RWTH AACHEN UNIVERSITY
Chair of Computer Science 2
Software Modeling and Verification

Master Thesis Proposal

Title tbd

Sascha Thiemann
Matr.-No.: 406187
Study Program: Master Computer Science
September 27, 2022

Supervisors: apl. Prof. Dr. Thomas Noll
Chair for Software Modeling and Verification
RWTH Aachen University

Contents

1 Introduction

- Short intro into QC
  - What is QC
  - Why is it important
  - What can it be used for

2 Motivation

- Classical control flow vs quantum control flow [YYF12]
- *Intro quantum control flow*

With the emergence of quantum computing, many quantum languages were introduced. While some may focus on a lower level representation of quantum circuits and others on high level interactions, most language restrict themselves to only quantum data while using classical control flow. Although quantum control flow was defined by Ying et al. [YYF12] over 10 years ago, only recently was a language with quantum control flow at its core proposed by Yuan et al. [YVC24].

The so-called "Quantum Control Machine"

- bounded by reversibility and synchronicity
  - what is reversibility, why is it needed
  - what is synchronicity, why is it needed
- based on classical assembly languages (jumps, registers, ...)
- Issues with qcm
- very unreadable
- can be reduced to basics

3 Concept

- Language features: qif-else, bounded loops, (boolean eval)
- Translation to quasm
- overall (more) realistic for NISQ
- Further (compliler optimizations)
- Example grammar

Bibliography

[YVC24] Charles Yuan, Agnes Villanyi, and Michael Carbin. Quantum control machine: The limits of control flow in quantum programming. *Proceedings of the ACM on Programming Languages*, 8(OOPSLA1):1–28, 2024.

[YYF12] Mingsheng Ying, Nengkun Yu, and Yuan Feng. Defining quantum control flow.

```
grammar Luie;

parse
 : block EOF
 ;

block
 : (definition | statement)*
 ;

definition
 : 'qubit' IDENTIFIER ';'
 ;

statement
 : GATE IDENTIFIER ';'
 | qifStatement
 ;

qifStatement
 : ifStat elseStat? END
 ;

ifStat
 : IF IDENTIFIER DO block
 ;

elseStat
 : ELSE DO block
 ;

GATE
 : XGATE
 | ZGATE
 | HGATE
 ;

XGATE : 'x';
ZGATE : 'z';
HGATE : 'h';

IF       : 'qif';
ELSE     : 'else';
DO       : 'do';
END       : 'end';

IDENTIFIER
 : [a-zA-Z_] [a-zA-Z_0-9]*
 ;
```

```
50
51      COMMENT
52       : ( '//' ~[\r\n]* | '/*' .*? '*/'
             ) -> skip
53       ;
54
55      SPACE
56       : [ \t\r\n\u000C] -> skip
57       ;
```