(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 111291892 B
(45)授权公告日 2023.01.17

(21)申请号 202010055786.2

(22)申请日 2020.01.17

(65)同一申请的已公布的文献号
申请公布号 CN 111291892 A

(43)申请公布日 2020.06.16

(73)专利权人 深圳大学
地址 518054 广东省深圳市南山区粤海街道南海大道3688号

(72)发明人 王平 刘光强

(74)专利代理机构 广州粤高专利商标代理有限公司 44102
专利代理师 张金福

(51)Int.Cl.
G06N 10/60(2022.01)

(56)对比文件
CN 106197455 A,2016.12.07
US 7383235 B1,2008.06.03
Lin Bie etc..Quantum genetics clustering algorithm based on high-Dimensional and multi-chain coding scheme.《2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)》.2018,

审查员 李若童

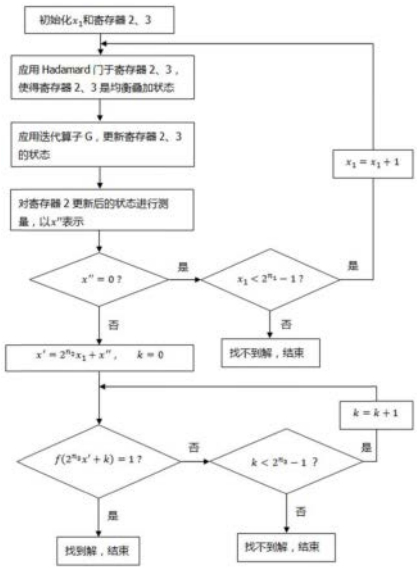
权利要求书2页 说明书12页 附图3页

(54)发明名称

一种量子并行搜索方法

(57)摘要

本发明涉及一种量子并行搜索方法，若将Oracle算子作为一个单元，则本发明所述方法能够在以运行时间复杂度为$O(2^{n/4})$和线路复杂度为$O(2^{n/4})$的情况下以接近1的概率找到搜索问题的解。由Grover算法的性质可知，量子迭代线路由G算子一个接一个串联构成，当输入量子比特数n比较大时，庞大的量子线路规模是Grover算法实际应用的主要障碍。本发明的主要目的在于提出一种改善的Grover量子搜索方法，旨在解决降低现有Grover算法的线路复杂度问题。



CN 111291892 B

1.一种量子并行搜索方法,其特征在于,所述方法基于量子并行搜索系统来实现,所述系统包括可调用的:寄存器1、寄存器2、寄存器3、G算子、辅助比特0、辅助比特1、Hadamard门、compare线路、受控U_H门;

辅助比特0的初始化状态为|0>、辅助比特1的初始化状态为|1>;

寄存器对应问题的输入;

G算子包括Oracle量子线路,算子U量子线路;

初始化为|0>状态的辅助比特0,作用是在Oracle算子里控制寄存器中的量子比特;

初始化为|1>状态的辅助比特1辅助翻转解的位置;

Oracle量子线路,用来检查辅助比特的相位来判断x是否为搜索问题的一个解;

算子U量子线路,用来放大搜索问题的解的概率幅值;测量线路,是对算法最后输出状态作测量;

U算子由单位矩阵、Hadamard门和条件相移U_x算子组成;

Hadamard门用于变换$H^{\otimes n_2}$;

条件相移U_x算子的作用是使得状态|0>以外的每一个计算基态获得-1的相位移动;

所述方法包括以下步骤:

S1:构建一个搜索问题;

S2:应用Hadamard门于寄存器2和寄存器3,使得寄存器2和寄存器3为均衡状态;

S3:应用迭代算子G,更新寄存器2和寄存器3的量子状态,增加目标状态的概率幅值同时降低非目标状态的概率幅值;

S4:对寄存器2更新后的状态进行测量;

S5:寻找搜索问题的解。

2.根据权利要求1所述的量子并行搜索方法,其特征在于,S1具体为:假设一个搜索问题f(x),其搜索空间为$N=2^n$,即用n个比特表示其搜索空间大小,将搜索问题表示为一个输入x的函数f(x),则x取值范围是$[0,2^n-1]$,函数f的定义是,若x是一个搜索问题的解,则f(x)=1,否则f(x)=0,如果f(x)有唯一解,为了方便起见,令x_0表示搜索问题的唯一解,则$f(x_0)=1$,当$x\neq x_0$使得f(x)=0;找到f(x)的解时即是找到了一个搜索问题的解;

对于搜索问题f(x),先求出g(x)的唯一解x'_0,再由x'_0求f(x)的x_0;

其中,$g(x)=f(2^{n_3}x+0)\oplus f(2^{n_3}x+1)\oplus f(2^{n_3}x+1)\cdots\oplus f(2^{n_3}x+2^{n_3}-1)$;x的取值范围是$[0,2^{n-n_3}-1]$,符号⊕是模二运算。

3.根据权利要求2所述的量子并行搜索方法,其特征在于,S2包括以下步骤:

S2.1:设每一个经典取值x_1取值范围为$[0,2^{n_1}-1]$,并将其经典值转变为二进制值,存储于寄存器1中;寄存器2初始化为状态$|0\rangle^{\otimes n_2}$,寄存器3初始化为状态$|0\rangle^{\otimes n_3}$;辅助量子比特0初始化为|0>;辅助量子比特1初始化为|1>;

其中,$|0\rangle^{\otimes n_2}$表示n_2个量子比特的状态都是|0>状态;

S2.2:对寄存器2、寄存器3中的量子比特做Hadamard变换,即分别应用$H^{\otimes n_2}$和$H^{\otimes n_3}$于寄存器2和寄存器3,得到系统所需的均衡叠加状态;对辅助量子比特1做Hadamard变换;

S2.3:应用Oracle算子于S2.2所得的状态,若存在解,则标志解的位置,否则保持不变;

S2.4:应用U算子于系统,假设g(x)存在解,则增大解的概率幅,同时减小非解的概率

幅；

其中，U算子由单位矩阵、Hadamard变换$H^{\otimes n_2}$和条件相移U_x算子组成；

S2.5：将步骤S2.3和步骤S2.4整合为一个G算子，G＝UO，使用G算子对S2.2后的系统的量子均衡叠加态进行$[\pi\sqrt{2^{n_2+1}}/4]$次迭代；

其中，0表示Oracle算子。

4.根据权利要求3所述的量子并行搜索方法，其特征在于，在对寄存器2测量之前，再应用一次$H^{\otimes n_2}$于寄存器2；

如果当前子问题不包含g(x)的解x'_0，则测量值为0；如果当前子问题包含g(x)的解x'_0，则测量结果值是随机坍缩到$[0, 2^{n_2}-1]$中的一个值；

若测量值为非零值时，则当前子问题包含x'_0，此时，去掉最后一次$H^{\otimes n_2}$，在当前的x_1值重新执行该系统一次，再测量即可得到x'_0对应于n_2的部分，假设测量值为x''_0，则$x_0^{'} = 2^{n_2}x_1 + x_0^{''}$。

5.根据权利要求4所述的量子并行搜索方法，其特征在于，S5中搜索问题的解采用穷举法进行寻找。

6.根据权利要求1或5所述的量子并行搜索方法，其特征在于，S5包括以下步骤：

S5.1：判断x''_0是否等于0，若不等于0，则$x_0^{'} = 2^{n_2}x_1 + x_0^{''}$，k＝0，并执行S5.2，若等于0，则判断$x_1 < 2^{n_1}-1$，是否成立，若不成立，则表示找不到解，并结束搜索，若成立，则令x_1＝x_1+1，并返回S3；

S5.2：判断$f\left(2^{n_2}x_0^{'}+k\right)=1$是否成立，若成立，则找到解$x_0 = 2^{n_2}x_0^{'} + k$，并结束搜索；若不成立，则执行S5.3；

S5.3：判断$k < 2^{n_3}-1$是否成立，若不成立，则表示找不到解，并结束搜索，若成立，则令k＝k+1，并返回S5.2。

7.根据权利要求6所述的量子并行搜索方法，其特征在于，条件相移U_x算子用来使寄存器2中状态$|0\rangle^{\otimes n_2}$以外的每一个计算基态获得-1的相位移动，U_x算子由2|0><0|-I表示。

