计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.250200097

基于量子萤火虫算法的2QAN量子电路调度优化

李　晖[1,2]　王杰鹏[1]　姬迎松[1]　陈禹彤[3]
1 哈尔滨商业大学计算机与信息工程学院　哈尔滨 150028
2 黑龙江省电子商务与信息处理重点实验室　哈尔滨 150028
3 北京理工大学信息与电子学院　北京 100081
(hrbcu_lh@163.com)

摘　要　针对传统量子调度策略缺乏对线路结构特征和层次需求的细致考量,优化执行过程中易产生冲突,导致并行度降低以及电路深度增加等问题,提出了量子萤火虫算法,并将其应用于2QAN量子电路调度优化。在传统萤火虫算法基础上引入量子信息,使得个体能够同时探索多个位置,增加搜索空间覆盖范围,通过波函数演化和坍缩机制,实现了对新解探索与已知解开发之间的平衡;同时引入随机扰动增强搜索多样性,利用量子隧穿效应避免陷入局部最优。通过4个基准测试函数进行测试,测试结果表明,与萤火虫算法相比,量子萤火虫算法的收敛速度提升约40%,解的质量约提升67%,搜索效率提升45%。算法通过评估不同调度方案适应度值,优化量子门操作顺序,减少电路深度和移动操作,进而提高了电路并行度。实验结果表明,在量子电路调度优化中,量子萤火虫算法相较于传统算法、2QAN电路、2HQAA算法以及LCRA与LTSA的结合算法,SWAP门数平均减少42%,6.7%,10.4%和3%,CNOT门数平均减少15.6%,10.8%,11%和2.2%。

关键词:量子电路;量子门调度;量子萤火虫算法;波函数演化;量子隧穿效应
中图分类号　TP391

2QAN Quantum Circuit Scheduling Optimization Based on Quantum Firefly Algorithm

LI Hui[1,2],WANG Jiepeng[1],JI Yingsong[1] and CHEN Yutong[3]
1 School of Computer and Information Engineering,Harbin University of Commerce,Harbin 150028,China
2 Heilongjiang Provincial Key Laboratory of Electronic Commerce and Information Processing,Harbin 150028,China
3 School of Information and Electronics,Beijing Institute of Technology,Beijing 100081,China

Abstract　Aiming at the underconsideration of line structure characteristics and hierarchical demands of the traditional quantum scheduling strategy,and the execution will be occurred to reduce parallelism and increase the circuit depth during optimization execution process,this paper proposes the Quantum Firefly Algorithm(QFA) to apply it to 2QAN quantum circuit scheduling optimization. Quantum information is introduced to explore multiple locations simultaneously,and it increases the coverage of the search space. A balance between the exploration of the new solutions and the development of known solutions through the wave function evolution and collapse mechanism,meanwhile,the random perturbations is imported to enhance the search diversity,and the solutions will be jump out of the local optimum with quantum tunneling effect. The algorithm optimizes the order of quantum gate operations by evaluating the fitness values of different scheduling schemes to reduce the circuit depth and move operations,which in turn improves the circuit parallelism. Tests are conducted on four benchmark functions. The test results show that,compared with the firefly algorithm,the convergence speed of the quantum firefly algorithm is improved by approximately 40%,the quality of the solutions is enhanced by about 67%,and the search efficiency is increased by 45%. In the optimization of quantum circuit scheduling,compared with the traditional algorithm,the 2QAN circuit,the 2HQAA algorithm,and the combined algorithm of LCRA and LTSA,the number of SWAP gates of the quantum firefly algorithm is on average reduced by 42%,6.7%,10.4%,and 3% respectively,and the number of CNOT gates is on average decreased by 15.6%,10.8%,11%,and 2.2% respectively.

Keywords　Quantum circuits,Quantum gate scheduling,Quantum firefly algorithm,Wave function evolution,Quantum tunneling

基金项目:黑龙江省自然科学基金(LH2024F042);黑龙江省普通本科高等学校青年创新人才培养计划(UNPYSCT2020212);哈尔滨商业大学"青年科研创新人才"培育计划(2023KYYWF0983)
This work was supported by the Natural Science Foundation of Heilongjiang Province,China(LH2024F042),University Nursing Program for Young Scholars with Creative Talents in Heilongjiang Province(UNPYSCT2020212) and Cultivation Program for Young Scholars with Creative Talents of Harbin University of Commerce(2023KYYWF0983).
通信作者:王杰鹏(ekkr18510@163.com)

1 引言

量子计算硬件设备的飞速发展极大延长了量子比特的相干时间，但目前可用的量子计算机仍处于中等规模含噪量子计算机阶段[1-2]。对于量子计算机来说，尽可能缩短量子比特操作时间非常重要，因为这样可以提高在任何量子比特解旋之前完成所有操作的概率，从而获得保真度更高的计算结果[3]。

量子编译器将量子电路(即量子操作序列)作为输入程序，生成相应控制序列在目标硬件执行。例如，在使用超导量子比特量子计算机中，一个量子操作会在一定时间内被编译成若干控制指令。在量子编译器中，确定每个量子操作的执行起始时间不发生任何重叠是一项必不可少的任务，称为量子电路调度[4-6]。量子电路调度必须考虑到 3 个主要约束：一个是逻辑依赖性，即算法中的固有操作顺序；另外两个都是硬件限制，包括任何量子比特都不能同时参与一个以上的门，以及双量子比特门只能在物理连接或者相互作用的量子比特之间实现[7]。这些约束导致量子算法无法直接在量子计算设备上执行。解决这一问题涉及映射、调度、错误矫正等方面的工作，其中量子电路调度就是一个至关重要的研究点。调度的主要目的是在满足硬件约束条件的同时，优化电路性能，包括减小延迟、深度、优化量子门数量、提高并行性和资源利用率，以确保在特定量子硬件上可以正确高效地执行量子算法[8-12]。

近年来研究人员致力于改进量子电路调度算法，以适应不断发展的量子计算硬件和应用需求。随着现有量子计算硬件不断推进和扩展，最小化操作数量成为量子电路调度研究热点。Guerreschi 等[13]提出了两步法，首先安排逻辑门，忽略连接考虑，然后在后续步骤中以最小化开销的方式添加路由操作。Lao 等[14]提出一种名为 Qmap 的定时和资源感知映射器，它能够确保量子电路在可扩展的超导处理器 Surface-17 上执行，并实现最短电路延迟。Alam 等[15]提出 4 种通用方法，利用门重新排序优化量子近似优化算法(Quantum Approximate Optimization Algorithm，QAOA)电路。这种重排序使更多量子门并行执行，减少编译 QAOA 电路所需的额外门数，从而降低电路深度，有助于降低电路运行时间，并提升噪声韧性。Alam 等[16]设计一个包含 3 种方法的编译流程，找到具有减少深度和门数的重新排序电路最优解。门模型量子计算机对于实现近期量子计算机架构和量子设备至关重要。Liu 等[17]提出了 QuCloud+新型量子比特映射方案，提高了 2D/3D NISQ(Noisy Intermediate-Scale Quantum)量子计算机在单任务和多任务编程中的保真度和资源利用率。QuCloud+通过解决现有映射方案面临的挑战，如串扰、SWAP 操作开销、不同设备拓扑结构等问题，从而提高了量子计算结果的准确性和计算效率。Silva 等[18]提出了多量子比特晶格手术调度算法用于优化二维拓扑量子纠错码中的量子电路编排。这种方法特别关注多量子比特长距离操作的调度问题，通过使用简单交换规则，可以将量子电路转换为仅包含非 Clifford 多量子比特门序列，显著减少了测试电路集上的电路长度，并且与串行执行相比，使多量子位门电路预期执行时间进一步减少。

