DOI: 10.3969/j.issn.1007-5461.2024.01.015

基于最小权和模板匹配的Oracle线路优化

杨冬晗， 李志强*， 吴 希， 潘文杰， 杨 辉

(扬州大学信息工程学院， 江苏 扬州 225100)

摘 要: 优化量子线路对于提高量子算法的计算效率和降低资源成本至关重要, 特别是在布尔函数构建的Oracle线路中。该优化过程分为两个关键阶段，第一个阶段基于最小权匹配算法对Oracle线路相同受控点的MCT门进行重排序, 最小化生成线路的门数; 第二个阶段利用模板匹配的方式进一步降低线路的门数和代价。实验结果表明, 相较于优化工具RCViewer+, 在4~10位量子比特数下, Deutsch-Jozsa算法下的Oracle线路门数降低了约48.3%, 代价减少了约64.5%; Grover算法下的Oracle线路门数降低了约25.0%, 代价减少了约18.2%。

关 键 词: 量子信息; 量子线路; Oracle线路优化; 最小权匹配; 模板匹配

中 图 分 类 号 : O431.2 **文 献 标 识 码 :** A **文章编号:** 1007-5461(2024)01-00151-10

Optimization of Oracle circuits based on minimum weight and template matching

YANG Donghan, LI Zhiqiang, WU Xi, PAN Wenjie, YANG Hui*

(College of Information Engineering, Yangzhou University, Yangzhou 225100, China)

Abstract: Optimizing quantum lines is essential to improve the computational efficiency and reduce the resource cost of quantum algorithms, especially in the case of Oracle circuit constructed from Boolean functions. This optimization process is divided into two key stages. In the first stage, the MCT gates of the same controlled points of Oracle circuits are reordered based on the minimum weight matching algorithm to minimize the number of gates for generating circuits. In the second stage, the method of template matching is utilized to further reduce the number of gates and the cost of the circuits. The experimental results show that, compared with the optimization tool of RCViewer+, for 4-10 qubits, the number of Oracle circuit gates can be reduced by about 48.3%, and the cost can be reduced by about 64.5% using Deutsch-Jozsa algorithm, while the number of Oracle circuit gates is reduced by about 25.0%, and the cost is reduced by about 18.2% using Grover algorithm.

Key words: quantum information; quantum circuit; Oracle optimization; minimum weight matching; template matching

基金项目: 国家自然科学基金 (62071240), 江苏省高校基金 (10KJB520021)

作者简介: 杨冬晗 (1995 -), 江苏南通人, 研究生, 主要从事量子线路综合方面的研究。E-mail: 1198391037@qq.com.

导师简介: 李志强 (1974 -), 江苏扬州人, 博士, 教授, 硕士生导师, 主要从事量子计算、量子可逆线路综合方面的研究。E-mail: yzqqLzq@163.com

收稿日期: 2022-03-04; **修改日期:** 2022-04-14

*通信作者。

0 引 言

量子算法往往需要计算一些经典的逻辑函数, 实现这种逻辑函数的量子线路被称为Oracle线路。研究人员对DJ和Grover算法的实现和模拟进行了研究: Li等[1]在IBM Q上实现了DJ的综合模拟; Figgatt等[2]在量子计算机上实现了3量子比特的Grover搜索模拟; Gheorghe-Pop等[3]讨论了DJ和Grover算法生成和优化Oracle线路的必要性, 但并未提出具体的优化方案; 谷歌的Cirq框架也未对这两个算法的Oracle线路进行优化[4]。

本文针对DJ和Grover算法, 通过模拟布尔函数$f(x)$, 根据MCT门的特性实现了Oracle线路的优化。

1 预备知识

1.1 Deutsch-Jozsa和Grover算法

Deutsch问题[5]给定一个承诺, 函数$f(x)$是常数函数或者平衡函数。通过构造DJ线路, 可以测量判断这个函数是常数函数还是平衡函数。对于一个n量子比特的二进制函数, 设$f:\{0,1\}^n\to\{0,1\}$, 如图1所示, 将前n个量子比特调制成量子状态$|0\rangle^{\otimes n}$, 最后一个目标量子比特处于$|1\rangle$状态。通过测量发现, 当$f(x)$为常数函数时测得$|0\rangle^{\otimes n}$的概率为100%, 否则为平衡函数。

