Systems Programming Project1 Documentation

Hakan Emre Aktas

Description

For this project I implemented a translator which takes MatLang code as input and translates it to C code. Matlang language includes various properties such as,

- Scalar, vectors and matrices as variables
- Assignment statements which can assign values to scalars, vectors, matrices and selected elements of matrices/vectors
- Functions such as tr(), choose() and sqrt()
- 3 operations: summations, subtraction and multiplication
- Single and double nested loops
- Print function to print results to console
- Printsep function to print a separator line

Translator code translates the MatLang code if it abides the rules given in the problem description. If it does not it finds the erroneous line and prints Error (line <x>) to the console.

Implementation

I used the BNF structure provided to parse the input. I used functions like **isExpression**, **isTerm** and **isFactor** recursively to implement it. This part of the code can perform syntax checking and type checking. I also used another recursive structure containing functions **MatrixSizeCheck, MatrixSizeCheck2 etc.** to check the dimensions of matrices/vectors that are being summed, multiplied or subtracted. I used functions to keep the variables' type information and matrices/vectors' dimension information.

After the syntax checks, code starts to form the output. **Initialize** function prints the necessary pre-written functions like **MatrixMultp**, **matrixSum** etc. and prints the basic C constructs. Code then edits the parts that are not compatible with C syntax. Every operation and function containing a matrix/vector are replaced with the functions I have written to make them compatible with C. To make assignments containing matrices/vectors the code generates intermediate variables in the format **ihopeitsfine**<int> to assign temporarily and then prints a for loop to assign each value one by one using **ihopenottaken** as the loop index value.

Editing the input is a big part of the code and it is mostly handled in **isFactor** and **MatrixSizeCheck4** functions. The rest of the editing is mostly just using templates corresponding to the MatLang format.

Notes: The output file uses math.h header file so it should be compiled with -lm. I am aware that the code consists of many unnecessary parts that can be excluded with some changes like opening the same file 3 times, but it was working and I didn't want to break it. Code contains a bug which is related to right associativity of minus operator. In the BNF structure it looks like minus operator has right associativity, but it doesn't. I tried to fix it by giving minus precedence over plus but it is still problematic. I noticed and changed it at the last day so the comments on some of the MatrixSizeCheck functions may be wrong.

Expression Grammar to be Used in Project1

expr → *term moreterms*

moreterms → + *term moreterms*
| - *term moreterms*
| **null**

term → *factor morefactors*

morefactors → * *factor morefactors*
| **null**

factor → **(** *expr* **)**
| id[expr, expr]
| id[expr]
| sqrt(expr)
| choose(expr1,exr2,expr3,expr4)
| **id**
| **num**
| tr(expr)