに変換する.

5 まとめ

本研究では,Racket 言語を用いた数学的記法である m*n 行列表現から S 式表現への変換ライブラリの設計と実装を行った. これは関数型プログラミング言語におけるデータ処理の拡張性等の視野が広がるものだと思っている. このプログラムはプログラム作成の初期段階での利用において, その真価を発揮するものなので今後の課題としては、変換処理の効率化、複雑な行列演算への対応ができるように工夫したい. 本研究では行列計算するコードに組み込んで実装するところまでが目標であったが, できるところまでしかできず実装までは至らなかった.

参考文献

[1] JAME W.STELLY, *RACKET PROGRAMMING THE FUN WAY FROM STRINGS TO TURING MACHINES,* No Starch Press, 2021.

[2] 湯浅太一, *Scheme* **入門**, 岩波書店, 1991.

[3] 浅井健一, **プログラミングの基礎**, サイエンス社, 2007.

[4] Beautiful Racket `https://beautifulracket.com/`.