Grover算法[6]解决的问题是在$N=2^n$个无结构的元素中搜索出目标解。每个元素都对应一个函数$f(x)$, 其中$x\in\{0,1\}^n$, 当满足$f(x)=1$时, x是搜索问题的解, 反之则不是解。将搜索元素写成二进制量子态形式, 表示为$|x\rangle=|x_{n-1}x_{n-2}\cdots x_0\rangle$。图2给出了一个完整的Grover算法量子线路框架, 包含均匀叠加态、Oracle线路u_f、平均反演算子和最终的测量模块。Oracle和平均反演算子组成一个完整的G迭代, 重复G迭代能够提高目标解的输出概率。假设有m个搜索元素, 那么测量输出概率最高的前m个量子态, 即为目标元素。

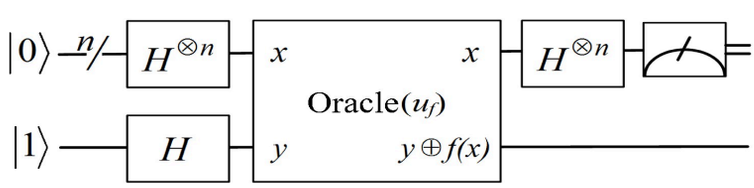

图1 Deutsch线路

Fig. 1 Deutsch circuit

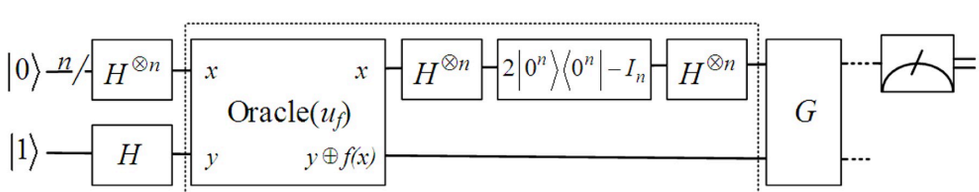

图2 Grover线路

Fig. 2 Grover circuit

1.2 Oracle线路

Oracle线路中函数$f(x)$的计算是可逆的, 其中添加了一个辅助量子比特$|y\rangle$。在图1和图2的Oracle线

路 u_f 中, 前面 n 个量子比特的输出与输入是一样的, 最后一个比特的输出值与 $f(x)$ 的值有关。当 $f(x)=0$, $y\oplus f(x)=y$, 即经过Oracle线路后的量子线路输出不变; 当 $f(x)=1$, $y\oplus f(x)=\bar{y}$, 最后一个量子比特翻转。

当 $n=3$ 时, 考虑Deutsch线路中的 $f(x)$ 为平衡函数, 以 $f(x)=\{1,0,1,0,0,0,1,1\}$ 为例, 记为 $f(0,2,6,7)=1$, $f(x)$ 综合出的Oracle线路如图3所示。可以发现, Oracle综合出的线路是 2^{n-1} 个级联的MCT门, 且随着量子比特 n 的增加, 线路的门数和代价呈倍数增加。

当 $n=3$ 时, 假设在Grover线路中存在2个目标元素{4, 6}, 记 $f(4,6)=1$, $f(x)$ 综合出的Oracle线路如图4所示, 当 $|x\rangle=|100\rangle$ 或 $|x\rangle=|110\rangle$ 时, 辅助量子比特 $|y\rangle$ 翻转, 表示找到目标元素。随着量子比特 n 和目标元素的增加, Oracle线路存在着优化的空间。因此, 本研究提出了一种Oracle线路的优化方法。

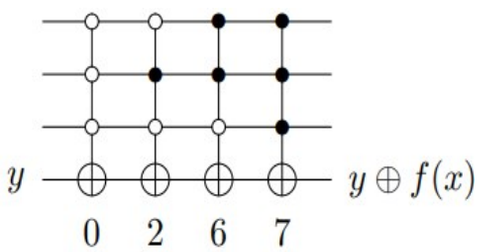

图3　Oracle线路 $f(0,2,6,7)=1$

Fig. 3　Oracle circuit $f(0,2,6,7)=1$

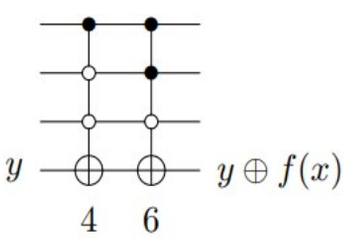

图4　Oracle线路 $f(4,6)=1$

Fig. 4　Oracle circuit $f(4,6)=1$

2　Oracle线路的优化

Oracle线路是根据 $f(x)$ 的真值表依次级联MCT门综合而成的。线路中MCT门具有相同的受控位, 可以通过修改门的级联顺序进行线路替换优化。优化方法分两个阶段完成: (1) 基于门互补控制点的最小权匹配实现MCT门的重新排序、组合替换; (2) 利用模板匹配的方式替换线路中的子结构, 以消除存在冗余。