针对现有量子电路调度方法在提升执行效率、减少量子门操作数及优化电路并行度等方面的不足，本文提出量子萤火虫算法(Quantum Firefly Algorithm，QFA)。融合波函数演化与经典搜索机制，在严格遵循量子比特连接拓扑及量子门依赖关系等约束条件下，对量子电路结构进行优化。通过此优化，可有效降低电路深度，增强并行性，同时减少 SWAP 操作数量，进而提升量子电路的整体执行效率。

2 量子 2QAN 电路调度优化的意义

2.1 量子电路层

定义 1(量子电路层) 量子电路层指在同一时间步内可以并行执行且量子比特间无冲突的一组量子门集合。

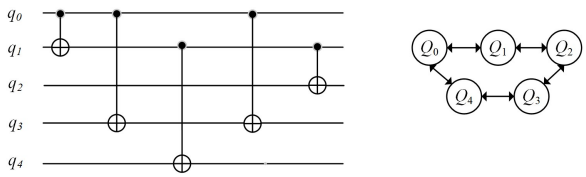
设一个量子电路有 n 个量子比特，第 l 层量子门集合为 $G_l=\{g_{l,1},g_{l,2},\cdots,g_{l,k}\}$，其中每个门 $g_{l,k}$ 作用于一个或多个量子比特，量子电路层有如下性质：

性质 1(并行性) 任意 $g_{l,i},g_{l,j}\in G_l$，同时 $g_{l,i}$ 作用于量子比特集合 $\boldsymbol{P}_i$，$g_{l,j}$ 作用于量子比特集合 $\boldsymbol{P}_j$，且 $P_i\cap P_j=\phi$，则 $g_{l,i},g_{l,j}$ 可以在层 l 框架下并行执行。

性质 2(顺序性) 设任意 $g_{l,i}\in G_l$，$g_{l+1,i}\in G_{l+1}$，$g_{l+1,i}$ 可以执行的必要条件是当且仅当 $g_{l,i}$ 被执行。由顺序性易知，量子电路的总体操作为：$U=U_L\circ U_{L-1}\circ\cdots\circ U_1$，其中 U_l 是第 l 层的整体操作，$\circ$ 表示操作的复合。

设 q 表示逻辑量子位，Q 表示物理量子位，图 1(a)所示逻辑电路的物理架构可以用图 1(b)表示，电路门序列形式为：

$$C=((q_0,q_1),(q_0,q_3),(q_1,q_4),(q_0,q_3),(q_1,q_2))$$



(a)逻辑电路 (b)架构图

图 1 量子电路结构

Fig. 1 Structure of quantum circuit

2.2 量子电路调度优化

量子电路调度优化主要是提高量子电路执行效率与可靠性。量子电路调度优化有多个方面，本文主要考虑以下 3 个方面：

(1)门级优化：减少量子门总数，合并可以组合的量子门操作，消除冗余门操作或者将复杂门分解成基本门集。

(2)并行优化：在考虑硬件拓扑约束情况下最大化量子门并行执行。

(3)深度优化：在硬件约束情况下，尽可能减小量子电路深度，深度减小即执行时间减少。

图 2 展示了图 1(a)中 5 个量子位哈密顿电路编译成图 1(b)中网络架构的实例，其中，节点表示量子位，边表示量子位之间的连接性，虚线划分电路层结构，同一层内门操作可以并行执行。为提高可读性并避免混淆，图中 SWAP 门操作被应用于相应硬件量子位，并将其绘制在电路量子位。

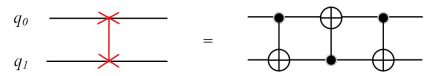
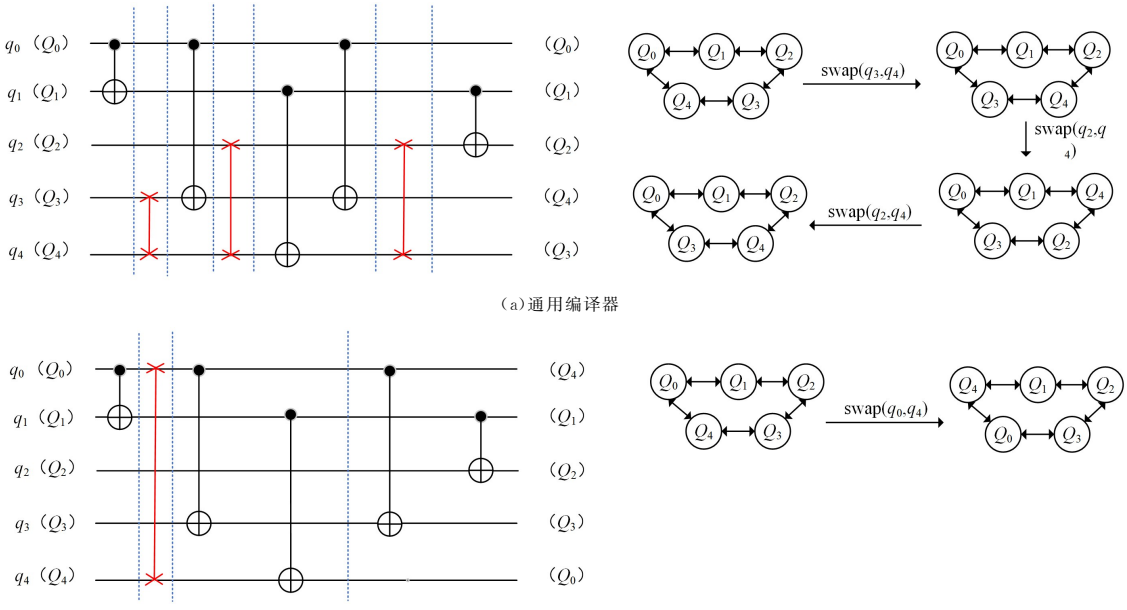


图 2 SWAP 门分解

Fig. 2 SWAP gate decomposition

图 3(a)描述了使用图 1(a)中门依赖性的通用编译器的编译过程，插入 3 个 SWAP 门(如图 2 所示，1 个 SWAP 门相当于 3 个 CNOT 门，因此每插入一个 SWAP 门深度增加 3)，并输出具有 14 个双量子位门且 $D=13$ 的电路。图 3(b)利用哈密顿模拟问题中灵活算子排列；相比之下，考虑运算符排列灵活性的编译器仅使用 1 个 SWAP 门，进行调度优化后电路只有 8 个双量子位门且 $D=7$ 且这个 SWAP 门可以与电路中其它门组合，可以进一步减少门数和电路深度。



