软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2011,22(5):877−886 [doi: 10.3724/SP.J.1001.2011.03979]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

量子程序设计语言 NDQJava-2*

刘 玲[1,2], 徐家福[1,2+]

[1](计算机软件新技术国家重点实验室(南京大学),江苏 南京 210093)

[2](南京大学 计算机科学与技术系,江苏 南京 210093)

Quantum Programming Language NDQJava-2

LIU Ling[1,2], XU Jia-Fu[1,2+]

[1](State Key Laboratory for Novel Software Technology (Nanjing University), Nanjing 210093, China)

[2](Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China)

+ Corresponding author: E-mail: xjf@nju.edu.cn, http://cs.nju.edu.cn

Liu L, Xu JF. Quantum programming language NDQJava-2. *Journal of Software*, 2011,22(5):877−886.
http://www.jos.org.cn/1000-9825/3979.htm

Abstract: This paper presents an overview of a quantum programming language NDQJava-2. NDQJava-2 is an extension of NDQJava with additional quantum components such as quantum conditional statement, quantum loop statement, quantum subprogram, quantum module, and quantum exception handling mechanism. The added commponents make NDQJava-2 a structured quantum programming language. Experience in writing quantum programs indicates that compared with NDQJava, NDQJava-2 is a more practical, more readable and more suitable (components setting) quantum programming language.

Key words: quantum programming language; quantum conditional statement; quantum loop statement; quantum subprogram; quantum module; quantum exception handling mechanism

摘 要: 简述了量子程序设计语言 NDQJava-2.该语言是在 NDQJava 的基础上增添了量子条件语句、量子循环语句、量子子程序、量子模块以及量子异常处理机制等量子成分,使其成为一种结构化的量子程序设计语言.书写量子程序的实践表明,相对于 NDQJava 而言,NDQJava-2 是一种更为实用、易读,其成分设定更为合适的量子程序设计语言.

关键词: 量子程序设计语言;量子条件语句;量子循环语句;量子子程序;量子模块;量子异常处理机制

中图法分类号: TP311 **文献标识码**: A

自 1982 年 Feynman[1]提出基于量子力学原理构建量子计算机的设想以来,迄今已近 30 年.研究表明,其前景看好.特别是 1994 年 Shor[2]关于大数质因子分解算法与 1996 年 Grover[3]关于数据库搜索算法等划时代意义的工作出现后,国际上对量子计算机的研究更加重视,虽然目前在其硬件实现上困难颇大,可用之量子算法又为数甚少,但由于量子计算机具有“存储容量大、运算速度快、安全性能好”等特点,加诸相关学科(如半导体电子

* 基金项目: 国家自然科学基金(60736044, 60903107); 国家重点基础研究发展计划(973)(2004CB318108); 国家高技术研究发展计划(863)(2006AA01Z141); 高等学校博士学科点专项科研基金(20090002120005)

收稿时间: 2009-07-20; 定稿时间: 2011-01-12

学、数学等)之发展,不少人认为,实用的量子计算机可望于 2030 年左右出现.作为其重要软件之一的量子程序设计语言及其处理系统之研究实为必要.

NDQJava 语言[4]的设计与实现乃国内在这一领域内的首次尝试,虽然该语言已在经典计算机上模拟实现,但语言中的量子成分不多,只有量子类型、量子变量、量子表达式、量子语句、量子分程序,因而在书写量子程序时感到欠便,有必要进行改进,增添若干量子成分以应书写量子程序之需.

1 准 则

NDQJava 的设计准则是:实用性、简明性、严谨性和快速性[5].NDQJava-2 的设计准则亦然.这 4 条准则并非孤立,既有分工,又有联系.实用性居首,其理自明,盖因设计语言的目的在于用,不实用的语言当然毫无意义.简明的对立面是繁琐,繁琐的语言当然用户不愿使用,因而其实用性也不高.但是,如果语言成分过多,用户虽然可用,但会感到不便,这样实用性也会下降.准此,严谨、快速与实用、简明之间亦相互联系.此次改进适当增添量子成分,使用户在书写量子程序时要感到比使用 NDQJava 更为好用,但又不能失之繁琐;语言之语法、语义定义务求严谨,不能引起歧义.为使用户使用时较能感到熟悉,得心应手,NDQJava-2 的语法形式和经典 Java 类似.快速性之含义有二:一为改进工作之快速,当然指的是在保证质量前提下的快速;二为用以书写出之量子程序运行时的快速.不过,这里以第 1 种含义为主.