2.1　阶段 (1) 基于最小权匹配的线路优化

此阶段根据Oracle线路中MCT门的互补控制点数目进行配对, 以达到最大化优化线路代价的目的, 每次替换操作后线路的代价减少。MCT门用 $G(C;t)$ 表示, 其中 C 是控制线, t 是受控线。当MCT门具有相同控制线和受控线时, 可以利用以下三种转换规则。

删除规则R1[7]: 如果存在两个连续的MCT门 $G_1(C_1;t)$ 和 $G_2(C_2;t)$, 其中 $C_1=C_2$, 则 G_1 和 G_2 可以相互抵消, 表示为 $G_1(C_1;t)\circ G_2(C_2;t)=\varnothing$。

合并规则R2[7]: 如果存在两个连续的MCT门 $G_1(C\cup c_i;t)$ 和 $G_2(C\cup \overline{c_i};t)$, 它们在控制线 c_i 处互补, 其他控制线处正负相同, 则 G_1 和 G_2 可以合并成一个门 G_3, G_3 只保留相同的控制线, 表示为

$G_1(C\cup c_i;t)\circ G_2(C\cup \overline{c_i};t)=G_3(C;t)$。

替换规则R3[8]: 如果存在两个连续的MCT门$G_1(C\cup c_i\cup c_j;t)$和$G_2(C\cup \overline{c_i}\cup \overline{c_j};t)$, 它们在控制线$c_i$和$c_j$处互补, 则$G_1$和$G_2$可以被替换成两个代价更低的门$G_3$和$G_4$, 表示为$G_1(C\cup c_i\cup c_j;t)\circ G_2(C\cup \overline{c_i}\cup \overline{c_j};t)=G_3(C\cup c_j;t)\circ G_4(C\cup \overline{c_j};t)$。图5推导了规则R3的转换过程。

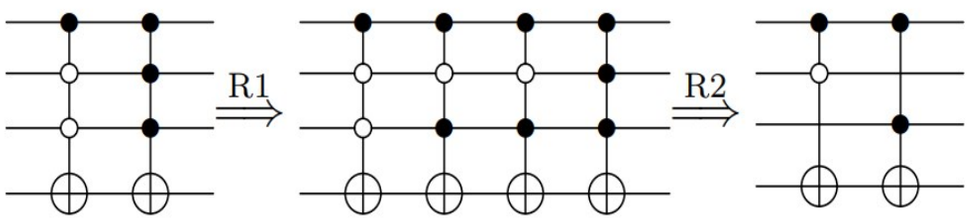

图5 规则R3推导过程

Fig. 5 Derivation process of R3

图6给出了上述每个规则的示例, 其中QC表示每个线路的代价。经过上述规则后, 线路的代价分别降低了26、21和16, 可以发现, 规则R1优化线路的能力最强, R2其次, R3优化能力最弱。

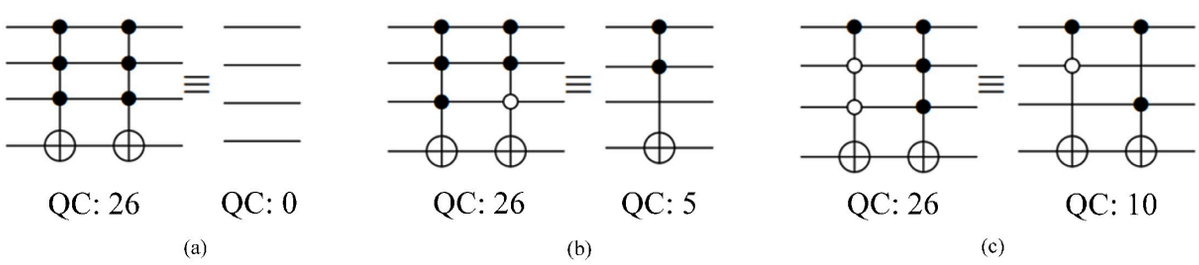

图6 规则R1~R3示例。 (a) R1; (b) R2; (c) R3

Fig. 6 Examples of R1-R3. (a) R1; (b) R2; (c) R3

根据上述规则的推导, 两个级联的MCT门可推广得到线路转换算法Merge: 对于一个n个量子比特的可逆线路, MCT门G_1和G_2具有相同的目标线t, 假设两门有k处互补的控制线, 其余$n-k-1$处控制线都相同。那么门G_1和G_2可以被等效替换成代价更低的k个MCT门。