8.根据权利要求7所述的量子并行搜索方法，其特征在于，

$U = \left(I_{n_1}\otimes H^{\otimes n_2}\otimes I_{n_3}\right)U_x\left(I_{n_1}\otimes H^{\otimes n_2}\otimes I_{n_3}\right)$。

9.根据权利要求8所述的量子并行搜索方法，其特征在于，O用来标志g(x)的解的位置，使得状态$|x_0'\rangle \rightarrow -|x_0'\rangle$，其他状态则保持不变，其对应的矩阵用$I-2|x_0'\rangle\langle x_0'|$表示。

一种量子并行搜索方法

技术领域

[0001] 本发明涉及信息安全技术领域，更具体地，涉及一种量子并行搜索方法。

背景技术

[0002] 量子计算是计算机科学、数学和物理学交叉的新领域，近几十年来，量子计算已经成为信息研究领域关注的焦点。作为一种新的计算模型，量子计算比经典计算要快得多。它依赖于量子力学原理来获得可满足性问题的解。量子的叠加性是量子计算的一个很重要的特性，以输入n个量子比特为例，量子计算可以一次计算2^n个数据，每个计算结果以一定的概率幅值给出。

[0003] Grover在1996年提出了一种量子搜索算法，Grover算法是目前应用最广泛的量子搜索算法，可以在时间复杂度为$O(\sqrt{N})$的情况下求解一个规模为N的无结构数据库中的搜索问题，Grove算法相对于经典算法进行了平方加速。Grover算法通过反复迭代$\lceil \pi\sqrt{N}/4 \rceil$次G算子，放大目标状态的概率幅，减小非目标状态的概率幅，最终测量其叠加状态将以接近1的概率找到目标状态。反复应用G算子，这相当于在量子线路串联G算子的量子线路，量子线路越复杂，则需要更多的基础量子门电路和量子比特，每设计一个基础量子门电路和量子比特都需要耗费大量资源。而受限于目前所掌握的技术，量子计算机只能配备少量的量子位，对于只配有少量的量子比特的量子计算机，其实际应用也将受到更大的限制。目前最新的量子计算机是谷歌公司旗下的拥有53位量子比特的量子计算机。因此，本发明设计一个改进的Grover算法，采用时间-空间折衷的方法，以牺牲运算时间次数来降低量子线路的复杂度，从而可以减少量子比特数，使得配有少量的量子比特的量子计算机可以得到更广泛的应用。量子计算机可以加速NP完全问题，如3SAT、图着色、旅行商等NP完全问题。

发明内容

[0004] 本发明为克服上述现有技术所述的量子搜索效率低的缺陷，提供一种量子并行搜索方法。

[0005] Grover算法是相对于经典算法是一种二次方加速的量子算法，该算法能够在以运行时间复杂度为O(1)和线路复杂度为$O(\sqrt{N})$的情况下以接近1的概率找到问题的解，其中量子线路由G算子一个接一个串联构成，在输入状态经过$\lceil \pi\sqrt{N}/4 \rceil$次G算子时，再测量最后的叠加态将以接近1的概率找到解。由Grover算法的性质可知，当输入量子比特数n比较大时，庞大的量子线路规模是Grover算法实际应用的主要障碍。

[0006] 本发明所述方法基于量子并行搜索系统来实现，所述系统包括可调用的：寄存器1、寄存器2、寄存器3、G算子、辅助比特0、辅助比特1、Hadamard门、compare线路、受控U_H门；

[0007] 辅助比特0的初始化状态为|0>、辅助比特1的初始化状态为|1>状；

[0008] 寄存器对应问题的输入；

[0009] G算子包括Oracle量子线路，算子U量子线路；

[0010] 初始化为|0>状态的辅助比特0，作用是在Oracle算子里控制寄存器中的量子比特；

[0011] 初始化为|1>状态的辅助比特1辅助翻转解的位置；

[0012] Oracle量子线路，用来检查辅助比特的相位来判断x是否为搜索问题的一个解；

[0013] 算子U量子线路，用来放大搜索问题的解的概率幅值；测量线路，是对算法最后输出状态作测量；还有一些基本的量子门电路；

[0014] U算子由单位矩阵、Hadamard门和条件相移U_x算子组成。

[0015] Hadamard门用于变换$H^{\otimes n_2}$；

[0016] 条件相移U_x算子的作用是使得状态|0>以外的每一个计算基态获得-1的相位移动；

[0017] 所述方法包括以下步骤：

[0018] S1：构建一个搜索问题；

[0019] S2：应用Hadamard门于寄存器2和寄存器3，使得寄存器2和寄存器3为均衡状态；

[0020] S3：应用迭代算子G，更新寄存器2和寄存器3的量子状态，增加目标状态的概率幅值同时降低非目标状态的概率幅值；

[0021] S4：对寄存器2更新后的状态进行测量；

[0022] S5：寻找搜索问题的解。

[0023] 优选地，S1具体为：假设一个搜索问题，其搜索空间为$N=2^n$，即可以用n个比特表示其搜索空间大小，将搜索问题表示为一个输入x的函数f(x)，则x取值范围是$[0,2^n-1]$，函数f的定义是，若x是一个搜索问题的解，则f(x)＝1，否则f(x)＝0，如果f(x)有唯一解，为了方便起见，令x_0表示搜索问题的唯一解，则$f(x_0)=1$，当$x\neq x_0$使得f(x)＝0；找到f(x)的解时即是找到了一个搜索问题的解。

[0024] 对于一个搜索问题f(x)，可以先求出g(x)的唯一解x'_0，再由x'_0求f(x)的x_0；

[0025] 其中，$g(x)=f(2^{n_3}x+0)\oplus f(2^{n_3}x+1)\oplus f(2^{n_3}x+1)\cdots\oplus f(2^{n_3}x+2^{n_3}-1)$；x的取值范围是$[0,2^{n-n_3}-1]$，符号$\oplus$是模二运算(异或)；

[0026] 优选地，S2包括以下步骤：

[0027] S2.1：设每一个经典取值x_1取值范围为$[0,2^{n_1}-1]$，并将其经典值转变为二进制值，存储于寄存器1中；寄存器2初始化为状态$|0\rangle^{\otimes n_2}$，寄存器3初始化为状态$|0\rangle^{\otimes n_3}$；辅助量子比特0初始化为|0>；辅助量子比特1初始化为|1>；

[0028] 其中，$|0\rangle^{\otimes n_2}$表示n_2个量子比特的状态都是|0>状态；

[0029] S2.2：对寄存器2、寄存器3中的量子比特做Hadamard变换，即分别应用$H^{\otimes n_2}$和$H^{\otimes n_3}$于寄存器2和寄存器3，得到系统所需的均衡叠加状态；对辅助量子比特1做Hadamard变换；

[0030] S2.3：应用Oracle算子于S2.2所得的状态，若存在解，则标志解的位置，否则保持不变；

[0031] S2.4：应用U算子于系统，假设g(x)存在解，则增大解的概率幅，同时减小非解的概率幅；

[0032] 其中，U算子由单位矩阵、Hadamard变换$H^{\otimes n_2}$和条件相移U_x算子组成；

[0033] S2.5：将S2.3和S2.4整合为一个G算子，G＝UO，使用G算子对S2.2后的系统的量子均衡叠加态进行$[\pi\sqrt{2^{n_2+1}}/4]$次迭代；

[0034] 其中，O表示Oracle算子；

[0035] 优选地，在对寄存器2测量之前，再应用一次$H^{\otimes n_2}$于寄存器2；

[0036] 如果当前子问题不包含g(x)的解x'_0，则测量值为0；如果当前子问题包含g(x)的解x'_0，则测量结果值是随机坍缩到$[0, 2^{n_2}-1]$中的一个值；

[0037] 若测量值为非零值时，则当前子问题包含x'_0，此时，去掉最后一次$H^{\otimes n_2}$，在当前的x_1值重新执行该系统一次，再测量即可得到x'_0对应于n_2的部分，假设测量值为x''_0，则$x'_0 = 2^{n_2}x_1 + x''_0$。