(a)通用编译器

(b)调度优化后编译结果

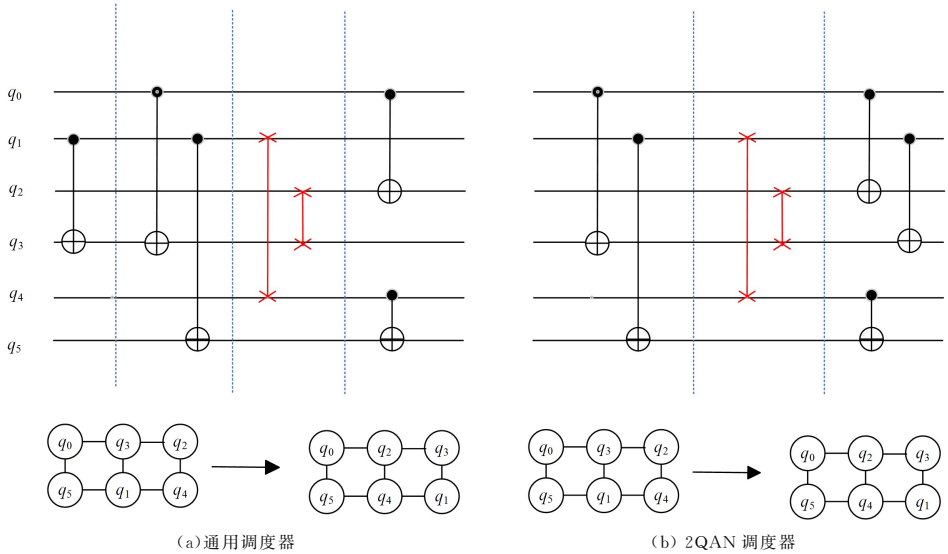
图 3　量子电路调度优化前后对比

Fig. 3　Comparison before and after quantum-circuit scheduling optimization

2.3　基于 2QAN 的量子门调度策略

2QAN[19]是一种优化 2-local 量子位哈密顿模拟问题的量子电路。其可以针对不同量子比特拓扑和硬件门集进行量子电路的优化。2QAN 中门调度方法采用混合策略，应用图着色算法对可交换的门进行调度，同时使用常规有向无环图(Directed Acyclic Graph，DAG) 对不可交换门进行调度。首先，创建变量和数据结构，复制初始路由指令。在详细输出模式下，有关调度过程的统计信息(如 SWAP 门操作数和两量子比特门数)将被获得。接下来，利用图着色算法对初始路由指令进行排序，以减小电路深度。通过考虑依赖关系，循环遍历各个门，并将其调度到相应的周期中。图 4(a)为通用调度器遵循布线阶段提供的门顺序。调度后的电路需 4 个周期。图 4(b)为 2QAN 的混合调度方式，其考虑了对电路门进行重排的灵活性，同时考虑了 SWAP 门与其对应的电路门之间的依赖关系。调度后的电路需 3 个周期。

现有量子调度策略均基于一维量子数组，在分层调度时执行串行操作，未充分利用量子计算的并行性，且忽略了量子电路运行时层内操作可以进一步并行化的事实。故本文在基 2QAN 的量子门调度研究基础上，提出量子萤火虫算法(Quantum Firefly Algorithm，QFA)来优化量子电路深度和并行性，进一步提高量子电路执行效率和电路可靠性。



(a)通用调度器　　(b) 2QAN 调度器

图 4　2QAN 的量子门调度策略

Fig. 4　Quantum gate scheduling strategy of 2QAN

3 所提算法

3.1 萤火虫算法

3.1.1 基本思想

萤火虫算法[20-23]的基本思想是模拟萤火虫群体在夜晚通过发光来互相吸引。每只萤火虫亮度表示适应度(即目标函数值),亮度越高的萤火虫越容易吸引其他萤火虫。而萤火虫运动方向和速度则受其亮度影响,亮度高的萤火虫会吸引亮度较低的萤火虫,后者会向前者移动。这样,在多次迭代过程中,萤火虫群体会趋向于优化问题最优解。

萤火虫的吸引力与相邻萤火虫看到的光强成正比,因此将萤火虫吸引力函数定义为:

$$\beta(r)=\beta_0 e^{-\gamma r_{ij}^2} \tag{1}$$

其中,β_0 为 $r_{ij}=0$ 时的吸引力,γ 为光吸收系数,r_{ij} 表示在 x_i 处和 x_j 处的任意两只萤火虫 i 与萤火虫 j 之间的欧氏距离。

$$r_{ij}=\| x_i-x_j \|=\sqrt{\sum_{k=1}^{d}(x_{i,k}-x_{j,k})^2} \tag{2}$$

其中,$x_{i,k}$ 表示第 i 只萤火虫的空间坐标 x_i 的第 k 个分量。在二维情况下,我们有:

$$r_{ij}=\sqrt{(x_i-x_j)^2+(y_i-y_j)^2} \tag{3}$$

萤火虫 i 被更具吸引力(更亮)的萤火虫 j 吸引时,萤火虫 i 的位置更新公式为:

$$x_i=x_i+\beta_0 e^{-\gamma r_{ij}^2}(x_j-x_i)+\alpha\left(rand-\frac{1}{2}\right) \tag{4}$$

其中,第一项 x_i 表示萤火虫 i 的原位置,第二项是吸引力函数,第三项是随机化函数,α 为随机化参数,$rand$ 是均匀分布在[0,1]的随机数。

如果萤火虫 i 亮度低于萤火虫 j,则萤火虫 i 向萤火虫 j 移动,且移动速度与亮度差异和它们之间距离成正比。如果萤火虫 i 亮度已经足够高,且没有其他萤火虫能进一步吸引它,它就可能在当前解附近进行探索。

3.1.2 算法局限性

(1)萤火虫算法性能很大程度依赖于算法参数,如果参数没有经过精确调优,会导致萤火虫算法收敛到一个次优解或者收敛速度过慢。

(2)萤火虫算法具有一定全局搜索能力,但它仍然可能在某些情况下陷入局部最优。特别是当问题具有多个局部极小值时,萤火虫可能会在某些局部最优解附近徘徊,无法跳出局部解空间。

(3)萤火虫算法多样性不足,尽管萤火虫算法通过引入随机扰动(如步长因子 α)来提高其跳出局部最优的能力,但在没有足够多样性或者在搜索空间较为平坦的区域,萤火虫可能无法有效探索到全局最优解。

(4)萤火虫算法本身缺乏自适应性,即它没有自动调整搜索策略的机制。在实际应用中,搜索策略需要根据当前搜索情况动态调整。例如,在算法初期阶段,可能需要更广泛探索;而在后期阶段,应该进行更精细开发。但萤火虫算法没有内置机制来自动调整探索和开发的平衡,从而可能影响其在复杂问题中的表现。

3.2 量子萤火虫算法

3.2.1 量子萤火虫

传统算法中萤火虫个体只含有自己的位置信息和适应度值,量子萤火虫算法在此基础增加了量子信息。引入量子信息是为了利用量子力学相关理论来增强算法搜索能力和优化性能。在萤火虫算法中,每只萤火虫在任意时刻只能处于一个确定位置,而在量子萤火虫算法中,量子萤火虫通过波函数表示处于叠加态,允许萤火虫同时探索多个位置,从而增加搜索空间覆盖范围。量子信息引入使得量子萤火虫能够通过量子隧穿效应穿越能量障碍,跳出局部最优解;量子态演化和波函数坍缩提供了一种自然机制来平衡探索新解和开发已知解;量子信息处理并行性和高效性使得算法能够更快地收敛到高质量解。通过引入量子信息,量子萤火虫算法能够在处理复杂优化问题时表现出更高效率和更强鲁棒性。

引入量子信息的萤火虫,量子萤火虫表达式为:

$$\begin{cases}\boldsymbol{P}=(p_1,p_2,\cdots,p_n)\\ I=f(\boldsymbol{P})\\ \psi(p_i)=\dfrac{1}{\sqrt{n}},i=1,2,\cdots,n\\ \hbar=1.0\end{cases} \tag{5}$$

其中,$\boldsymbol{P}$ 是位置向量,表示量子萤火虫的位置,是一个长度为 n 序列,p_i 表示量子萤火虫在第 i 个位置的坐标;用 I 表示每只量子萤火虫的亮度,$f(P)$ 表示萤火虫适应度函数值,亮度初始为 0,后续会根据适应度函数更新;$\psi(p_i)$ 表示第 i 个位置上的波函数值,初始时,波函数被初始化为一个均匀叠加态,其中每个元素值为 $1/\sqrt{n}$,即每个位置概率幅度相等;波函数模平方给出了系统各个位置的概率密度;$\hbar$ 为约化普朗克常数。