算法1 线路转换算法Merge

输入: *key* , *value1*, *value2*

输出: 等效线路 *gates*

```
1. gates←ϕ                                  // 定义输出线路 gates
2. count←value1⊕value2                      //确定控制点不同的位置(位运算)
3. while (count){
4.     lastIndex←count & (-count)           //获取最低位的1(位运算)
5.     key←Set the log(lastIndex)-th 1 of the key to 0   // 得到新的控制线位置
6.     gates[key].add(((value1 >> 1) & ~(lastIndex - 1)) + ((lastIndex - 1) & value1))
7.     count←count & (count-1) }            //最低位的1清0, 迭代次数减1
8. return gates
```

算法 1 将具有相同受控、控制线位置的MCT门 G_1 和 G_2 等效替换为新的线路"*gates*"。算法实现时, 用哈希表"*key* : *value*"进行存储, "*key*"表示门控制线的位置, "*value*"表示对应控制点的正负信息。"*key*"和"*value*"的值用二进制表示。"1" 在 "*key*" 中表示有控制线, 在"*value*" 中表示正控制点; "0"在"*key*" 中表示无控制线, 在"*value*" 中表示负控制点。如图7(c) 中的 (3) 和 (4) 两个门可以表示为 "0b1101 : {0b101, 0b110}"。两个门具有相同的控制线 "*key*" 且"*value*"中有 2 个互补控制点, 所以可以替换生成 2 个少 1 根控制线的MCT门, 对应于图7(d) 中的 (6) 和 (7), 表示为 "0b1100 : {0b10}, 0b1001: {0b10}"。

算法 1 的第 3~6 步迭代生成 k 个MCT门, k 是 *count* (mod2) 中 1 的个数。其中, 第 4 步利用位运算确定替换的最低位控制线, 第 5、6 步控制线数目减 1, 将控制位置 *key* 的第 *log* (*lastIndex*) 个 1 置为 0, 同时删除 *value1* 中控制点的正负信息, 那么 *value1* 前面的高位需要右边移一位填补空缺。如将 *key* = 0b1101 的第 *log* (*0b001*) 位 1 置 0, *value* = 0b101 的第 *log*(0b001) 位剔除, 得到新门的键值对表示: "0b1100:{0b10}"。

对于 n 比特的函数 $f(x)$, 假设构建的Oracle线路需要 N 个MCT门。由于控制线和受控线相同, 通过修改MCT门的级联顺序进行两两配对, 一共有 $C_N^2C_{N-2}^2C_{N-4}^2...C_2^2$ 种组合情况。不同的组合情况利用转换算法替换生成的线路代价不同, 而线路的代价与替换得到的门数相关。目标生成 $min\{k_1+k_2+...+k_{N/2}\}$ 个门数目的替换线路, 其中 k 是门 G_1 和 G_2 的互补控制线的数目, 共有 $N/2$ 组配对。

每个相同控制位置的MCT门集合"*gates*"能构成一个有权无向图 G, 图中的点是MCT门, 两点构成边上的权重是互补控制点的数量。目标是所有MCT门配对后, 互补控制点总数最少。本研究利用开花算法[9]实现了一般图的最小权匹配, 开花算法常用于解决一般图 $G=(V, E)$ 的最大匹配问题, 其通过迭代的方式在图 G 中寻找增广路径, 时间复杂度为 $O\left(|E|\sqrt{|V|}\right)$。搜索过程中, 如果遇到奇数环, 则将环缩成一个点, 然后继续在收缩的图中迭代。每增广一条路径, 匹配数目增加一个, 直至找不到新的路径, 得到最大匹配。假设MCT门有 n 个控制线, 每次循环在互补控制点为 i 的子图中寻找最大匹配, 记最大匹配数为 c_i, 最终能替换得到 $\sum_{i=1}^{n}ic_i$ 个MCT门。算法 2 总结了匹配算法Match。