[0038] 优选地，S5中搜索问题的解采用穷举法进行寻找。

[0039] 优选地，S5包括以下步骤：

[0040] S5.1：判断x''_0是否等于0，若不等于0，则$x'_0 = 2^{n_2}x_1 + x''_0$，k＝0，并执行S5.2，若等于0，则判断$x_1 < 2^{n_1} - 1$，是否成立，若不成立，则表示找不到解，并结束搜索，若成立，则令$x_1 = x_1 + 1$，并返回S3；

[0041] S5.2：判断$f(2^{n_2}x'_0 + k) = 1$是否成立，若成立，则找到解$x_0 = 2^{n_2}x'_0 + k$，并结束搜索；若不成立，则执行S5.3；

[0042] S5.3：判断$k < 2^{n_3} - 1$是否成立，若不成立，则表示找不到解，并结束搜索，若成立，则令k＝k+1，并返回S5.2。

[0043] 优选地，条件相移U_x算子的作用是使得寄存器2中状态$|0\rangle^{\otimes n_2}$以外的每一个计算基态获得-1的相位移动，U_x算子由$2|0\rangle\langle 0| - I$表示。

[0044] 优选地，$U = (I_{n_1} \otimes H^{\otimes n_2} \otimes I_{n_3}) U_x (I_{n_1} \otimes H^{\otimes n_2} \otimes I_{n_3})$

[0045] 优选地，O的作用是标志g(x)的解的位置，使得状态$|x_0'\rangle \rightarrow -|x_0'\rangle$，其他状态则保持不变，其对应的矩阵可以用$I-2|x_0'\rangle\langle x_0'|$表示。

[0046] 与现有技术相比，本发明技术方案的有益效果是：现有的Grover算法的线路复杂度为$O(2^{n/2})$。而改进的Grover算法，如果将Oracle算子看成一个单元，则改进的算法量子线路复杂度为$O(2^{n/4})$，运行时间复杂度为$O(2^{n/4})$，因此，减少了大量的量子门的应用和所需的量子比特数量，使得配有少量的量子计算机可以得到更为广泛的应用。

附图说明

[0047] 图1为量子并行搜索方法的流程图；

[0048] 图2为具有通用输入的Grover算法的线路框架图；

[0049] 图3为改进Grover量子搜索算法的线路框架图；

[0050] 图4为改进Grover算法的Oracle算子线路框架图；

[0051] 图5为Grover量子搜索算法的几何过程描述图。

具体实施方式

[0052] 附图仅用于示例性说明，不能理解为对本专利的限制；

[0053] 为了更好说明本实施例，附图某些部件会有省略、放大或缩小，并不代表实际产品的尺寸；

[0054] 对于本领域技术人员来说，附图中某些公知结构及其说明可能省略是可以理解的。

[0055] 下面结合附图和实施例对本发明的技术方案做进一步的说明。

[0056] 实施例1：

[0057] 本实施例提供一种量子并行搜索方法，所述方法基于量子并行搜索系统来实现，所述系统包括可调用的：寄存器1、寄存器2、寄存器3、G算子、辅助比特0、辅助比特1、Hadamard门、compare线路、受控U_H门；

[0058] 辅助比特0的初始化状态为|0>、辅助比特1的初始化状态为|1>状；

[0059] 寄存器对应问题的输入；

[0060] G算子包括Oracle量子线路，算子U量子线路；

[0061] 初始化为|0>状态的辅助比特0，作用是在Oracle算子里控制寄存器中的量子比特；

[0062] 初始化为|1>状态的辅助比特1辅助翻转解的位置；

[0063] Oracle量子线路，用来检查辅助比特的相位来判断x是否为搜索问题的一个解；

[0064] 算子U量子线路，用来放大搜索问题的解的概率幅值；测量线路，是对算法最后输出状态作测量；还有一些基本的量子门电路；

[0065] U算子由单位矩阵、Hadamard门和条件相移U_x算子组成。

[0066] Hadamard门用于变换$H^{\otimes n_2}$；

[0067] 条件相移U_x算子的作用是使得状态|0>以外的每一个计算基态获得-1的相位移动；

[0068] 如图1所示，所述方法包括以下步骤：

[0069] S1：构建一个搜索问题；

[0070] S2：应用Hadamard门于寄存器2和寄存器3，使得寄存器2和寄存器3为均衡状态；

[0071] S3：应用迭代算子G，更新寄存器2和寄存器3的量子状态，增加目标状态的概率幅值同时降低非目标状态的概率幅值；

[0072] S4：对寄存器2更新后的状态进行测量；

[0073] S5：寻找搜索问题的解。

[0074] 具体来说，首先假设函数f(x)表示一个搜索问题，若f(x)＝1则表示找到解，f(x)＝0则找不到解，其搜索空间为$N=2^n$的问题，即x的取值范围是$[0,2^n-1]$，如果f(x)有唯一解x_0使得$f(x_0)=1$，x的其他取值都使得f(x)＝0。Grover算法可以在线路复杂度为$O(\sqrt{N})$的情况下找到f(x)的唯一解x_0，当输入比特数n比较大时，量子线路消耗过大，为了降低线路规模，考虑将n个输入比特分成两部分：n_1和n_2，其中$n_1+n_2=n$并且n_1为经典比特。经典比特n_1的输入范围是$[0,2^{n_1}-1]$，可以将原来的问题转为2^{n_1}个子搜索问题，每个子问题对应于n_2个量子比特的输入的Grover搜索算法。对于每一个经典取值$x_1 \in [0,2^{n_1}-1]$，算法的子搜索

空间为$X_1 = \{2^{n_2}x_1, 2^{n_2}x_1 + 1, \cdots, 2^{n_2}(x_1 + 1) - 1\}$，对于$x_1$的全部取值，即$\bigcup_{x_1=0}^{2^{n_1}-1} X_1 = X$，其对应搜索空间为原来问题的搜索空间$2^n$，因此，f(x)的解$x_0$必定存在于其中一个搜索子问题。该方法可以在运行时间复杂度为$O(2^{n_1})$和线路复杂度为$O(2^{n_2/2})$的情况下以接近1的概率找到问题的解。假设n_1取值为n/3，n_2取值为2n/3，(若n不能被3整除，则取$n_1 = n - n_2,\ n_2 = \lceil 2n/3 \rceil$)，则该方法的运行时间复杂度为$O(2^{n/3})$和线路复杂度为$O(2^{n/3})$。因此，相比于原来的Grover算法，该方法将量子线路从$O(2^{n/2})$降至$O(2^{n/3})$。为实现该算法，需要准备两个寄存器和一个辅助量子比特，寄存器1存储n_1个经典比特，寄存2存储n_2个量子比特。整个量子线路图如图2所示。

[0075] 步骤1.x_1取值范围为$[0, 2^{n_1} - 1]$，并将其经典值转变为二进制值，存储于寄存器1中；寄存器2初始化为状态$|0\rangle^{\otimes n_2}$；辅助量子比特初始化为|1>。

[0076] 步骤2.对寄存器2中的量子比特做变换，即应用Hadamard变换$H^{\otimes n_2}$于寄存器2，得到系统所需的均衡叠加状态；对辅助量子比特做Hadamard变换H得到辅助量子比特的叠加态。

[0077] 步骤3.应用Oracle算子于步骤2所得的状态，Oracle算子可以查找系统的均衡叠加态中是否存在问题解的状态，若存在所述问题解的状态，则标记所述问题解的状态，若不存在所述问题解的状态，则保持原状态，即Oracle算子对系统不起任何作用。

[0078] 步骤4.对在Oracle算子应用于系统后得到的状态，应用Hadamard变换$H^{\otimes n_2}$于寄存器2。

[0079] 步骤5.在系统执行条件相移U_x算子，使得寄存器2中状态$|0\rangle^{\otimes n_2}$以外的每一个计算基态获得-1的相位移动。

[0080] 步骤6.对在U_x算子应用于系统后得到的状态，应用Hadamard变换$H^{\otimes n_2}$于寄存器2。

[0081] 对于步骤4、5、6所做的算子运算可以写成算子U，该算子亦称为关于均值的反演运算。将Oracle算子和U结合写成G算子，该算子对应于Grover算法中的G算子。使用G算子对步骤2后的系统的量子均衡叠加态进行$\lceil \pi\sqrt{2^{n_2}}/4 \rceil$次迭代并在每次迭代后更新系统的叠加态。在对寄存器2测量之前，再应用一次$H^{\otimes n_2}$于寄存器2，如果当前子问题不包含原问题的解，测量得到的结果值一定是0。如果测量结果是非零值，则去掉最后一次变换$H^{\otimes n_2}$，在不改变当前x_1的值，重新执行一次该系统再测量，假设测量值为x′，则$x_0 = 2^{n_2}x_1 + x'$。