3.2.2 位置更新机制

引入量子信息后,量子萤火虫通过波函数更新来进行演化,并根据波函数坍缩进行位置更新。

薛定谔方程在量子力学中用于描述一个系统波函数随时间演化,因此我们可以利用时间依赖的薛定谔方程来更新波函数演化。基于时间依赖的薛定谔方程为:

$$i\hbar\frac{\partial\psi(x,t)}{\partial t}=\hat{H}\psi(x,t) \tag{6}$$

其中,$\psi(x,t)$为波函数,依赖位置 x 和时间 t,$\hat{H}$ 为哈密顿量。

$$\hat{H}=-\frac{\hbar^2}{m}\nabla^2+V(x) \tag{7}$$

其中,动能项为$-\frac{\hbar^2}{m}\nabla^2$,势能项为 $V(x)$。当哈密顿量作用于波函数时为:

$$\hat{H}\psi(x)=\left(-\frac{\hbar^2}{m}\nabla^2+V(x)\right)\psi(x) \tag{8}$$

可见势场与动能项一起构成了系统哈密顿量,决定了波函数如何随时间演化。

结合萤火虫算法中位置更新公式将势场 $V(x)$定义为:

$$V(x)=-\beta\cdot I(f_j)\cdot e^{-\gamma|x-x_j|} \tag{9}$$

其中,β 为萤火虫吸引度:

$$\beta=\beta_0 e^{-\gamma r^2} \tag{10}$$

式(10)中,$I(f_j)$表示萤火虫 j 亮度(适应度值),γ 为光吸收系数,$|x-x_j|$是当前位置 x 到萤火虫 j 位置 x_j 的距离。势场参数(β,γ)可以进行调节,β 控制吸引强度,影响收敛速度;γ 控制势场衰减,影响搜索范围。

势场公式中负号表示这是一个势阱,会吸引其他萤火虫,由 $\beta\cdot I(f_j)$可以看出势阱深度与萤火虫亮度和吸引度成正比,亮度越大(适应度值越高)萤火虫产生的势阱越深。$e^{-\gamma|x-x_j|}$为衰减项,距离越远,势能影响越小,光吸收系数 γ

控制势能衰减速度。

动能项$-\frac{\hbar^2}{m}\nabla^2$可以让萤火虫具有量子隧穿效应,即量子势场下波函数演化允许萤火虫以一定概率穿越势垒,跨越局部最优解,其演示图如图 5 所示。在量子势场下,利用波函数演化来进行位置更新,在波函数坍缩后得到概率分布受势场影响,势场越深,波函数振幅越大,探索后在这些区域采样概率更高,从而引导萤火虫移动到更优解。

图 5 展示了 QFA 通过量子隧穿效应可以更有效地探索解空间,避免陷入局部最优。图中黑色曲线表示目标函数势能面,黄色曲线表示量子波函数的概率分布,蓝色曲线表示解的概率密度分布。量子隧穿效应通过波函数的概率分布特性,使得粒子能够"穿透"势能障碍(越过局部最优),从而自然地实现了在不同搜索区域间的跳转,有效避免了陷入局部最优的问题。

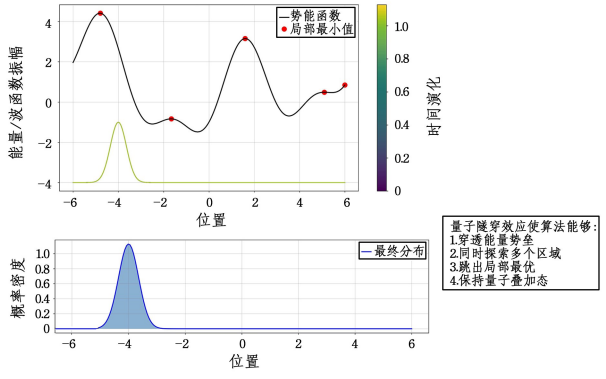


图 5　量子隧穿效应演示图

Fig. 5　Demonstration diagram of the quantum tunneling effect

当哈密顿量作用于波函数时为:

$$\hat{H}\psi(x)=\left(-\frac{\hbar^2}{m}\nabla^2+V(x)\right)\psi(x) \tag{11}$$

波函数时间演化关系为:

$$\frac{\partial\psi(x,t)}{\partial t}=\frac{1}{i\hbar}\hat{H}\psi(x,t) \tag{12}$$

利用 odeint 函数求解上述方程,一般解形式为:

$$\psi(x,t)=\psi(x,0)e^{-iEt/\hbar} \tag{13}$$

其中,$\psi(x,0)$为初始波函数,E 为系统能量。

odeint 函数会在时间步长 Δt 内更新波函数:

$$\psi(x,t+\Delta t)=\psi(x,t)+\Delta t\cdot\frac{1}{i\hbar}\hat{H}\psi(x,t) \tag{14}$$

通过式(14)实现波函数在时间上的演化,同时通过式(15)计算波函数的概率密度:

$$P(x)=\psi(x,t)\psi^*(x,t)=|\psi(x,t)|^2 \tag{15}$$

利用概率密度来实现波函数坍缩,对概率密度进行归一化:

$$P'(x)=\frac{P(x)}{\sum P(x)} \tag{16}$$

对于位置空间$(x_1,x_2,\cdots,x_n)$,每个位置被选中的概率为$P'(x_i)$,波函数模平方越大,被选中的概率越大。进行无放回采样,保证位置不重复,得到新位置排列$(x_{\pi(1)},x_{\pi(2)},\cdots,x_{\pi(n)})$,该过程实现了基于量子态概率分布的位置更新,同时保持了位置唯一性。

区别于传统萤火虫算法,利用波函数演化机制通过薛定谔方程进行量子态演化,使得量子萤火虫具有量子隧穿效应,更容易跳出局部最优解,且能增强系统动态适应力,使系统能自然响应搜索空间变化。通过波函数坍缩,即测量波函数来更新位置,平衡探索新解和开发已知解,并且通过调整量子态演化参数,可以灵活地控制搜索过程中的探索与开发。量子信息的引入使得量子萤火虫算法收敛速度更快,跳出局部最优能力更强,搜索空间覆盖更全面,优化结果更稳定。

此外,在量子萤火虫算法中加入了随机扰动步骤,用来增加量子萤火虫算法的搜索多样性。对于第 i 个萤火虫位置向量 X_i,在迭代中以概率 α 执行随机交换操作:

$$P(swap)=\alpha \tag{17}$$

其中 $\alpha\in[0,1]$是随机扰动系数,用来控制是否进行交换。若执行交换,则继续以下步骤;否则,跳过该步骤。

随机选取两个不同的索引:

$$j_a,j_b\sim Uniform(\{1,2,\cdots,n\}),j_a\neq j_b \tag{18}$$

其中,j_a 和 j_b 是从位置向量维度 n 中随机选择的两个不同索引。

选择了 j_a 和 j_b 后,交换 X_i 中这两个位置元素,其它元素保持不变,得到更新后的位置向量 X_i^{new}:

$$\begin{cases}X_i^{\text{new}}[j_a]=X_i[j_b]\\X_i^{\text{new}}[j_b]=X_i[j_a]\\X_i^{\text{new}}[k]=X_i[k],\forall\, k\neq j_a,j_b\end{cases} \tag{19}$$

通过上述交换操作,可以引入一定随机性,增加探索空间多样性,避免量子萤火虫在局部最优解处停滞。

3.3　量子萤火虫算法流程图和伪代码

量子萤火虫算法伪代码如算法 1 所示。

算法 1　Quantum Firefly Algorithm

```
Input:Objective function f(x),Number of fireflies N,Control parame-
      ters α,β0,γ,ħ,MaxIter
Output:Global best solution
1. Initialize N fireflies:
   For i=1 to N do
     xi←Random_Position()
      ψi←Initial_Wave_Function()
      Ii← f(xi)
   End for
2. best ←Find_Best_Firefly()
3. while iter<MaxIter do
4.     for each firefly i do
5.       for each firefly j do
6.           if Ij>Ii then
7.             r ← |xi−xj|
8.             β ← β0 * exp(−γ * r^2)
9.             V ←Construct_Potential(β,xi,xj)
10.            ψi←Solve_Schrodinger_Equation(V,ħ)
11.            xi←Collapse_Wave_Function(ψi)
12.      end if
13.    end for
14.    Apply_Random_Walk(α)
15.    Ii← f(xi)
16.    Update_Best()
17. end for
18. iter←iter+1
19. end while
20. return best
```