算法 2 门集合的最小权匹配算法 Match

输入: 相同控制线位置的门集合 *gates*, 门集合构成图 G

输出: 配对后的门集合 *retGates*

```
1. retGates←Ø   // 初始化输出门集合
2. while gates = Ø{
3.   for each G(i) ∈G do{   // 提取边权为 i 的子图, 即MCT门互补控制点数目为 i
4.   gates'←Blossom(G(i))   // 利用开花算法找到互补控制点为 i 的最大匹配
5.   retGates.add(gates')    // 添加到结果
6.   gates←gates － gates'}} // 去除已匹配的点
7. return retGates
```

算法 2 输入具有相同控制线的门集合"*gates*"和无向图 G, $G(i)$ 是所有边权为 i 的子图, 输出是最大匹配的门集合。算法第 3~6 步首先提取出边权为 1 的子图 G', 采用开花算法实现规则 R2 下的最大匹配输出"*gates*´", 即找到MCT门互补控制点为 1 的最多配对, 并将配对后的门存储进输出线路; 然后从MCT门集合"*gates*" 中剔除已匹配的门, 并继续从未配对的点中提取边权为 2 的子图, 进行规则 R3 的最大匹配, 以此类

推, 直到所有门匹配。算法2的时间复杂度为$O\left(|E|\sqrt{|V|}\right)$, E、V分别表示无向图的边和点。在DJ算法的Oracle线路中, 最坏的情况下图G是一个全连接图, $|E|$有$N(N-1)/2$个边, $|V|$对应$N=2^{n-1}$个MCT门, 因此算法2的最差时间复杂度为$O\left(\sqrt[5]{N}\right)$。

根据算法1和算法2对Oracle线路进行递归优化, 分为两个步骤: (1) 先利用算法2对具有相同控制线的MCT门进行配对, 实现MCT门的最小权匹配, 再利用算法1对配对后的门两两替换, 得到新的等效线路; (2) 按照MCT门控制线的位置对新的等效线路进行分类, 然后再对分类后的MCT门集合继续执行第一步操作, 直至不能转换生成新的线路。算法3总结了阶段1的线路优化算法Optimize。

算法3 基于最小权匹配的线路替换算法Optimize

输入: MCT门集合 *gates*

输出: 等效变换后的门集合 *gates*1

```
1. gates1←new hash()     // 初始化输出的门集合
2. for key in gates{         // key表示MCT门的控制线位置信息
3.     n←gates[key].length   // 具有相同控制线的MCT门集合, 有n个元素
4.     Match(gates[key])     // 算法2重排序, 实现最小权匹配
5.     for i←0 to n/2 do{     // 共有n/2个门对
6.         gates1.add(Merge(key, gates[key][i*2], gates[key][i*2+1]))}} // 算法1两两替换
7.     if n % 2 = 1{           // n为奇数, 最后一个门直接存储
8.         gates1.add(gates[key][n-1]) }
9. if gates1 = gates{      // 不再替换生成新的线路
10.        return gates1 }   // 返回结果线路
11. Optimize (gates1)     // 在新的线路中递归算法3
```

算法3 第1步初始化哈希表, 用于存储替换后的MCT门, 哈希表的键"*key*"表示MCT门的控制点位置, 值"*value*"表示对应控制点的正负信息。第2~8步利用算法2和算法1对MCT门进行配对替换, 第9、10步判断线路能否继续优化, 如果不再生成新门, 则返回最终的门集合"*gates*1", 否则继续递归Optimize算法。

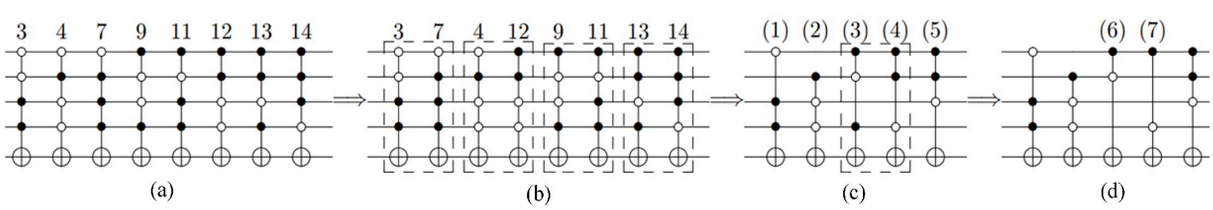

图7 (a) $f(3,4,7,9,11,12,13,14)=1$初始线路; (b) 算法2门匹配; (c) 算法1线路转换; (d) 最终线路

Fig. 7 (a) Initial circuit of $f(3,4,7,9,11,12,13,14)=1$; (b) Gate matching of algorithm 2 (c) Circuit conversion of algorithm 1; (d) Final circuit