[0082] 为了更进一步降低量子线路规模，对于搜索空间为$N=2^n$的f(x)，将n个输入比特分为三个部分：n_1、n_2和n_3，其中$n_1+n_2+n_3=n$。对于原问题f(x)，当且仅当x的取值为x_0时，使得$f(x_0)=1$，对于$x \neq x_0$，f(x)=0。设计一个布尔函数g(x)。

[0083] $$g(x) = f(2^{n_3}x + 0) \oplus f(2^{n_3}x + 1) \oplus f(2^{n_3}x + 1) \cdots \oplus f(2^{n_3}x + 2^{n_3} - 1)$$

[0084] 其中，x的取值范围是$[0, 2^{n-n_3} - 1]$，符号⊕是模二运算(异或)。因此，对于f(x)存在唯一解x_0，使得$f(x_0)=1$，相应地，对应g(x)也同样存在唯一的值x_0'，使得$g(x_0')=1$，对于$x \neq x_0'$，则g(x)=0。假设可以找到x_0'，则同样可以找到原问题的解x_0，$x_0 = 2^{n_3}x_0' + k$，其中$k \in [0, 2^{n_3} - 1]$。为了找到具体的解x_0，可以通过简单的穷举搜索计算$f(2^{n_3}x_0' + k)$是否

等于1来确定k值，其运行时间复杂度为$O(2^{n_3})$。

[0085] 为了获得布尔函数g(x)的唯一解x'_0使得g(x)＝1，利用前面所述的方法，n_1为经典比特，其取值范围是$[0,2^{n_1}-1]$，因此，可以将求解布尔函数g(x)的问题转为2^{n_1}个子搜索问题，每个子问题对应于n_2个量子比特的输入的量子搜索算法，则x'_0必定存在于其中一个子问题中。因此，可以在运行时间复杂度为$O(2^{n_1})$和线路复杂度为$O(2^{n_2/2})$的情况下以接近1的概率找到g(x)的唯一解x'_0。在获得g(x)的解x'_0之后，可以通过简单的穷举搜索计算$f(2^{n_3}x'_0+k)$来找到f(x)的解$x_0=2^{n_3}x'_0+k$，其运算时间复杂度为$O(2^{n_3})$。假设n_1取值为n/4，n_2取值为n/2，n_3取值为n/4(若n不能被4整除，则取$n_1=[n/4],n_2=[n/2],n_3=n-n_1-n_2$)，如果将Oracle算子看做一个单元，则该方法寻找g(x)的解x'_0的时间复杂度为$O(2^{n/4})$和线路复杂度为$O(2^{n/4})$，最后再以运算时间复杂度为$O(2^{n/4})$简单穷举搜索获得f(x)的解x_0。实现该算法的整个量子线路框架图如图3所示，其中寄存器1存储n_1个经典比特，寄存2存储n_2个量子比特，寄存3存储n_3个量子比特，辅助比特0主要作用是在Oracle算子里控制寄存器3中的量子比特，辅助比特1和前面所述方法一致。该算法与前面述所方法的步骤类似，下面作简要概述。对于图3，将最里面的虚线框包含两个Hadamard变换$H^{\otimes n_2}$和条件相移U_x算子写成一个U算子，U算子主要作用是对寄存器2中的量子比特作均值的反演运算，条件相移U_x算子与前面述所一样。

[0086] 步骤1.x_1取值范围为$[0,2^{n_1}-1]$，并将其经典值转变为二进制值，存储于寄存器1中；寄存器2初始化为状态$|0\rangle^{\otimes n_2}$，寄存器3初始化为状态$|0\rangle^{\otimes n_3}$；辅助量子比特0初始化为|0>；辅助量子比特1初始化为|1>。

[0087] 步骤2.对寄存器2、寄存器3中的量子比特做Hadamard变换，即分别应用$H^{\otimes n_2}$和$H^{\otimes n_3}$于寄存器2和寄存器3，得到系统所需的均衡叠加状态；对辅助量子比特1做Hadamard变换。

[0088] 步骤3.应用Oracle算子于步骤2所得的状态，Oracle算子主要是标志g(x)的解x'_0的位置，如果不存在解，则保持原状态，即Oracle算子对系统不起任何作用。

[0089] 步骤4.应用U算子于系统，假设g(x)存在解，则增大解的概率幅，同时减小非解的概率幅。

[0090] 对于步骤3和步骤4，可写成一个G算子，使用G算子对步骤2后的系统的量子均衡叠加态进行$[\pi\sqrt{2^{n_2+1}}/4]$次迭代。与前面所述方法一样，在对寄存器2测量之前，再应用一次$H^{\otimes n_2}$于寄存器2。如果当前子问题不包含g(x)的解x'_0，则测量值为0；如果当前子问题包含g(x)的解x'_0，则测量结果值是随机坍缩到$[0,2^{n_2}-1]$中的一个值。若测量值为非零值时，则当前子问题包含x'_0，此时，只需去掉最后一次$H^{\otimes n_2}$，在当前的x_1值重新执行该系统一次，再测量即可得到x'_0对应于n_2的部分，假设测量值为x''_0，则$x'_0=2^{n_2}x_1+x''_0$。在获得x'_0的基础下可以通过简单的穷举法寻找f(x)的解x_0。

[0091] 作为一个具体的实施例，下面结合具体情况进行比较分析：

[0092] 对于一个搜索问题f(x)，假设存在唯一解x_0使得f(x_0)＝1，x的其他取值都使得f

(x)＝0,其搜索空间为N＝2^n,n为输入比特数量,我们的任务是在N个待搜索元素中找到x_0。结合图2,将n个输入比特分成两部分:n_1和n_2,其中n_1+n_2＝n,即将原来的问题转为2^{n_1}个子搜索问题,每个子问题对应于n_2个量子比特的输入的Grover搜索算法。以$|\varphi_1\rangle$, $|\varphi_2\rangle$, $|\varphi_3\rangle$, $|\varphi_4\rangle$分别表示系统中的初始状态,初始状态经过Hadamard门线路的状态,经过Oracle算子的状态,经过相位概率幅值放大后的状态。初始化寄存器1,2和辅助量子比特,其中x_1取值范围为$[0, 2^{n_1}-1]$,将x_1的值转为二进制值,并将其对应的位从高到低存储于寄存器1中;寄存器2初始化为状态$|0\rangle^{\otimes n_2}$;辅助量子比特初始化为|1>。以$|\varphi_1\rangle$表示系统的初始化状态,则:

[0093] $$|\varphi_1\rangle = |x_1\rangle|0\rangle^{\otimes n_2}|1\rangle$$

[0094] 将系统的初始状态$|\varphi_1\rangle$对应的n_2量子比特经过Hadamard变换$H^{\otimes n_2}$,则在寄存器2中可得到对应n_2量子比特的均衡叠加态$\frac{1}{\sqrt{2^{n_2}}}\sum_{x=0}^{2^{n_2}-1}|x\rangle$;辅助比特经过Hadamard变换H,得到状态$\frac{|0\rangle-|1\rangle}{\sqrt{2}}$。以$|\varphi_2\rangle$表示此时对应的状态。

[0095] $$|\varphi_2\rangle = I_{n_1}\otimes H^{\otimes n_2}\otimes H|\varphi_1\rangle = |x_1\rangle\frac{1}{\sqrt{2^{n_2}}}\sum_{x=0}^{2^{n_2}-1}|x\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

[0096] 其中I_{n_1}表示单位矩阵。应用Oracle算子于系统,其可以标志f(x)的解的位置,即Oracle可以使得状态$|x_0\rangle\rightarrow -|x_0\rangle$,对于其他状态则保持不变,其对应的矩阵可以用$I-2|x_0\rangle\langle x_0|$表示。用O表示Oracle算子,则:

[0097] $$|\varphi_3\rangle = O|\varphi_2\rangle = |x_1\rangle\frac{1}{\sqrt{2^{n_2}}}\sum_{x=0}^{2^{n_2}-1}(-1)^{f(x)}|x\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

[0098] 条件相移U_x算子的作用是使得寄存器2中状态$|0\rangle^{\otimes n_2}$以外的每一个计算基态获得-1的相位移动,U_x算子可以由$2|0\rangle\langle 0|-I$表示。将U_x算子与其两边的Hadamard变换以U表示,则$U = (I_{n_1}\otimes H^{\otimes n_2})U_x(I_{n_1}\otimes H^{\otimes n_2})$。

[0099] $|\varphi_4\rangle$表示应用U于状态$|\varphi_3\rangle$

$$|\varphi_4\rangle = U|\varphi_3\rangle = |x_1\rangle\sum_{x=0}^{2^{n_2}-1}[-a_x+2\langle a\rangle]|x\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

[0100]
$$= |x_1\rangle\{\left[\frac{1}{\sqrt{2^{n_2}}}+\frac{2}{2^{n_2}}\frac{1}{\sqrt{2^{n_2}}}(2^{n_2}-2)\right]|x_0\rangle$$
$$+\sum_{x\neq x_0}[-\frac{1}{\sqrt{2^{n_2}}}+\frac{2}{2^{n_2}}\frac{1}{\sqrt{2^{n_2}}}(2^{n_2}-2)]\}\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

[0101] 式中$a_x = \frac{1}{\sqrt{2^{n_2}}}(-1)^{f(x)}$表示状态|x>的幅值,$\langle a\rangle = \frac{1}{2^{n_2}}\sum_{x=0}^{2^{n_2}-1}a_x$表示叠加态的幅值的平均值。对于图2中的外层虚线框部分以算子G表示,即G＝U0。对于每个

$x_1 \in \{0,1,2,\cdots,2^{n_1}-1\}$，在系统中重复应用G算子$r=\lfloor \pi\sqrt{2^{n_2}}/4 \rfloor$次。以$|\varphi\rangle$表示：

[0102]
$$|\varphi\rangle = G^r|\varphi_2\rangle = G^r|x_1\rangle\frac{1}{\sqrt{2^{n_2}}}\sum_{x=0}^{2^{n_2}-1}|x\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$
$$= |x_1\rangle\sum_{x=0}^{2^{n_2}-1}[-a_{x,r}+2\langle a_r\rangle]|x\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

[0103] 其中$a_{x,r}$是第r迭代对应状态|x>的幅值，

[0104] $a_{x,r}=-a_{x,r-1}+2\langle a_{r-1}\rangle$

[0105] $$\langle a_r\rangle = \frac{1}{2^{n_2}}\sum_{x=0}^{2^{n_2}-1}a_{x,r}$$

[0106] 当r=1时，

[0107] $$a_{x,1} = \frac{1}{\sqrt{2^{n_2}}}(-1)^{f(x)}$$

[0108] $$\langle a_1\rangle = \frac{1}{2^{n_2}}\sum_{x=0}^{2^{n_2}-1}a_{x,1}$$

[0109] 如果当前子问题不包含f(x)的解x_0，即$x_0 \notin X_1$，$X_1=\{2^{n_2}x_1, 2^{n_2}x_1+1,\cdots,2^{n_2}(x_1+1)-1\}$，则：

[0110] $$|\varphi\rangle = |x_1\rangle\frac{1}{\sqrt{2^{n_2}}}\sum_{x=0}^{2^{n_2}-1}|x\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

[0111] 如果当前子问题包含f(x)的解x_0，即$x_0 \in X_1$，则：

[0112] $$|\varphi\rangle = [-a_{x,r}+2\langle a_r\rangle]|x_0\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}} \approx |x_0\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

[0113] 在使用G算子对系统的量子均衡叠加态进行$\lfloor \pi\sqrt{2^{n_2}}/4 \rfloor$次迭代后，对寄存器2的量子叠加态进行测量，如果当前子搜索问题包含原问题的解x_0，测量结果将于接近1的概率得到解x_0对应于n_2的部分，假设测量值为x′，则$x_0 = 2^{n_2}x_1 + x'$。若当前子搜索问题不包含原问题的解x_0，则测量的结果值是量子均衡叠加态随机坍缩到$[0, 2^{n_2}-1]$中的一个值。因此，这不能区分当前子问题是否包含原问题的解。考虑到多数情况下，当前子问题是不包含原问题的解，因此，在对寄存器2测量之前，再应用一次Hadamard变换$H^{\otimes n_2}$于寄存器2。如果当前子问题不包含原问题的解，在最后应用$H^{\otimes n_2}$到系统中之前，寄存器2的量子状态是均衡叠加状态，因此，应用$H^{\otimes n_2}$于寄存器2，测量得到的结果值一定是0；如果当前子问题包含原问题的解，则测量结果值是随机坍缩到$[0, 2^{n_2}-1]$中的一个值。由此可知，当测量结果为非零值时，可以判断当前子问题包含解，此时，只需去掉最后一次Hadamard变换$H^{\otimes n_2}$，在当前的x_1值重新执行该系统一次，再测量即可得到x_0对应于n_2的部分，假设测量值为x′，则

$x_0 = 2^{n_2}x_1 + x'$。如果测量结果为0值，则取x_1的下一个值再次执行此系统。总的来说，该方法可以在运行时间复杂度为$O(2^{n_1})$和线路复杂度为$O(2^{n_2/2})$的情况下找到问题的解。

[0114] 值得注意的是，如果当前子问题包含原问题的解，由该系统得到叠加态经测量后，测量值有可能为0值，此时，该方法无法找到正确的解，由于测量状态是随机坍缩到$[0, 2^{n_2}-1]$中的其中一个状态，而测量之前的寄存器2中的量子状态接近于均匀叠加状态，因此，由于随机坍缩到0值而找不到正确的解的概率约等于$1/2^{n_2}$。

[0115] 为了更进一步降低量子线路规模，对于一个搜索问题f(x)，可以先求出g(x)的唯一解x'_0，再由x'_0求f(x)的x_0。结合图3，将n个输入比特分成两部分：n_1，n_2和n_3，其中$n_1+n_2+n_3=n$。以$|\varphi_1\rangle$，$|\varphi_2\rangle$，$|\varphi_3\rangle$分别表示系统中的初始状态，初始状态经过Hadamard门线路的状态，经过r此迭代算子G的状态。以x_1对应的二进制初始化寄存器1；寄存器2、3分别初始化为$|0\rangle^{\otimes n_2}$，$|0\rangle^{\otimes n_3}$；辅助比特0、1分别初始化为|0>，|1>，辅助比特0的主要作用是控制寄存器3(以下分析过程忽略辅助比特0，这并不影响结果)，则：

[0116] $$|\varphi_1\rangle = |x_1\rangle|0\rangle^{\otimes n_2}|0\rangle^{\otimes n_3}|1\rangle$$

[0117] $$|\varphi_2\rangle = I_{n_1}\otimes \mathrm{H}^{\otimes n_2}\otimes \mathrm{H}^{\otimes n_3}\otimes \mathrm{H}|\varphi_1\rangle = |x_1\rangle\frac{1}{\sqrt{2^{n_2}}}\sum_{x=0}^{2^{n_2}-1}|x\rangle\frac{1}{\sqrt{2^{n_3}}}\sum_{y=0}^{2^{n_3}-1}|y\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

[0118] 其中，y为对应寄存器3中的量子状态；

[0119] 以O表示Oracle算子，这里O的主要作用是标志g(x)的解的位置，使得状态$|x_0'\rangle \rightarrow -|x_0'\rangle$，其他状态则保持不变，其对应的矩阵可以用$I-2|x_0'\rangle\langle x_0'|$表示。以U表示条件相移$U_x$算子与其两边的Hadamard变换的结果，即：