算法流程图如图 6 所示。

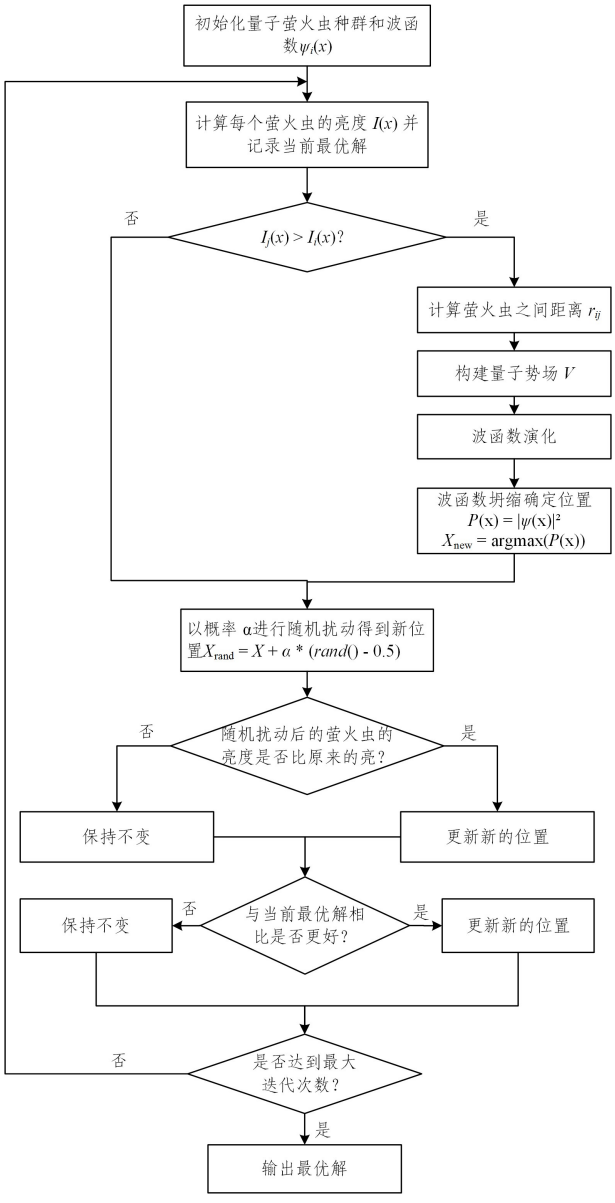


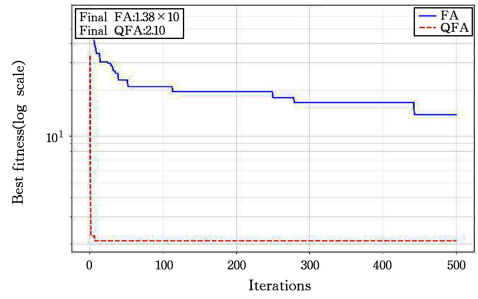
图 6 量子萤火虫算法流程图

Fig. 6 Flowchart of the quantum firefly algorithm

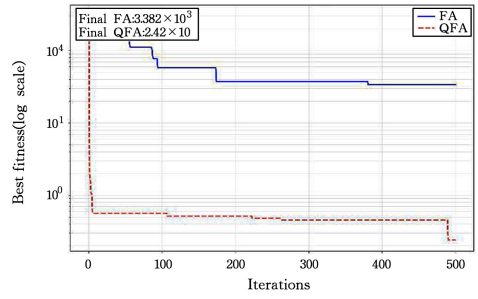
首先初始化 N 个萤火虫，每个萤火虫包含随机初始位置 x_i、初始量子波函数 ψ_i，以及位置 x_i 对应的适应度值 I_i，找出并记录初始种群中的最优解(第 1－2 行)。在每次迭代中对每个萤火虫 i 与其他所有萤火虫 j 进行比较，如果萤火虫 j 更亮($I_j > I_i$)，则计算萤火虫之间的欧氏距离 r 以及吸引度 β，构建量子势能场 V，再通过求解薛定谔方程求解波函数 ψ_i，通过波函数坍缩更新位置 x_i，在位置更新时以 α 概率执行随机扰动，并重新计算适应度值，更新全局最优解(第 3－19 行)。最后返回找到全局最优解(第 20 行)。QFA 算法中的控制参数对算法性能有显著影响，萤火虫个数 N、α 和 γ 是高敏感性参数，而 $\hbar$，MaxIter 和 β_0 是中等敏感性参数。其中普朗克常数 $\hbar$ 影响量子演化速度和波函数扩散程度，最佳取值为 1.0；萤火虫数量 N 影响搜索空间覆盖范围和计算复杂度，建议取值范围在 40～60 之间；最大迭代次数 MaxIter 影响算法收敛性和解的质量，最佳取值在 100～150 之间；随机移动系数 α 影响局部搜索能力和算法稳定性，建议取值在 0.95～0.98 之间；最大吸引度 β_0 影响全局搜索能力和个体间交互强度，最佳取值在 0.8～1.2 之间；光吸收系数 γ 影响搜索范围和局部搜索精度，建议取值在 0.8～1.2 之间。这些参数之间存在交互作用，例如 $\hbar$ 和 N 共同影响算法性能，α 和 γ 共同平衡全局和局部搜索能力。实际应用中，应根据问题规模、特性和计算资源限制来选择合适的参数值。

3.4 性能测试

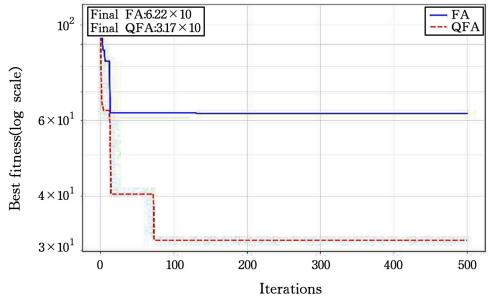
图 7 展示了量子萤火虫算法与萤火虫算法在 sphere 函数、rosenbrock 函数、rastrigin 函数和 ackley 函数上进行收敛性测试的结果，横坐标为迭代次数，纵坐标为适应度值。QFA 与 FA 的参数配置为：萤火虫数量 40 个，最大迭代次数 500 次，$\alpha=0.97$，$\gamma=1.0$，$\beta_0=1.0$，问题维度 10 维，测试次数 10 次。