2 语言简介

如前所述,NDQJava-2 是在 NDQJava 的基础上适当增添若干量子成分的结果,为完备计,本文在附录中给出了 NDQJava 语言量子成分的语法与语义.

NDQJava-2 所增添之量子成分有量子条件语句、量子循环语句、量子子程序、量子模块以及量子异常处理机制,今分述如下.

2.1 量子条件语句

NDQJava 中并未引入量子条件语句,在程序书写中,遇到条件控制则需借助经典部分,从而引起经典部分与量子部分之切换,以致影响功效.为此,引入量子条件语句.考虑用户使用 Java 的习惯性,该语言中的量子条件语句和经典 Java 中形式类似.

如所熟知,在 QCL 等语言中通过两个量子变量公用一个使能量子来实现量子控制,我们在 NDQJava-2 中则是每个量子变量都有一个相应的使能量子.这样,可能比两个量子变量公用一个使能量子更为清晰明了与合理.为此,今给出如下之使能量子声明:

语法:

〈使能量子声明〉::=〈量子类型〉〈使能量子名〉:〈量子变量名〉

〈使能量子名〉::=〈标识符〉

语义:

使能量子声明用以定义使能量子及其相应的量子变量,使能量子一直处于基态,可以对其进行测量运算、比较等.其初值为 0,测量后也要变成初态.

量子条件语句定义如下:

语法:

〈量子条件语句〉::=if (〈量子条件表达式〉)〈量子语句块〉

|if (〈量子条件表达式〉)〈量子语句块〉 else 〈量子语句块〉

〈量子条件表达式〉::=〈使能量子名〉

|〈使能量子名〉〈量子关系运算符〉〈使能量子名〉

〈量子关系运算符〉::=〈|〉|〈=|〉=|==|!=

语义:

使能量子为|1⟩时或者两个使能量子比较的结果是|1⟩时,量子条件表达式为真,执行 if 后的量子语句块;否则为假,便掠过量子条件表达式后之量子语句块或执行 else 后面的量子语句块.量子关系运算符是使能量子的简单比较操作符,为便于用户使用,其意义和经典意义相同.

2.2 量子循环语句

NDQJava 中亦未引入量子循环语句,循环控制也是由经典部分来实现的,从而影响了功效.因此,有必要引入量子循环语句.NDQJava-2 中引入了 3 种量子循环语句,其定义如下:

语法:

⟨量子循环语句⟩::=while ⟨量子条件表达式⟩⟨量子语句块⟩

|until ⟨量子语句块⟩ while ⟨量子条件表达式⟩

|for (⟨量子初化⟩;⟨量子条件表达式⟩;⟨量子更新表达式⟩)

⟨量子更新表达式⟩::=increase|decrease

语义:

while 语句首先判断量子条件表达式是否为真,为真时执行量子语句块,重复上述过程;一旦量子表达式为假,便退出循环,执行 while 语句后面的语句.until-while 语句是先执行一次量子语句块,然后判断循环量子条件,条件为真时重复执行量子语句块;直到量子条件为假时,退出循环.for 语句和经典 for 语句的语义相同,这里不再赘述.量子更新表达式为递增和递减操作.

2.3 量子子程序

量子子程序是与量子子计算任务相应的处理对象和处理规则的描述,它是一个可被量子程序(单位)调用的量子程序单位.它包括定义和调用两个方面.前者称为量子子程序定义,后者称为量子子程序调用.在语言中设有子程序,不但可以减少重复编程,而且可以使程序结构更为清晰.在量子计算中,当出现代码复用时,便可以将多处重复出现的代码编写成一个子程序.比如在 Grover 算法中,会用到两次 Oracle O 操作,如果不引入量子子程序,这些代码就要在程序中书写两次,这样的程序易读性低,重复、冗长,简明性差.因此,有必要在量子程序设计语言中引入量子子程序.