图7举例展示了算法3的整体流程。例如, 当$n=4$时, Deutsch随机生成的平衡函数为$f(3,4,7,9,11,12,13,14)=1$。图7(a)是Oracle初始线路, 经过算法2重新排序得到图7(b)线路, 图中四个虚线框表示配对后的MCT门。此时, 四组门的互补控制线总数为5且最少, 所以最终生成5个少一个控制点的MCT门, 对应图7(c)中的门(1)~(5)。继续执行算法3, 门(3)和(4)转换成门(6)和(7), 得到图7(d)所示最终线路, 递归结束。

2.2　阶段2基于模板匹配的线路优化

经过阶段1优化后, Oracle线路中的MCT门不再具有相同的控制线位置, 即 "*key*" 中只剩一个门, 此时通过模板匹配的方式对线路进一步优化。记 G_1 和 G_2 相同的控制线为 C, 互补的控制线为 c_p, 只在 G_1 中的控制线为 c_m, 只在 G_2 中的控制线为 c_n。

模板T1: 如果有两个级联的MCT门 $G_1\left(C\cup\{c_m\};t\right)$ 和 $G_2\left(C;t\right)$, 其中控制线 c_m 只有一根, 那么可以合并成一个门 $G_3(C\cup\{\overline{c_m}\};t)$, 表示为: $G_1\left(C\cup\{c_m\};t\right)\circ G_2\left(C;t\right)=G_3(C\cup\{\overline{c_m}\};t)$。

模板T2: 如果有两个级联的MCT门 $G_1\left(C\cup\{c_p\}\cup\{c_m\};t\right)$ 和 $G_2\left(C\cup\{\overline{c_p}\};t\right)$, 其中控制线 c_p 和 c_m 只有一根, 那么可以替换成代价更低的两个门 $G_3(C\cup\{c_p\}\cup\{\overline{c_m}\};t)$ 和 $G_4(C;t)$, 表示为: $G_1\left(C\cup\{c_p\}\cup\{c_m\};t\right)\circ G_2\left(C\cup\{\overline{c_p}\};t\right)=G_3(C\cup\{c_p\}\cup\{\overline{c_m}\};t)\circ G_4(C;t)$。

模板T1和T2的推导过程如图8、9所示, 其中虚线框表示需要处理的门, 大括号表示变换得到的门。图10展示了阶段2模板匹配的过程, 其中虚线框表示可以模板匹配的门对, 门 (1)~(5) 表示匹配生成的门。

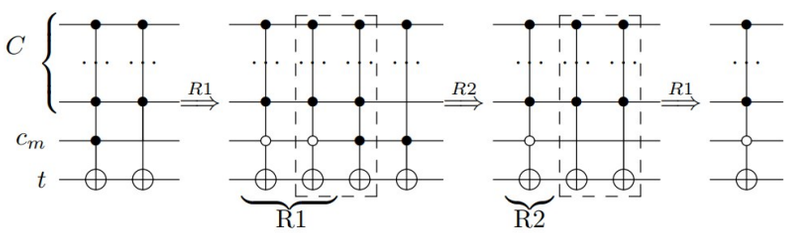

图8　模板1推导过程

Fig. 8　Derivation process of template 1

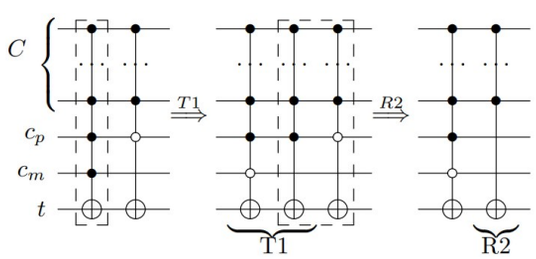

图9　模板2推导过程

Fig. 9　Derivation process of template 2

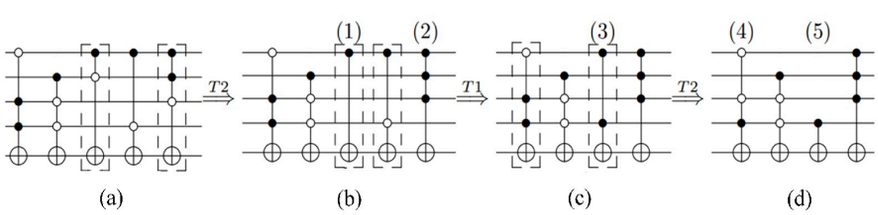

图10　模板匹配流程。 (a) T2匹配; (b)T1匹配; (c) T2匹配; (d) 最终线路

Fig. 10　Matching process of template. (a) T2 matching; (b) T1 matching; (c) T2 matching; (d) Final circuit