[0120] $$U = \left(I_{n_1}\otimes \mathrm{H}^{\otimes n_2}\otimes I_{n_3}\right)U_x\left(I_{n_1}\otimes \mathrm{H}^{\otimes n_2}\otimes I_{n_3}\right)$$

[0121] 对于图3将虚线框对应算子G，则G＝UO。$|\varphi_3\rangle$表示系统经过迭代G算子$r = \lceil \pi\sqrt{2^{n_2+1}}/4 \rceil$次对应的状态。

[0122] $$|\varphi_3\rangle = G^r|\varphi_2\rangle = |x_1\rangle\frac{1}{\sqrt{2^{n_2}}}\sum_{x=0}^{2^{n_2}-1}|x\rangle\frac{1}{\sqrt{2^{n_3}}}\sum_{y=0}^{2^{n_3}-1}|y\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

[0123] $$= |x_1\rangle\sum_{x=0}^{2^{n_2}-1}[-a_{x,r}+2\langle a_r\rangle]|x\rangle\frac{1}{\sqrt{2^{n_3}}}\sum_{y=0}^{2^{n_3}-1}|y\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

[0124] 当r＝1时，$a_{x,1} = \frac{1}{\sqrt{2^{n_2}}}(-1)^{g(x)}$，$\langle a_1\rangle = \frac{1}{2^{n_2}}\sum_{x=0}^{2^{n_2}-1}a_{x,1}$。如果当前子问题不包含g(x)的解$x_0'$，即$x_0' \notin X_1$，则：

[0125] $$|\varphi_3\rangle = |x_1\rangle\frac{1}{\sqrt{2^{n_2}}}\sum_{x=0}^{2^{n_2}-1}|x\rangle\frac{1}{\sqrt{2^{n_3}}}\sum_{y=0}^{2^{n_3}-1}|y\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

[0126] 如果当前子问题包含g(x)的解x_0'，即$x_0' \in X_1$，则：

[0127] $$|\varphi_3\rangle \approx \left[-a_{x,r}+2\langle a_r\rangle\right]|x_0'\rangle\frac{1}{\sqrt{2^{n_3}}}\sum_{y=0}^{2^{n_3}-1}(-1)^{\langle y,y_0\rangle}|y\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

$$\approx |x_0'\rangle\frac{1}{\sqrt{2^{n_3}}}\sum_{y=0}^{2^{n_3}-1}(-1)^{\langle y,y_0\rangle}|y\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}$$

[0128] 其中$\langle y,y_0\rangle=(y^0\cap {y_0}^0)\oplus(y^1\cap {y_0}^1)\oplus\cdots\oplus(y^{2^{n_3}-1}\cap {y_0}^{2^{n_3}-1})$；

[0129] y^i表示状态|y>对应的i位的状态，y_0表示搜索问题的解对应寄存器3中的n_3比特的部分。

[0130] 在系统经过r次迭代后，对寄存器2进行测量，和前面所述方法一样，在测量之前，再应用一次Hadamard变换$H^{\otimes n_2}$于寄存器2。如果当前子问题不包含g(x)的解x_0'，则测量结果值一定是0；如果当前子问题包含x_0'，则去掉最后一次Hadamard变换$H^{\otimes n_2}$，在当前的x_1值重新执行该系统一次，再测量即可得到x_0'对应于n_2的部分，假设测量值为x′，则$x_0'=2^{n_2}x_1+x'$。在获得g(x)的解x_0'后，以简单的穷举法计算$f(2^{n_3}x_0'+k)$是否等于1从而确定k值，若等于1即找到原来搜索问题的解$x_0=2^{n_3}x_0'+k$。

[0131] 图4为该算法对应的Oracle的线路，比较器(线路compare)的主要作用是对比状态$|0\rangle^{\otimes n_3}$和寄存器3的结果。如果当前子问题不包含x_0'，则此时寄存器3中的量子状态是$|0\rangle^{\otimes n_3}$，经过比较器后，使得辅助量子比特0从状态|0>变为状态|1>从而使得受控U_H门应用于寄存器3对应的量子比特，受控U_H门的主要作用是在当前子问题不包含解的情况下，系统进行下一次迭代时保证寄存器3中的量子比特是均衡叠加状态；辅助量子比特1则保持不变。如果当前子问题包含x_0'，则量子比特0保持不变，从而受控U_H门不对系统起作用；辅助量子比特1则从状态$\frac{|0\rangle-|1\rangle}{\sqrt{2}}$变为状态$\frac{|1\rangle-|0\rangle}{\sqrt{2}}$从而标志g(x)的解。如果当前子问题包含$x_0'$，以$|y_0\rangle$表示系统经过图4里面虚线框的线路对应寄存器3的量子状态，以$|y_0'\rangle$表示状态$|y_0\rangle$经过$H^{\otimes n_3}$的状态，则：

[0132] $$|y_0'\rangle=H^{\otimes n_3}|y_0\rangle\approx\frac{1}{\sqrt{2^{n_3}}}\sum_{y=0}^{2^{n_3}-1}(-1)^{\langle y,y_0\rangle}|y\rangle$$

[0133] 其对应的状态有一半的量子比特获得-1的相位移动。该算法整个Oracle算子的作用是标志g(x)的解x_0'，效果等价于在n_2个量子搜索空间中标志出解，如果在n_2+n_3个量子搜索空间中标志，则相当于简单地标志了2^{n_3}个解，量子线路的复杂度为$O\left(\sqrt{\frac{2^{n_2+n_3}}{2^{n_3}}}\right)=O\left(2^{n_2/2}\right)$。由于$|y_0'\rangle$有一半的量子比特获得-1的相位移动，等效于标志了$2^{n_3-1}$个解，因此，量子线路复杂度应为$O\left(\sqrt{\frac{2^{n_2+n_3}}{2^{n_3-1}}}\right)=O\left(\sqrt{2^{n_2+1}}\right)$，因此，在该系统中需要迭代$r=\lfloor\pi\sqrt{2^{n_2+1}}/4\rfloor$次G算子，如果忽略系数$\sqrt{2}$，则线路复杂度为$O\left(2^{n_2/2}\right)$。由图4可知，在Oracle算子里，里面的G算子迭代了$O\left(2^{n_3/2}\right)$，因此整个量子线路复杂度为$O\left(2^{n_2/2}\cdot 2^{n_3/2}\right)$，

通常n_3的占比较小，假设将Oracle算子看为一个单元，则线路复杂度为$O\left(2^{n_2/2}\right)$。最后，在获得g(x)的解x_0'时，通过简单穷举确定k值，即可确定原问题的解$x_0 = 2^{n_3}x_0' + k$。由于该算法是基于前面所述方法实现的，因此，在对寄存器2测量时，则会以概率为$1/2^{n_2}$得到的结果为0，这意味着该算法找不到解。

[0134] 此外，Grover算法本身就是一种概率型搜索算法，因此，对于这两种方法还存在Grover算法本身对应的测量误差，假设f(x)存在唯一解x_0，以$|\alpha\rangle = \frac{1}{\sqrt{N-1}}\sum_{x\neq x_0}|x\rangle$表示非解，$|\beta\rangle = |x_0\rangle$表示解，则系统的均衡叠加状态：

[0135] $$|\varphi\rangle = \frac{\sqrt{N-1}}{\sqrt{N}}|\alpha\rangle + \frac{1}{\sqrt{N}}|\beta\rangle$$

[0136] $|\alpha\rangle$和$|\beta\rangle$张量成一个二维空间，如图5所示。在系统经过r次的迭代算子G，状态$G^r|\varphi\rangle$接近$|\beta\rangle$，两者之间存在一定的夹角。令：

[0137] $$\cos\frac{\theta}{2} = \frac{\sqrt{N-1}}{\sqrt{N}},$$

[0138] $$\sin\frac{\theta}{2} = \frac{1}{\sqrt{N}},$$

[0139] 则：

[0140] $$|\varphi\rangle = \cos\frac{\theta}{2}|\alpha\rangle + \sin\frac{\theta}{2}|\beta\rangle$$

[0141] 在系统上应用G，则：

[0142] $$G|\varphi\rangle = \cos\frac{3\theta}{2}|\alpha\rangle + \sin\frac{3\theta}{2}|\beta\rangle$$