2.3.1 量子子程序定义

语法:

⟨量子子程序定义⟩::=⟨量子函数定义⟩|⟨量子过程定义⟩

⟨量子函数定义⟩::=⟨量子函数首部⟩⟨量子函数体⟩

⟨量子函数首部⟩::=qufunction qtype ⟨量子函数名⟩(⟨形参部分⟩)

⟨量子函数体⟩::=⟨量子分程序⟩⟨回送语句⟩

⟨量子过程定义⟩::=⟨量子过程首部⟩⟨量子过程体⟩

⟨量子过程首部⟩::=quprocedure ⟨量子过程名⟩(⟨形参部分⟩)

⟨量子过程体⟩::=⟨量子分程序⟩

⟨量子分程序⟩::=begin ⟨量子变量声明块⟩⟨量子语句块⟩⟨量子子程序调用表⟩ end

⟨回送语句⟩::=return;| return ⟨量子表达式⟩;

⟨量子函数名⟩::=⟨标识符⟩

⟨量子过程名⟩::=⟨标识符⟩

⟨形参部分⟩::=⟨形参元⟩|⟨形参元⟩,⟨形参部分⟩

⟨形参元⟩::=qtype ⟨形参⟩

⟨形参⟩::=⟨标识符⟩

⟨量子子程序定义表⟩::=⟨量子子程序定义⟩|⟨量子子程序定义表⟩;⟨量子子程序定义⟩

语义:

量子子程序定义起定义作用.有两类量子子程序,即量子函数和量子过程,它们分别由量子函数首部与量子函数体以及量子过程首部与量子过程体组成.量子函数有回送值,量子过程无回送值.

2.3.2 量子子程序调用

量子子程序调用引起相应子程序体的执行[6].参数传递方式同 Java,为按值传递.

〈量子子程序调用〉::=〈量子函数调用〉|〈量子过程调用〉

〈量子函数调用〉::=〈量子函数名〉(〈实参表〉)

〈量子过程调用〉::=〈量子过程名〉(〈实参表〉);

〈实参表〉::=〈实参〉|〈实参〉,〈实参表〉

〈实参〉::=〈标识符〉

〈量子子程序调用表〉::=〈量子子程序调用〉|〈量子子程序调用表〉;〈量子子程序调用〉

例:量子翻转过程:

```
quprocedure qtype flip(qtype q)          //量子翻转
{
    qtype p:q;
    int n=q.size();                      //量子变量 q 的量子比特数
    for (p:=init q.size():0; q<=q.size()−1; q.increase)
        q.u_swap(p,n−p−1);
}
```

2.4 量子模块

经典程序中往往包含一些彼此相对独立但又有一定逻辑联系的成分,将这些成分集中起来,便构成一个模块,它和模块外亦可存在一定的联系.在语言中设置模块,既可以提高程序的结构性,隐蔽相关信息,又可使之易读、易写、易维护.在量子程序中情况类似,故而在 NDQJava-2 中设置了量子模块.

语法:

〈量子模块〉::=〈量子模块式〉〈量子模块体〉

〈量子模块式〉::=qumodule specification [main]〈量子模块名〉
[imports 〈移入表〉][exports 〈移出表〉]〈声明部分〉
end specification

〈移入表〉::=〈标识符〉|〈移入表〉,〈标识符〉

〈移出表〉::= 〈标识符〉|〈移出表〉,〈标识符〉

〈声明部分〉::=[〈量子变量声明表〉][〈量子函数首部表〉][〈量子过程首部表〉]