算法4总结了Oracle的整体优化流程, 输入 $f(x)$ 生成的MCT门集合 "*gates*", 第2步使用阶段1进行线路优化, 第3、4步根据模板T1和T2进行匹配。算法递归地执行阶段1和2的操作, 直至线路不再改变, 输出最终优化的线路。

3 实验结果及优化情况分析

RCViewer+[10]是一款支持tfc文件格式的软件, 用于逻辑线路综合优化、等效验证、代价计算。为了验证阶段1和阶段2生成线路的正确性, 本研究将优化后的Oracle线路存储为tfc文件, 并通过RCViewer+进行校验。实验使用的计算机CPU型号为i5-10210U, 内存为8 G, 编程语言为Python。

算法4 Oracle整体优化算法

输入:MCT门集合 *gates*
输出:优化后的线路 *gates*
1. do{
2. *g*1←Optimize(*gates*) //阶段1优化
3. *g*2←T1_Match (*g1*) //阶段2模板T1优化
4. *gates*←T2_Match (*g2*) //阶段2模板T2优化
5. }while(*gates* != *g*1)
6. return *gates*

本研究随机生成10组4量子比特的平衡函数$f(x)$作为DJ算法的输入。表1对比了RCViewer+和本研究提出算法在原始Oracle线路上的优化结果, 评价指标分别为GC和QC, GC表示线路门数, QC表示线路代价。与RCViewer+自带的优化功能对比, 经过两阶段的优化后, 10组函数的平均线路门数和代价分别降低了46.4%和73.5%, 验证了所提出算法的有效性。

表1 n = 4 Deutsch优化实验结果

Table 1 Deutsch optimized experimental results with n = 4

	$f(x)$	Original		RCViewer+		Phase1		Phase2		Impr/%	
		GC	QC	GC	QC	GC	QC	GC	QC	GC	QC
1	{0, 1, 3, 4, 5, 9, 12, 15}	8	232	8	108	5	42	5	33	37.5	69.4
2	{0, 1, 7, 10, 11, 12, 14, 15}	8	232	8	106	6	45	5	38	37.5	64.2
3	{0, 2, 3, 6, 9, 10, 11, 14}	8	232	7	93	5	36	3	25	57.1	73.1
4	{1, 2, 5, 7, 8, 11, 12, 15}	8	232	7	91	4	26	4	16	42.9	82.4
5	{1, 3, 8, 11, 12, 13, 14, 15}	8	232	6	78	5	65	3	39	50.0	50.0
6	{2, 3, 4, 5, 10, 11, 12, 14}	8	232	4	52	4	38	4	28	0.0	46.2
7	{2, 5, 6, 7, 8, 10, 12, 13}	8	232	8	232	5	43	4	34	50.0	85.3
8	{3, 4, 7, 9, 11, 12, 13, 14}	8	232	11	143	5	49	4	40	63.6	72.0
9	{4, 5, 6, 7, 9, 11, 12, 14}	8	232	4	52	3	11	2	6	50.0	88.5
10	{4, 7, 8, 9, 10, 11, 12, 15}	8	232	6	78	4	19	3	15	50.0	80.8
	Avg			6.9	103.3			3.7	27.4	46.4	73.5

为了测试Oracle线路优化算法在n比特情况下的实验效果, 表2给出了DJ算法下100组随机平衡函数的平均GC、QC和执行时间ET。由实验结果可知, 随着量子位数的增加, 线路的门数、代价和执行时间不断增加。相较于RCViewer+, 所提出算法的平均GC、QC值分别降低48.3%和64.5%。当比特数n超过10时,

生成MCT的门数超过2^{10}个, 其中阶段1寻找最小权匹配的时间复杂度为$O\left(\sqrt[5]{N}\right)$, 花费的时间呈指数级增长, 因此所提出方法仅适用于小规模的Oracle线路优化, 对比实验只展示$n\leq 10$的部分。

针对Grover算法, 表3给出了2^n搜索空间中100组随机目标元素集合的平均优化结果和执行时间。相较于RCViewer$^+$, 本研究优化后的平均GC、QC分别降低了25.0%、18.2%。对比表2和表3可知, Grover算法下的Oracle优化效果不及DJ算法, 主要原因在于Grover的初始Oracle线路包含较少的MCT门, 配对方式相应减少, 线路的可优化程度降低。