[0143] 在系统迭代r次G，

[0144] $$G^r|\varphi\rangle = \cos\frac{(2r+1)\theta}{2}|\alpha\rangle + \sin\frac{(2r+1)\theta}{2}|\beta\rangle$$

[0145] 当N很大时，$\theta \approx \sin\theta \approx \frac{2}{\sqrt{N}}$，因此最终状态$|\varphi'\rangle$和$|\beta\rangle$的角误差至多为$\frac{\theta}{2} \approx \frac{1}{\sqrt{N}}$测量最终的误差概率为$\frac{1}{N}$，因此，对于上述两种方法，存在$1/2^{n_2}$的误差概率。为了使$|\varphi'\rangle$最接近$|\beta\rangle$，则：

[0146] $$\sin\frac{(2r+1)\theta}{2} \approx 1$$

[0147] 解得$r = \lfloor\pi\sqrt{N}/4\rfloor$。

[0148] 以上所述都是针对f(x)只有一个解x_0，假设f(x)存在M个解，以$|X_0| = M$表示解空间，则：

[0149] $$|\alpha\rangle = \frac{1}{\sqrt{N-M}}\sum_{x\notin X_0}|x\rangle$$

[0150] $$|\beta\rangle = \frac{1}{\sqrt{M}} \sum_{x \in X_0} |x\rangle$$

[0151] $$|\varphi\rangle = \sqrt{\frac{N-M}{N}} |\alpha\rangle + \sqrt{\frac{M}{N}} |\beta\rangle$$

[0152] 令：

[0153] $$\cos\frac{\theta}{2} = \sqrt{\frac{N-M}{N}}$$

[0154] $$\sin\frac{\theta}{2} = \sqrt{\frac{M}{N}}$$

[0155] 在系统迭代r次G，

[0156] $G^r|\varphi\rangle = \cos\frac{(2r+1)\theta}{2}|\alpha\rangle + \sin\frac{(2r+1)\theta}{2}|\beta\rangle$ ；

[0157] 当M<<N时，$\theta \approx \sin\theta \approx 2\sqrt{\frac{M}{N}}$，因此，

[0158] $$\sin\frac{(2r+1)\theta}{2} \approx 1$$

[0159] 解得$r = \lceil \pi/4\sqrt{N/M} \rceil$。

[0160] 对于第二个算法，由于⊕算子不能保证g(x)等于1，因此，需要重新设计g(x)，令$g(x) = f(2^{n_3}x + 0) \cup f(2^{n_3}x + 1) \cup f(2^{n_3}x + 1) \cdots \cup f(2^{n_3}x + 2^{n_3} - 1)$。

[0161] 可以保证当f(x)存在解时，则g(x)＝1。以上所述算法都是将原来搜索问题转化为2^{n_1}个子问题，不妨假设，每个子问题对应有$M/2^{n_1}$个解，令$M' = \max(\frac{M}{2^{n_1}}, 1)$，则对每子问题，需要迭代$O\left(\sqrt{2^{n_2}/M'}\right)$次G。当M<<N时，系统量子线路迭代次数约等于$O\left(\sqrt{2^{n_2}}\right)$。

[0162] 附图中描述位置关系的用语仅用于示例性说明，不能理解为对本专利的限制；

[0163] 显然，本发明的上述实施例仅仅是为清楚地说明本发明所作的举例，而并非是对本发明的实施方式的限定。对于所属领域的普通技术人员来说，在上述说明的基础上还可以做出其它不同形式的变化或变动。这里无需也无法对所有的实施方式予以穷举。凡在本发明的精神和原则之内所作的任何修改、等同替换和改进等，均应包含在本发明权利要求的保护范围之内。

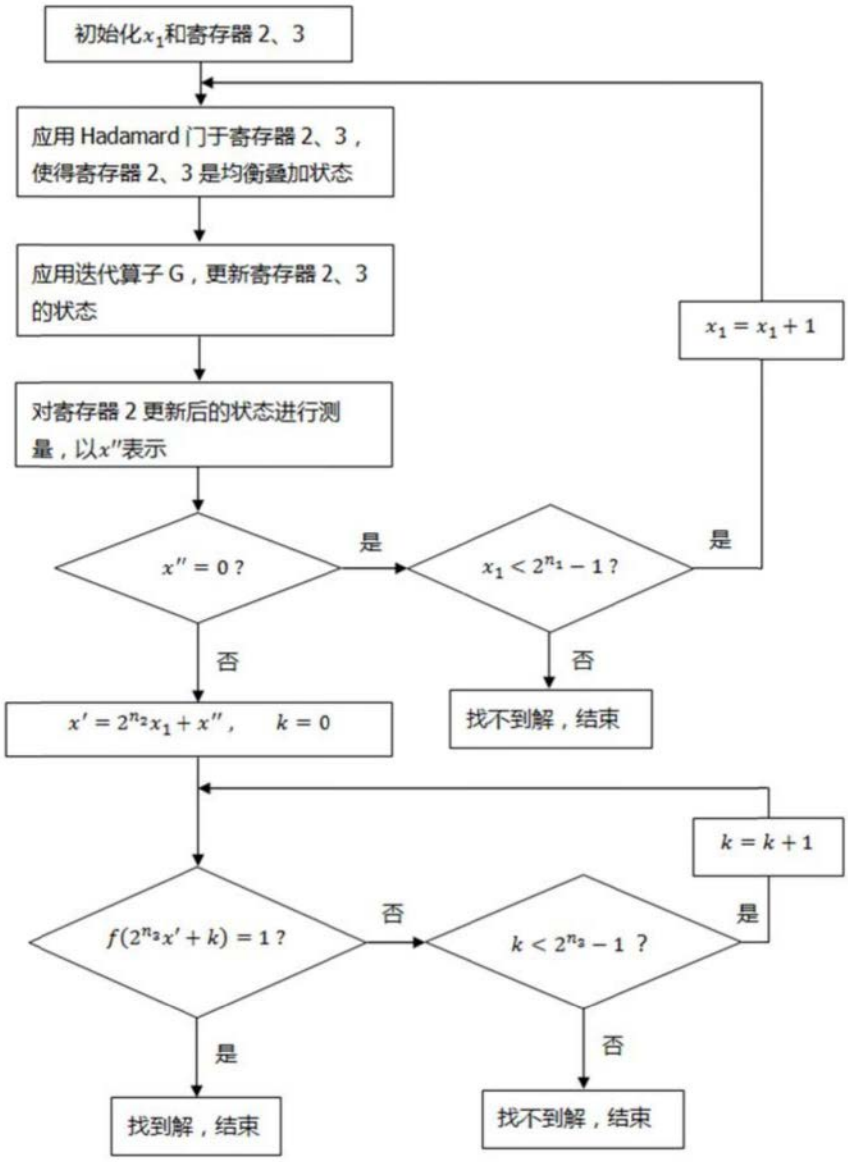

初始化x_1和寄存器2、3
应用Hadamard门于寄存器2、3，使得寄存器2、3是均衡叠加状态
应用迭代算子G，更新寄存器2、3的状态
对寄存器2更新后的状态进行测量，以x''表示
$x'' = 0$？
是
$x_1 < 2^{n_1} - 1$？
是
$x_1 = x_1 + 1$
否
否
找不到解，结束
$x' = 2^{n_2}x_1 + x''$， $k = 0$
$k = k + 1$
$f(2^{n_2}x' + k) = 1$？
否
$k < 2^{n_3} - 1$ ？
是
是
否
找到解，结束
找不到解，结束