〈量子函数首部表〉::=〈量子函数首部〉|〈量子函数首部表〉;〈量子函数首部〉

〈量子过程首部表〉::=〈量子过程首部〉|〈量子过程首部表〉;〈量子过程首部〉

〈量子模块体〉::=qumodule body [main]〈量子模块名〉〈体声明部分〉〈量子子程序定义表〉
〈量子子程序调用表〉 end body

〈体声明部分〉::=[〈量子变量声明表〉]

〈量子程序〉::=〈量子模块列〉

〈量子模块列〉::=〈量子模块〉|〈量子模块列〉;〈量子模块〉

语义:

量子模块用以封装一组逻辑上互有关联的成分,如量子变量、量子函数、量子过程等.它是一种信息隐蔽的设施.每个量子模块由两部分组成[7]:一部分是量子模块式,另一部分是量子模块体.前者用来代表和外界的接口,可以有移入表、移出表和声明以及量子函数首部、量子过程首部;后者表示模块的实现细节.移入表中之标

识符代表在其他模块中定义、而在本模块中可用的成份名,移出表中之标识符代表在本模块中定义而其他模块中可用之成份名.量子程序由一个或多个相继的量子模块组成,其中有一个为主模块(模块名前冠以 main),主模块中第 1 个子程序调用为程序入口.

2.5 量子异常处理

量子变量性质独特,不能复制,测量会发生塌缩等等,这些都会引发相应异常.比如,量子赋值语句在满足一定条件下才可使用,如果在相应条件不满足时执行赋值语句便会引发异常.

本语言中的异常包括语言定义的异常和用户定义的异常:前者是由语言定义的(如初态异常、使能量子异常、测量异常等);后者是通过如下之异常定义来定义的.

2.5.1 异常定义

语法:

〈异常定义〉::=edef〈异常名〉=〈串〉

〈异常名〉::=〈标识符〉

语义:

异常定义用以定义异常,其中,异常名表示所定义之异常的名字,串表示用以定义相应的异常之一串字符.

2.5.2 异常引发

异常引发借助引发异常语句实现.

语法:

〈引发异常语句〉::=raise〈异常名〉;

语义:

执行引发异常语句引起发生相应异常,挂起发生异常的程序的执行,转至相应异常处理程序进行处理.

2.5.3 异常处理

异常处理借助异常处理程序实现,今从略.

3 示 例

Grover 算法的复杂度是 $O\left(\sqrt{n}\right)$, 但是经典算法的复杂度至少是 $O(\log n)$.

算法描述如下[3]:

1. 初化:

对 n 位量子比特初化为|0000…..00〉,对其应用 Hadamard 变换 $H^{\otimes n}$.

2. 进行 Grover 算法迭代.该迭代由以下 4 步组成:

(a) 应用 Oracle O;进行选择性变换,令 s 为输入的一个基态,当 $O(s)=1$ 时,把基态 s 旋转 180°,当 $O(s)=0$ 时,不变.

(b) 应用 Hadamard 变换 $H^{\otimes n}$.

(c) 执行条件相移 R,使|0〉以外的每个基态的相位都取反.

(d) 应用 Hadamard 变换 $H^{\otimes n}$.

3. 进行测试,如果结果与查询的结果相符,则结束,并输出结果;如果不符,则重新开始.