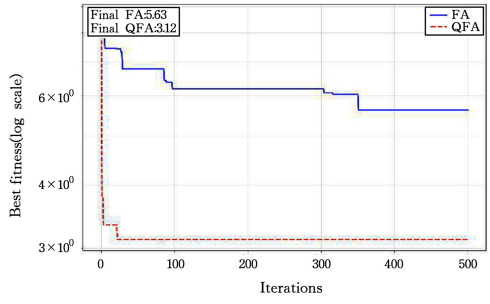
(a) Sphere function convergence



(b) Rosenbrock function convergence



(c) Rastrigin function convergence



(d) Ackley function convergence

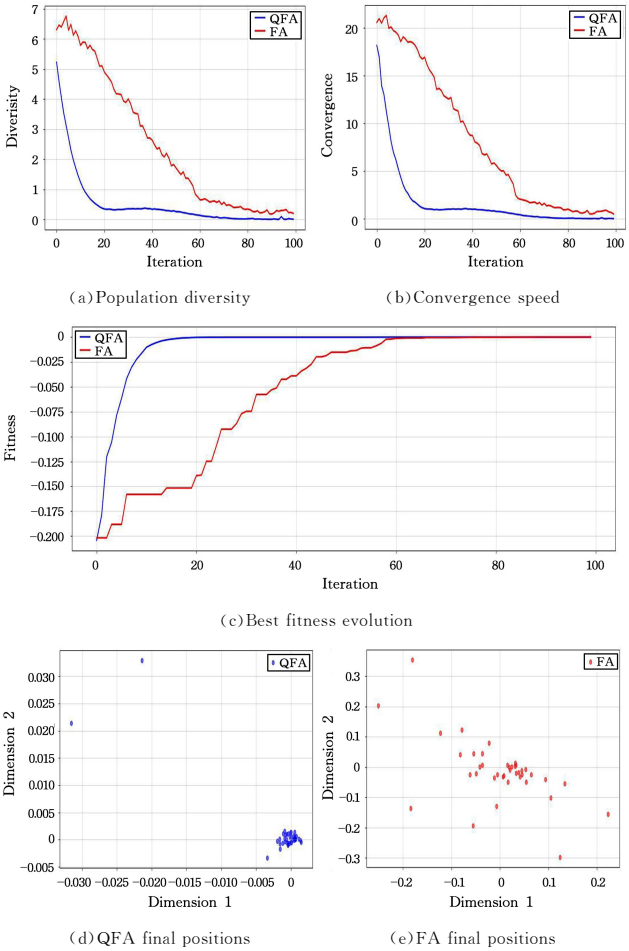
图 7 QFA 收敛性测试图

Fig. 7 Convergence test diagram of QFA

在 4 个测试函数上，QFA 相比 FA 都表现出更快的收敛

速度，QFA 最终到达的适应度值都低于 FA，表明 QFA 所得解的质量更好，QFA 收敛的曲线更加平滑没有大幅波动，表明 QFA 搜索稳定性良好。综合在 4 个测试函数上的测试结果可知，与 FA 相比，QFA 的收敛速度提升约 40%，解的质量提升约 67%，搜索效率提升 45%。可见 QFA 通过引入量子机制，在收敛速度、解的质量、全局搜索能力和算法稳定性等方面都显著优于传统的 FA 算法。

图 8 展示了 QFA 和 FA 的种群多样性、收敛速度、最优适应度进化过程和最终位置（解）分布。测试函数为 Sphere 函数。从种群多样性（Population diversity）图（图 8(a)）可以看出 QFA 在早期保持较高的多样性，体现了其强探索能力，随后多样性逐渐降低，表示算法转向开发阶段。收敛速度（Convergence speed）曲线展现 QFA 快速收敛，结合适应度变化曲线可以看出 QFA 早期适应度快速提升，表明其探索效果较好，中期曲线平滑稳定，适应度稳步改进，展现了较强的适应度曲线，后期与 FA 相比找到了更好的解，说明算法仍保持着适度的探索能力且曲线平滑，说明在开发与探索之间转换较为平稳。由最终位置图看出 QFA 最终位置分布更加集中同时保持着一定的分散性，仍具有一定探索能力。



(a)Population diversity　(b)Convergence speed

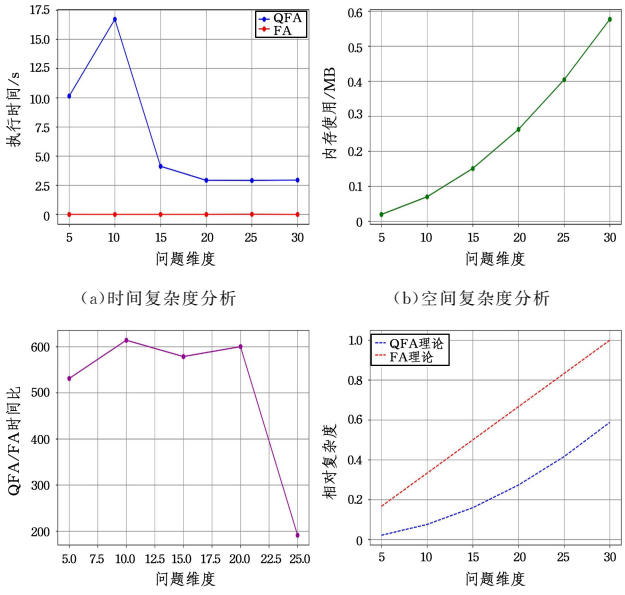
(c)Best fitness evolution

(d)QFA final positions　(e)FA final positions

图 8　QFA 对比分析

Fig. 8　Comparison and analysis of QFA

量子萤火虫算法复杂度为 $O(T*N*(d^2+d*\log(d)))$。其中 T 为总迭代次数，N 为萤火虫种群大小，d 为问题维度。图 9 展示了量子萤火虫的计算复杂度，表明量子萤火虫算法在低维（5～15）性能较差，在中维（15～20）时量子态稳定性提高，QFA 性能得到显著改善，在高维（$d>20$）QFA 性能趋于稳定。



(a)时间复杂度分析　(b)空间复杂度分析

(c)QFA 与 FA 时间比　(d)理论复杂度趋势

图 9　量子萤火虫算法计算复杂度分析

Fig. 9　Analysis of the computational complexity of the quantum firefly algorithm

4　量子萤火虫算法的应用

每个量子萤火虫位置信息表示一个可能的门操作调度顺序，亮度则表示每个方案的优劣程度，波函数表示其量子态，描述调度序列概率分布。

将量子门分为：

（1）初始路由门：可以直接在当前量子比特映射下执行。

（2）未直接路由的门：需要通过 SWAP 操作才能够执行的门。

（3）移动门：SWAP 门，可以分为普通 SWAP 门和 dressed-SWAP 门。dressed-SWAP 门是一种特殊的 SWAP 操作，它将原始电路中量子门操作与 SWAP 操作合并在一起执行，同时完成量子比特交换和原量子电路中的门操作，合并操作以减少量子门数量和电路深度。

从第一层开始，尽可能多地并行安排可以同时执行的门，即在该层内进行了直接路由的门操作、未直接路由的门操作、SWAP 门移动操作后，层内仍有其他门需要调度但因为量子比特冲突，无法在当前层执行，就需要创建新一层。在新一层中首先考虑之前无法执行的门，同时在执行 SWAP 操作后更新量子比特的映射关系，重复该过程，直到所有门都被调度完成。按照从后往前的顺序合并所有层的操作，记录每个操作类型，保存每个操作对应的物理量子比特位置。

根据该分层机制设计量子萤火虫算法适应度函数：

$$f=\omega_1\frac{1}{d+1}+\omega_2\frac{P}{N}+\omega_3\left(1-\frac{m}{g}\right) \tag{20}$$

其中，ω_1，ω_2，ω_3 为权重系数，d 为电路深度，$N(2\leqslant N\leqslant 22)$ 为量子比特总数，m 为移动操作数量，g 为量子门总数。P 为每个周期平均执行门数，即电路平均并行度，计算式如下：

$$P=\frac{\sum_i |C_i|}{n} \tag{21}$$

其中，$|C_i|$ 表示第 i 个周期中可以并行执行的量子门数量，n

为总周期数。在量子电路优化设计过程中，追求量子电路深度最小，并行度最高，移动操作数量最少，以此来增强量子计算的效能与性能。

量子萤火虫算法采用量子-经典混合搜索策略，每个萤火虫除了具有经典位置和亮度信息外，还包含一个量子波函数，用于描述其量子态，其中波函数演化遵循薛定谔方程，通过求解薛定谔方程来更新萤火虫波函数，再通过波函数的坍缩，基于波函数概率分布更新萤火虫位置。同时，算法保留了经典萤火虫算法的吸引度机制，并结合随机扰动策略以增强全局搜索能力，有效避免陷入局部最优解。

利用量子萤火虫算法找到最优调度方式，最后根据得到的最优解即最优的调度方案来执行门操作。

伪代码如算法 2 所示。

算法 2 Quantum Firefly Algorithm for 2QAN Scheduling

Input: Circuit gates G, Number of fireflies N, Max iterations MaxIter, Control parameters $\alpha, \beta_0, \gamma, \hbar$