图1

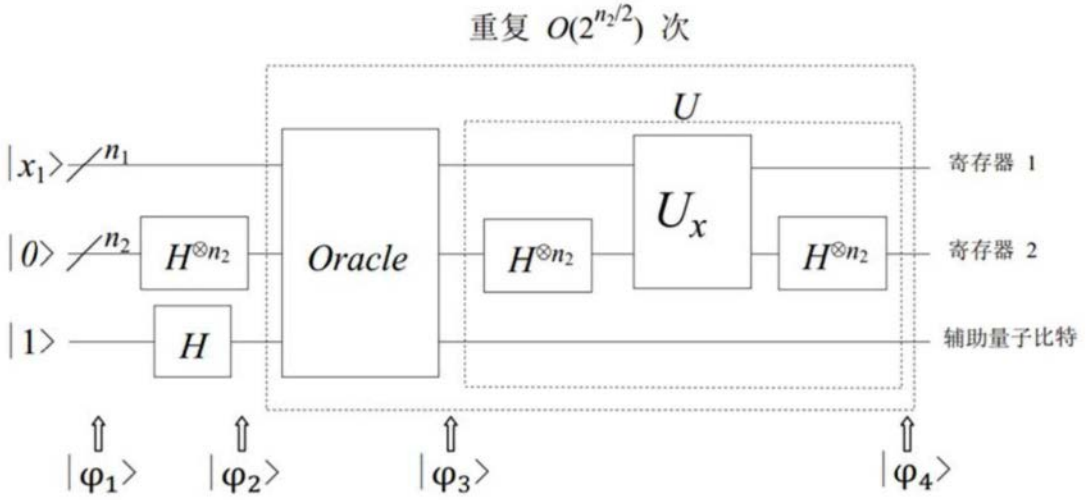

重复 $O(2^{n_2/2})$ 次
U
$|x_1\rangle$
n_1
$|0\rangle$
n_2
$|1\rangle$
$H^{\otimes n_2}$
H
Oracle
$H^{\otimes n_2}$
U_x
$H^{\otimes n_2}$
寄存器 1
寄存器 2
辅助量子比特
$|\varphi_1\rangle$
$|\varphi_2\rangle$
$|\varphi_3\rangle$
$|\varphi_4\rangle$

图2

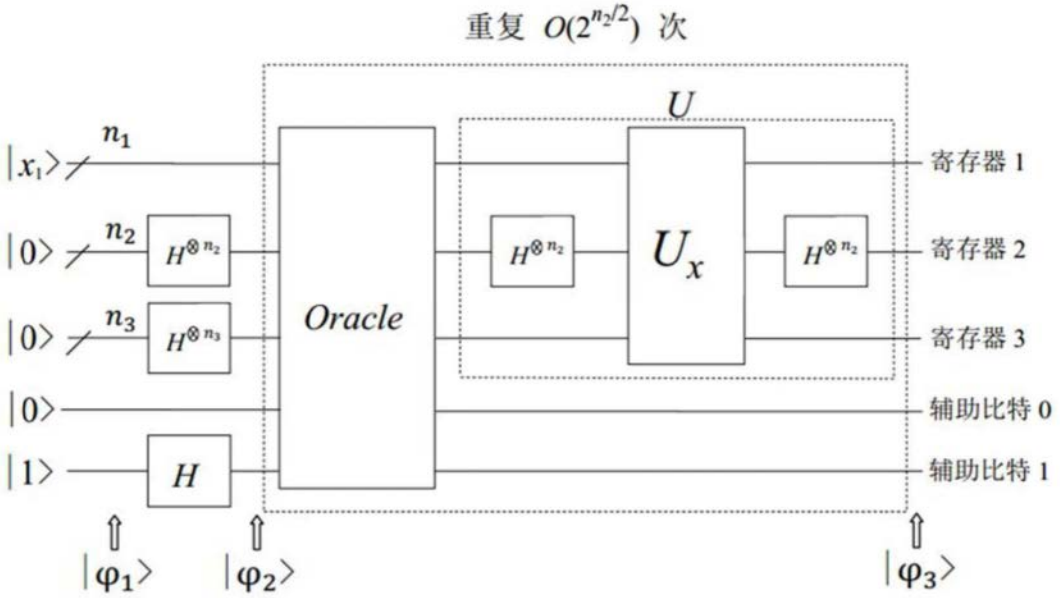

重复 $O(2^{n_2/2})$ 次
U
$|x_1\rangle$
n_1
$|0\rangle$
n_2
$|0\rangle$
n_3
$|0\rangle$
$|1\rangle$
$H^{\otimes n_2}$
$H^{\otimes n_3}$
H
Oracle
$H^{\otimes n_2}$
U_x
$H^{\otimes n_2}$
寄存器 1
寄存器 2
寄存器 3
辅助比特 0
辅助比特 1
$|\varphi_1\rangle$
$|\varphi_2\rangle$
$|\varphi_3\rangle$

图3

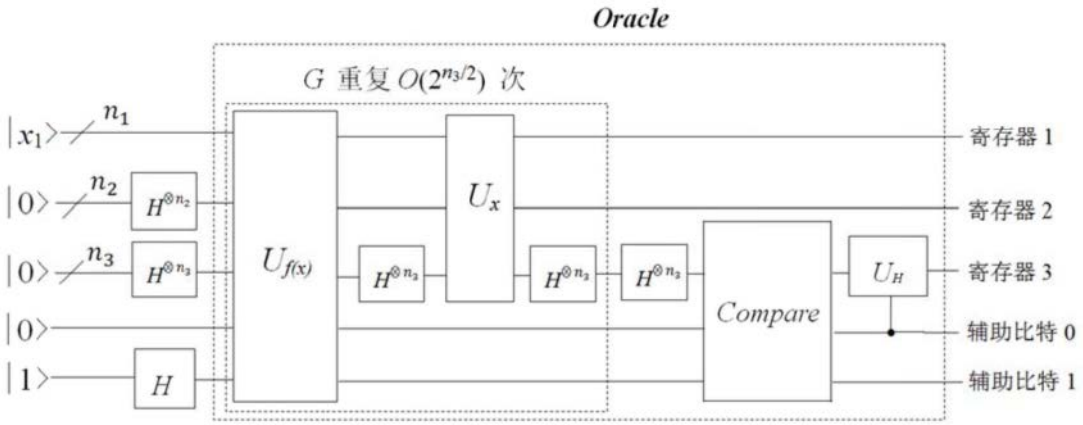

Oracle
G 重复 $O(2^{n_3/2})$ 次
$|x_1\rangle$
n_1
$|0\rangle$
n_2
$H^{\otimes n_2}$
$|0\rangle$
n_3
$H^{\otimes n_3}$
$|0\rangle$
$|1\rangle$
H
$U_{f(x)}$
$H^{\otimes n_3}$
U_x
$H^{\otimes n_3}$
$H^{\otimes n_3}$
Compare
U_H
寄存器 1
寄存器 2
寄存器 3
辅助比特 0
辅助比特 1

图4

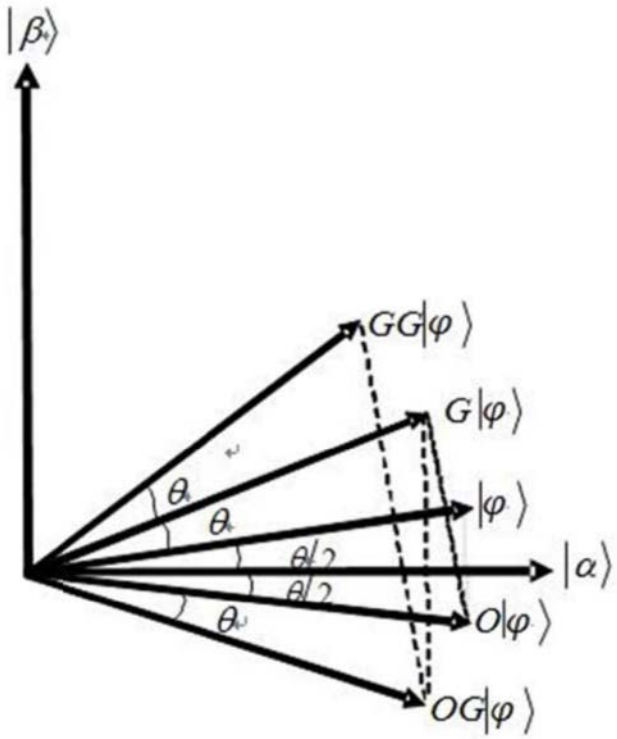

$|\beta\rangle$
$GG|\varphi\rangle$
$G|\varphi\rangle$
$|\varphi\rangle$
$|\alpha\rangle$
$O|\varphi\rangle$
$OG|\varphi\rangle$
θ
θ
$\theta/2$
$\theta/2$
θ

图5