Grover 算法的代码如下:

```
qumodule specification main Grover:
        imports     bit, size(),rq,
                    PI, ceil, floor, sqrt, pow, log;
        exports     query, diffuse, grover;
end specification
```

```
qumodule body main Grover:
        grover(rq);
        quprocedure grover(qtype eq)
        {
            int n=eq.mearsure();                          //得到要查询的值
            //qubites 的个数,是以 2 为底的 n 的对数
            int num=(int)floor(log(n)/log(2))+1;
            //迭代的次数
            int m=(int)ceil(PI/8*sqrt(pow(2, num)));
            System.out.println(num+"qubits using,"+m+":iterations");
            qytpe tempq;
            tempq:=init num:n;                            //初态为|n⟩
            int x=0, i;
            qtype q;
            q:=init num::0;
            qtype p:q;                                    //量子 q 的使能量子
            p:=init 1::0;
            while (x!=tempq)
            {
                q.reset();                                //重新初化 q
                q:=q.u_h();                               //H 酉操作
                for (i=1; i<=m; i++){
                q.query(q,p,tempq);                       //Oracle O 操作
                q.u_Cphase(Math.PI,p);                    //相位变化
                q.query(q,p,tempq);                       //Oracle O 操作
                q:=diffuse(q);                            //变换 D 操作
            }
            x=q.measure();                                //全测量
        System.out.println("measure:"+x);
        }
        quprocedure query(qtype q,qtype p,qtype n)        //Oracle O 操作
        {
            qtype ee:q;
            int i;
            for (ee:=init q.size():0; ee<q.size(); ee.increase)
            {
                if (!bit(n,ee))
                {
                    i=ee.measure();
                    q.u_x(i);
                }
            }
```

```
            q.u_cnot(p);
            //让 q 复原
            for (ee:=init q.size():0; ee<q.size(); ee.increase)
            {
                if (!bit(n,ee))
                {
                    i=ee.measure();
                    q.u_x(i);
                }
            }
        }
        quprocedure diffuse(qtype q,qytpe p)
        {
            q.u_h();                    //H 操作
            q.u_x();                    //X 操作
            q.u_cphase(pi,p);           //Cphase 操作
            q.u_x();                    //X 操作
            q.u_h();                    //H 操作
        }
end body
```

4 结 语

历史上看,如下 4 种量子程序设计语言颇具代表性:

- 量子伪码[8].一种用于书写量子算法之量子伪码,其中给出了量子寄存器与经典寄存器之区别约定、量子寄存器纠缠范围约定以及量子寄存器之初化、使用、测量等约定.严格说来,量子伪码尚难称作一种量子程序设计语言.
- Qgol[9].一种描述量子算法之图示语言(图之结点表示数据,连线表示操作).作者阐明了对函数式语言的看法,指出函数式语言与量子系统间的某些对应,尚难看出 Qgol 本身是一种函数式语言.已开发出一个图形编辑程序.
- QCL[10−12].一种结构化命令式量子程序设计语言.程序是语句与定义列.定义分常量定义与变量定义.语句有简单语句、流程控制语句、交互命令这 3 类.类型有标量、张量、量子、布尔、串等.其中,量子类型又分通用、常量、目标、清除这 4 种类型.作者考虑深入细致,似嫌成分过多,失之繁琐,但颇具参考价值.
- QML[13].一种有限类型之函数式量子程序设计语言.其主要特点是:和“经典控制、量子计算”不同,提出了“量子数据、量子控制”,利用一阶严格线性逻辑使弱化得以显式表达,程序是一列项定义.以范畴论为基础,给出了 QML 的指称语义与操作语义.内容亦嫌庞杂,有失简明.

以上 4 种语言各有短长,NDQJava 与 NDQJava-2 是在吸取前述语言设计经验之基础上设计的,当然仍属实验性工作,欠妥之处,尚祈读者指正.

References:

[1] Feynman R. Simulating physics with computers. Int'l Journal of Theoretical Physics, 1982,21(6):467−488. [doi: 10.1007/BF02650179]

[2] Shor PW. Algorithms for quantum computation: Discrete log and factoring. In: Santa F, ed. Proc. of the 35th Annual Symp. on the Foundations of Computer Science. New Mexico: IEEE Computer Society Press, 1994. 20−22.

[3] Grover L K. A fast quantum mechanical algorithm for database search. In: Miller GL, ed. Proc. of the 28th Annual ACM Symp. on the Theory of Computing. Philadelphia: ACM Press, 1996. 212−219.

[4] Xu JF, Song FM, Qian SJ, Dai JA, Zhang YJ. Quantum programming language NDQJava. Journal of Software, 2008,19(1):1−8 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/19/1.htm [doi: 10.3724/SP.J.1001. 2008.00001]

[5] Song FM, Qian SJ, Dai JA, Zhang YJ, Xu JF. Processing system of quantum programming language NDQJava. Journal of Software, 2008,19(1):9−16 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/19/9.htm [doi: 10.3724/SP.J.1001. 2008.00009]

[6] Xu JF. Introduction to Programming Language Ada (1). In: Selected Works. Nanjing: Nanjing University Press, 1990. 169−179 (in Chinese).