Output: Optimized scheduled circuit

1. Initialize N fireflies with random schedules and wave functions:
   For i=1 to N do
   F_i.position←Random_Schedule(G)
   F_i.wave_function← $1/\sqrt{|G|}$
   F_i.intensity←Evaluate_Fitness(F_i.position)
   End for
2. best ←Find_Best_Firefly(F)
3. while iter<MaxIter do
4. for each firefly F_i do
5. for each firefly F_j do
6. if F_j.intensity>F_i.intensity then
7. r ← Distance(F_i,F_j)
8. $\beta \leftarrow \beta_0 * \exp(-\gamma * r)$
9. $V \leftarrow -\beta * F_j.\text{intensity} * \exp(-\gamma * |F_i.\text{pos} - F_j.\text{pos}|)$
10. Fi.wave_function←Solve_Schrodinger(V,$\hbar$)
11. F_i.position←Update_Position(F_i.wave_function)
12. end if
13. end for
14. if Random(0,1) < α then
15. Swap_Random_Gates(F_i.position)
16. end if
17. F_i.intensity←Evaluate_Fitness(F_i.position)
18. if F_i.intensity>best.intensity then
19. best ← F_i
20. end if
21. end for
22. iter←iter+1
23. end while
24. return Construct_Circuit(best.position)

创建 N 个萤火虫个体，每个萤火虫 F_i 包含 3 个属性：位置属性 position、波函数 wave_function、适应度函数 intensity。对萤火虫进行初始化，position 为随机生成的初始调度方案，波函数初始化为均匀叠加态 $1/\sqrt{|G|}$，其中 G 表示需要调度的量子门集合，$|G|$ 表示集合 G 的大小，即需要调度的量子门总数(第 1－2 行)。在每次迭代中，将萤火虫 F_i 与其他所有萤火虫 F_j 进行比较，如果萤火虫 F_j 比萤火虫 F_i 更亮，则计算萤火虫之间的欧氏距离 r 和吸引度 β，构建量子势能场 V，再通过求解薛定谔方程求解 F_i 波函数，通过波函数坍缩更新 F_i 位置(第 4－13 行)。以概率 α 对萤火虫位置进行随机扰动，通过随机交换门位置实现扰动，重新计算萤火虫适应度并更新全局最优解(第 14－20 行)。根据找到的最优调度方案构建最终量子电路(第 24 行)。

5 实验结果

5.1 评价指标

本文使用以下指标来评估不同编译器性能：插入 SWAP 门总数和在硬件上执行的两量子比特门总数。较少门数表示拥有更好的性能。这些测试能够有效衡量算法在处理复杂量子电路时的效率。基准测试选自 IBM 的 Qiskit 量子程序，对 CX/CNOT 门集，使用 Qiskit 编译器进行分解和优化。采用基准测试方法，主要关注 QAOA 模型。评估范围涵盖了量子位数从 4～22，每个量子位数的映射过程运行 5 次，并选择其中的最佳结果进行分析。

量子萤火虫算法是在 Python3.9 中实现，执行所有编译的笔记本电脑配置为 Intel Core i7 处理器(5.0 GHz 和 16 GB RAM)。

5.2 模型评估

表 1－表 4 展示了 QFA 应用到 2QAN 中优化后编译开销与 t|ket⟩、Qiskit、HQAA、LCRA、LTSA，以及 LCRA 和 LTSA 两者结合策略的比较。本文装配了推荐的“FullPass”的 t|ket⟩编译器(版本 0.11.0)和 Qiskit 编译器(版本 0.26.2，优化级别为 3)，评估量子计算机编译结果，且上述两个编译器仅限于 CNOT 或 CZ 门集。

表 1 Qiskit 上 QFA 与其他算法的 SWAP 编译成本比较

Table 1 Comparison of SWAP compilation costs between QFA and other algorithms on Qiskit

Qubits	Qiskit	2QAN	HQAA	LCRA	LTSA	combination	QFA
22	123	45	49	35	36	41	37
20	93	35	41	33	33	32	32
18	75	36	32	30	30	30	28
16	55	27	28	25	26	24	24
14	68	25	25	22	22	22	21
12	42	20	22	18	18	18	18
10	28	19	17	17	17	17	17
8	18	12	12	12	12	12	12
6	10	9	9	9	9	9	9
4	8	6	6	6	6	6	6

表 2 Qiskit 上 QFA 与其他算法的 CNOT 编译成本比较

Table 2 Comparison of CNOT compilation costs between QFA and other algorithms on Qiskit

Qubits	Qiskit	2QAN	HQAA	LCRA	LTSA	combination	QFA
22	105	106	109	80	81	81	83
20	84	79	83	75	73	71	72
18	78	88	73	68	67	68	63
16	69	60	61	56	56	55	54
14	57	56	57	48	48	50	46
12	48	43	47	40	40	40	40
10	43	47	40	38	41	38	38
8	30	26	28	26	26	26	26
6	24	20	20	20	20	20	20
4	15	13	13	13	13	13	13

表 3　t|ket〉上 QFA 与其他算法的 SWAP 编译成本比较

Table 3　Comparison of SWAP compilation costs between QFA and other algorithms on t|ket〉

Qubits	t|ket〉	2QAN	HQAA	LCRA	LTSA	combination	QFA
22	65	41	40	37	37	38	37
20	58	34	36	32	32	35	32
18	46	30	30	29	29	29	28
16	40	27	28	24	25	24	24
14	43	22	24	22	21	23	21
12	35	19	20	18	18	18	18
10	24	17	17	17	15	17	17
8	18	12	12	12	12	12	12
6	11	9	9	9	9	9	9
4	8	6	6	6	6	6	6

表 4　t|ket〉上 QFA 与其他算法的 CNOT 编译成本比较

Table 4　Comparison of CNOT compilation costs of QFA and other algorithms on t|ket〉

Qubits	t|ket〉	2QAN	HQAA	LCRA	LTSA	combination	QFA
22	102	95	93	81	82	80	83
20	82	77	83	72	72	72	72
18	75	68	71	66	65	66	63
16	64	60	67	54	56	54	54
14	57	48	53	47	46	45	46
12	45	44	45	40	40	40	40
10	42	39	40	37	35	37	38
8	30	26	26	26	26	26	26
6	24	20	20	20	20	20	20
4	15	13	13	13	13	13	13

根据表中对比结果可以观察到随着量子比特数量增加,优化算法效果越来越明显。

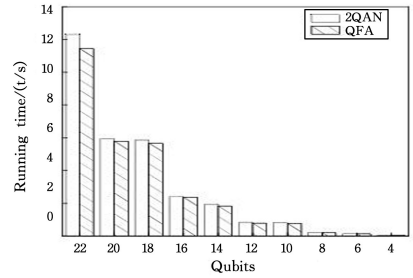
在 t|ket〉编译器中,QFA 通过量子波函数演化和经典搜索相结合的机制显著提升了量子电路优化效果。由于量子波函数能够同时探索多个可能状态,并通过波函数坍缩确定最优位置,使得 QFA 与传统 t|ket〉相比可以减少约 42% SWAP 门数以及约 15.6%CNOT 门数。

与 2QAN 相比,QFA 通过引入量子概率振幅来指导搜索方向,平均减少约 6.7%SWAP 门数和约 11%CNOT 门数。在 18～22 量子比特数规模电路中,量子波函数的全局搜索能力会随着问题规模的增大而更显优势。由图 10 可见,QFA 减少了操作门等待时间,提高了运算效率,增强了算法性能。

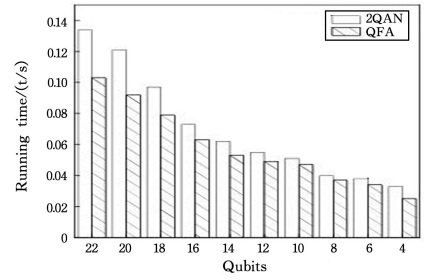
与 HQAA 相比,QFA 利用波函数演化特性来避免局部最优,平均减少约 10.4% SWAP 门数以及约 11.8% CNOT 门数。随着量子比特数的增多,量子态叠加性为算法提供了更大搜索空间,增强了全局搜索能力,使得优化效果更好。