表2　*n* bit DJ的Oracle线路平均优化结果

Table 2　Optimized experimental results of DJ's Oracle circuit with *n* bit

n	Original		RCViewer$^+$		Phase1		Phase2		Impr/%		ET/s
	GC	QC	GC	QC	GC	QC	GC	QC	GC	QC	
4	8	232	7	86	4	39	4	32	42.9	62.8	<0.1
5	16	976	13	396	10	151	7	115	46.2	71.0	0.1
6	32	4000	29	1555	22	471	15	342	48.3	78.0	0.3
7	64	16192	60	4840	48	1428	30	993	50.0	79.5	1.4
8	128	65152	122	12204	106	3619	59	2379	51.6	80.5	9.2
9	256	261376	245	31413	226	9578	123	6328	49.8	79.9	87.2
10	512	1047040	496	147448	475	109340	265	60153	46.6	59.2	741.3
Avg			138.9	28277.4			71.9	10048.9	48.3	64.5	

表3　*n* bit Grover的Oracle线路平均优化结果

Table 3　Optimized experimental results of Grover's Oracle circuit with *n* bit

n	Original		RCViewer$^+$		Phase1		Phase2		Impr/%		ET/s
	GC	QC	GC	QC	GC	QC	GC	QC	GC	QC	
4	3	88	2	36	2	36	2	34	0.0	5.6	<0.1
5	5	301	4	114	4	112	4	108	0.0	5.3	<0.1
6	8	1122	7	343	8	330	7	308	0.0	10.2	0.1
7	17	4351	17	939	17	921	15	831	11.8	11.5	0.4
8	31	15809	36	2306	35	2233	29	1984	19.4	14.0	1.2
9	58	59994	87	6307	83	6126	67	5200	23.0	17.6	8.3
10	134	266443	235	15937	219	15507	167	12798	28.9	19.7	123.9
Avg			55.4	3711.7			41.6	3037.6	25.0	18.2	

4　结　论

针对DJ和Grover算法下的Oracle线路, 提出了一种基于最小权和模板匹配的优化算法, 旨在降低Oracle的门数与代价。相较于优化工具RCViewer$^+$, 所提出算法下的Oracle线路门数和代价进一步降低。但在真实的物理量子计算机中, 量子门需要满足线性最近邻约束, 本研究并没有考虑这一约束, 将在后续研究中进行探讨。

参考文献:

[1] Li Z Q, Dai J, Pan S H, *et al*. Synthesis of Deutsch-Jozsa circuits and verification by IBM Q [J]. *International Journal of Theoretical Physics*, 2020, 59(6): 1668-1678.

[2] Figgatt C, Maslov D, Landsman K A, *et al*. Complete 3-Qubit Grover search on a programmable quantum computer [J]. *Nature Communications*, 2017, 8: 1918.

[3] Gheorghe-Pop I D, Tcholtchev N, Ritter T, *et al*. *Computer Scientist's and Programmer's View on Quantum Algorithms: Mapping Functions' APIs and Inputs to Oracles* [M]. Lecture Notes in Networks and Systems, Cham: Springer International Publishing, 2021: 188-203.

[4] Google. A python library for NISQ circuits [OL]. https://quantumai.google/cirq.

[5] Qiu D W, Zheng S G. Revisiting Deutsch-Jozsa algorithm [J]. *Information and Computation*, 2020, 275(1): 104605.

[6] Gilliam A, Pistoia M, Gonciulea C. Optimizing quantum search using a generalized version of Grover's algorithm [J]. *Quantum Physics*, 2020, 1(1): 1-7.

[7] Datta K, Sengupta I, Rahaman H. A post-synthesis optimization technique for reversible circuits exploiting negative control lines [J]. *IEEE Transactions on Computers*, 2015, 64(4): 1208-1214.

[8] Bandyopadhyay C, Parekh S, Rahaman H. Improved circuit synthesis approach for exclusive-sum-of-product-based reversible circuits [J]. *IET Computers & Digital Techniques*, 2018, 12(4): 167-175.

[9] Shoemaker A, Vare S. Edmonds' blossom algorithm [J]. *CME*, 2016, 1(1): 1-27.

[10] Maslov D. Reversible logic synthesis benchmarks page [OL]. https://reversiblebenchmarks.github.io.