[7] Xu JF. Module: A structured programming tool. In: Xu Jiafu Selected Works. Nanjing: Nanjing University Press, 1992. 171−173 (in Chinese).

[8] Knill EH. Conventions for quantum pseudocode. LANL Report, LAUR-96-2724, Los Alamos: Los Alamos National Laboratory, 1996.

[9] Baker GD. “Qgol”: A system for simulating quantum computations: Theory, implementation and insights [Honours Degree Thesis] Macquarie University, 1996.

[10] Ömor B. A procedural formalism for quantum computing [MS. Thesis]. Vienna: Technical University of Vienna, 1998.

[11] Ömor B. Structured quantum programming [Ph.D. Thesis]. Vienna: Technical University of Vienna, 2003.

[12] Ömor B. Quantum programming in QCL [MS. Thesis]. Vienna: Technical University of Vienna, 2000.

[13] Gratlage JJ. QML: A functional quantum programming language [Ph.D. Thesis]. Nottingham: University of Nottingham, 2006.

附中文参考文献:

[4] 徐家福,宋方敏,钱士钧,戴静安,张云洁.量子程序设计语言 NDQJava.软件学报,2008,19(1):1−8. http://www.jos.org.cn/1000-9825/19/1.htm [doi: 10.3724/SP.J.1001.2008.00001]

[5] 宋方敏,钱士钧,戴静安,张云洁,徐家福.量子程序设计语言 NDQJava 处理系统.软件学报,2008,19(1):9−16. http://www.jos.org.cn/1000-9825/19/9.htm [doi: 10.3724/SP.J.1001. 2008.00009]

[6] 徐家福.程序设计语言 Ada 简介(一).见:论文选集.南京:南京大学出版社,1990.169−179.

[7] 徐家福.模块:一种结构化的程序设计工具.见:徐家福文集.南京:南京大学出版社,1992.171−173.

附录. NDQJava 量子成分之定义

1. 量子类型

语法:

〈量子类型〉::=*qtype*

语义:

量子类型是描述量子数据以及在量子数据上的一组操作,量子类型 *qtype* 的值集是量子态,操作集是酉操作及测量.

2. 量子变量

语法:

〈量子变量声明〉::=〈量子类型〉〈量子变量名表〉

〈量子变量名表〉::=〈量子变量名〉|〈量子变量名〉,〈量子变量名表〉

〈量子变量名〉::=〈标识符〉

〈量子变量声明块〉::=〈量子变量声明〉|〈量子变量声明块〉,〈量子变量声明〉

语义:

量子变量是量子类型的变量,即以 *qtype* 类型为类型的变量,其值为量子态.

3. 量子表达式

语法:

⟨量子表达式⟩::=⟨量子预初化表达式⟩|⟨酉表达式⟩
|⟨张量积表达式⟩|⟨测量表达式⟩

⟨量子预初化表达式⟩::=⟨表达式⟩::⟨表达式⟩

⟨酉表达式⟩::=⟨量子变量名⟩⟨酉操作符⟩((⟨实参表⟩))
|⟨酉表达式⟩⟨酉操作符⟩((⟨实参表⟩))

⟨酉操作符⟩::=*u_h*|*u_x*|*u_i*|*u_z*|*u_s*|*u_rotx*
|*u_roty*|*u_rotz*|*u_cnot*|*u_crotx*|*u_croty*
|*u_crotz*|*u_swap*|*u_chpa*

⟨实参表⟩::=⟨经典变量表⟩

⟨张量积表达式⟩::=⟨张量积操作数⟩⟨张量积操作符⟩⟨张量积操作数⟩

⟨张量积操作数⟩::=⟨量子变量名⟩|⟨张量积表达式⟩

⟨张量积操作符⟩::=*q_tensor*

⟨测量表达式⟩::=⟨量子变量名⟩⟨量子测量⟩

⟨量子测量⟩::=*q_measure*(⟨实参表⟩)