与 LCRA,LTSA,以及 LCRA 和 LTSA 结合策略相比,QFA 在优化 SWAP 门数量上约提升 3%,这是因为这些算法都采用了较为成熟的局部搜索策略。在 CNOT 门优化方面,由于这些算法都基于相似的门分解原理,且都达到了接近理论下界的优化效果,因此算法之间的差距不大,减少约 2%。

在 4～12 量子比特时各算法优化性能差距不大,在 12～18 量子比特 QFA 开始显示优势,在 18～22 量子比特情况下 QFA 表现最佳。QFA 对大规模电路有更好的处理能力,在 SWAP 门和 CNOT 门优化上都有更好的处理能力,性能波动较小,稳定性强。同时,QFA 增强了量子电路并行度,与传统算法相比优化了约 42% SWAP 门,与其他现代算法相比优化了 5～10% SWAP 门。SWAP 门减少意味着更多门可以并行执行,减少串行依赖,提高了电路执行效率。QFA 显著减少不必要 SWAP 操作,优化了量子比特间数据移动,减少了中间态转换开销。此外 CNOT 门也显著减少,表明 QFA 优化了量子门组合方式,减少了冗余量子门操作,提高了量子电路执行效率。SWAP 门和 CNOT 门数减少,使得电路能够更有效地利用量子比特,提高量子资源使用效率;门数减少,有效降低了电路中积累错误的可能,提高了电路可靠性。



(a)在 t|ket〉上 QFA 与 2QAN 运行时间对比



(b)在 Qiskit 上 QFA 与 2QAN 运行时间对比

图 10　QFA 与 2QAN 运行时间对比

Fig. 10　Comparison of running time between QFA and 2QAN

结束语　本文为量子电路调度的优化问题设计了一种量子萤火虫算法。量子萤火虫算法在量子电路调度问题中展现出独特优势。

(1)实现了量子-经典混合优化,通过将量子计算特性与群体智能优化相结合,利用量子波函数演化增强种群搜索能力,同时借助量子叠加和干涉效应扩展解空间,实现经典优化与量子特性的优势互补。

(2)采用多层次方法优化策略,从群体、个体、局部到全局层面形成完整优化体系,既保证了搜索广度,又确保了优化深度。在调度性能方面,算法能够同时优化电路深度、并行度、量子比特移动等多个目标,通过适应度函数实现目标间的灵活权衡。

(3)与传统量子电路调度算法相比,量子萤火虫算法在收敛速度、解的质量、多目标优化能力等方面都表现出了明显优势。QFA 与 FA 相比收敛速度提升约 40%,解的质量提升约 67%,搜索效率提升 45%。此外,量子萤火虫算法对不同规模问题具有良好适应性和鲁棒性。这些特性使得该算法能够更好地处理复杂量子线路调度问题,在优化效果和计算效率方面都取得了显著提升。与传统算法相比,量子萤火虫算法 SWAP 门数平均减少 42%,CNOT 门数平均减少 15.6%。与现代算法相比,改进算法 SWAP 门数平均减少 7.3%,CNOT 门数平均减少 10.8%。

量子萤火虫算法为量子电路中的调度问题提供了创新有效的解决途径。但量子萤火虫算法在低维度计算开销大、内存需求高等方面存在不足。通过改进量子演化计算方法、开发混合量子经典算法以及拓展多目标优化等,可望在未来

实现更高效稳定的优化性能。

参 考 文 献

[1] PRESKILL J. Quantum computing in the NISQ era and beyond [J]. Quantum, 2018, 2: 79.

[2] HUANG H K, ZHANG X S. Qubit Mapping Algorithm for Noisy Intermediate-Scale Quantum Computers [J]. Computer Engineering and Applications, 2024, 60(24): 110-118.

[3] ITOKO T, IMAMICHI T. Scheduling of operations in quantum compiler[C]// 2020 IEEE International Conference on Quantum Computing and Engineering(QCE). IEEE, 2020: 337-344.

[4] VENTURELLI D, DO M, RIEFFEL E, et al. Temporal planning for compilation of quantum approximate optimizationcircuits [C]// Scheduling and Planning Applications Workshop(SPARK). 2017: 58.

[5] METODI T S, THAKER D D, CROSS A W, et al. Scheduling physical operations in a quantum information proces-sor[C]// Quantum Information and Computation IV. SPIE, 2006: 210-221.

[6] SHI Y, LEUNG N, GOKHALE P, et al. Optimized compilation of aggregated instructions for realistic quantum computers [C]// Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems. 2019: 1031-1044.

[7] GUERRESCHI G G, PARK J. Gate scheduling for quantum algorithms[J]. Quantum Science and Technology, 2018, 3(4): 045003.

[8] JOHNSON M W, AMIN M H S, GILDERT S, et al. Quantum annealing with manufactured spins[J]. Nature, 2011, 473(7346): 194-198.

[9] DEVITT S J. Performing quantum computing experiments in the cloud[J]. Physical Review A, 2016, 94(3): 032329.

[10] BARENDS R, SHABANI A, LAMATA L, et al. Digitized adiabatic quantum computing with a superconducting circuit[J]. Nature, 2016, 534(7606): 222-226.

[11] VERSLUIS R, POLETTO S, KHAMMASSI N, et al. Scalable quantum circuit and control for a superconducting surface code [J]. Physical Review Applied, 2017, 8(3): 034021.

[12] SETE E A, ZENG W J, RIGETTI C T. A functional architecture for scalable quantum computing[C]// 2016 IEEE International Conference on Rebooting Computing (ICRC). IEEE, 2016: 1-6.

[13] GUERRESCHI G G, PARK J. Two-step approach to scheduling quantum circuits[J]. Quantum Science and Technology, 2018, 3(4): 045003.

[14] LAO L, VAN SOMEREN H, ASHRAF I, et al. Timing and resource-aware mapping of quantum circuits tosupercon ducting processors[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2021, 40(2): 359-371.

[15] ALAM M, ASH-SAKI A, GHOSH S. Circuit compilation methodologies for quantum approximate optimization algorithm [C]// Proceedings of the 2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture. 2020: 215-228.

[16] ALAM M, ASH-SAKI A, GHOSH S. An efficient circuit com pilation flow for quantum approximate optimization algorithm [C]// Proceedings of the 2020 57th ACM/IEEE Design Auto mation Conference, 2020: 1-6.

[17] LIU L, DOU X. QuCloud+: A holistic qubit mapping scheme for single/multi-programmingon 2D/3D NISQ quantum computers[J]. ACM Transactions on Architecture and Code Optimization, 2024, 21(1): 1-27.

[18] SILVA A, ZHANG X, WEBB Z, et al. Multi-qubit Lattice Surgery Scheduling[J]. arXiv: 2405. 17688, 2024.

[19] LAO L, BROWNE D E. 2qan: A quantum compiler for 2 local qubit hamiltonian simulation algo rithms[C]// Proceedings of the 49th Annual International Symposium on Computer Architecture. 2022: 351-365.

[20] YANG X S. Firefly algorithm, stochastic test functions and design optimisation[J]. International journal of bio-inspired computation, 2010, 2(2): 78-84.

[21] CHAUDHARY K. A Modified Firefly Algorithm for Solving Optimization Problems[J]. International Journal of Computational Intelligence and Applications, 2024: 2450012.

[22] YANG X S. Firefly algorithms for multimodal optimization [C]// Internationalsymposium on stochastic algorithms. Berlin: Springer, 2009: 169-178.

[23] LIU X N, AN J L, HE M, et al. Chaotic Adaptive Quantum Firefly Algorithm[J]. Computer Science, 2023, 50(4): 204-211.

LI Hui, born in 1985, Ph.D, professor, M. S. supervisor, is a member of CCF (No. K9013M). His main research interests include quantum computing and quantum information processing, etc.

WANG Jiepeng, born in 2002, postgraduate. His main research interest is quantum circuit optimization.