语义:

量子表达式分为量子预初化表达式、酉表达式、张量积表达式和测量表达式 4 类:

- 预初化表达式由两个经典非负整值表达式组成,第 1 个表达式给出量子变量所具有的量子比特数即量子比特的位数,第 2 个表达式给出量子变量的初态的基态序号.
- 酉表达式是对量子变量或酉表达式进行酉运算.这里的酉操作符有 14 类酉操作,分别为 *u_i*,*u_h*,*u_x*,*u_s*,*u_z*,*u_rotx*,*u_roty*,*u_rotz*,*u_cnot*,*u_crotx*,*u_croty*,*u_crotz*,*u_swap*,*u_chpa*.其中,*u_h* 是对量子变量施行 Hadamard 变换,*u_x* 是对量子变量施行 *X* 变换,*u_cnot* 是对其施行控制非运算,等等.
- 张量积表达式是量子变量与量子变量、量子变量与张量积表达式以及张量积表达式与张量积表达式之间的张量积操作,其中,*q_tensor* 是张量积操作符.
- 测量表达式用以将量子变量值转换为经典值,*q_measure* 是测量操作.当实参表为空时是全测量,当实参表不为空时是部分测量.

4. 量子语句

语法:

⟨量子语句⟩::=⟨量子空语句⟩|⟨量子预初化语句⟩|⟨量子赋值语句⟩
|⟨量子测量语句⟩|⟨量子分程序⟩

语义:

量子语句有量子空语句、量子预初化语句、量子赋值语句、量子测量语句和量子分程序 5 种.

(1) 量子空语句

语法:

⟨量子空语句⟩::=;

语义:

空语句对程序执行不起任何作用

(2) 量子预初化语句

语法:

〈量子预初化语句〉::=〈量子变量名〉:=〈量子预初化表达式〉;

语义:

量子预初化语句用来对量子变量置量子比特数和量子初态,:=是量子复制操作符

(3) 量子赋值语句

语法:

〈量子赋值语句〉::=〈量子变量名〉:=〈量子变量名〉;

|〈量子变量名〉:=〈酉表达式〉;

|〈量子变量名〉:=〈张量积表达式〉;

语义:

量子赋值语句是用一个量子常量或一个量子表达式的值作为一个量子变量的当前值,在:=两边的类型要一致.量子变量所赋之值有 3 种情况:一个是量子变量之值,一个是酉表达式之值,另一个是张量积表达式之值.其中,所赋变量的值或者表达式的值是基态,或者为非基态但是经历足迹可知.

(4) 量子测量语句

语法:

〈量子测量语句〉::=〈经典变量名〉:=〈测量表达式〉;

语义:

量子测量语句用来执行量子测量表达式,将其值赋予左边的经典变量.它是量子空间和经典空间之间转换的渠道.

(5) 量子分程序

语法:

〈量子语句块〉::=〈量子语句〉|〈量子语句块〉〈量子语句〉

〈量子分程序〉::=begin 〈量子变量声明块〉〈量子语句块〉 end

〈量子变量声明块〉::=〈量子变量声明〉|〈量子变量声明块〉〈量子变量声明〉

语义:

量子分程序是括以 begin 及 end 的一组量子变量说明或(与)量子语句.它可以提高程序的易读性,也便于处理器系统对经典部分和量子部分进行分别处理.

5. 量子程序

语法:

〈程序〉::=〈经典程序〉|〈量子程序〉

〈经典程序〉::=〈Java 程序〉

〈量子程序〉::=〈量子分程序〉

语义:

NDQJava 程序有经典程序和量子程序两种,经典程序是 Java 程序,量子程序是量子分程序.

刘玲(1981－),女,山东菏泽人,硕士生,主要研究领域为新型程序设计信息检索,机器学习.

徐家福(1924－),男,教授,博士生导师,主要研究领域为高级语言,新型程序设计,软件自动化,自然语言处理.