第5章

基本的量子算法

量子计算的研究始于 20 世纪 80 年代。1982 年，Benioff 和 Feynman 最早提出了量子计算的概念。Feynman 指出，用经典计算机模拟量子力学系统存在本质的困难，当量子系统规模较大时，所需的计算代价将会呈指数级上升，即对 n 个量子粒子（例如二能级原子）系统的经典模拟需要存储 2^n 个复振幅。因此，需要以指数大小存储这些信息。进一步地，他提出构造基于量子力学原理的计算机以克服这些困难，利用量子计算机使用 n 量子比特就可以自然地表示这些振幅。

1985 年，Deutsch 给出了第一个量子计算模型，一个在一台经典计算机上需要两次查询才能解决的黑盒问题，在量子计算机上仅用一次量子查询就可以解决。一系列相关的结果（Deutsch-Jozsa 算法，1992 年；Bernstein-Vazirani 算法，1997 年）给出了经典查询和量子查询复杂性之间显著的差别。最终，Simon（1997 年）的一个例子提供了一个指数级加速。正是以这些工作为基础，才诞生了后面两个重要的量子算法——Shor 大数因子分解算法和 Grover 搜索算法。从此，量子优势也日益凸显，量子计算迅速引起全世界学者的关注，它利用量子系统所特有的叠加性、相干性和纠缠性，通过设计相关算法可以实现大规模并行计算，相对于经典计算，量子计算在信息存储、执行速度等方面可以实现本质上的超越。

此外，量子算法的一个相关技术是绝热演化。量子绝热理论保证了基态下的量子系统在哈密顿量改变时将保持接近基态，前提是这种变化足够慢，这取决于哈密顿量的光谱特性。绝热演化可以用于解决优化问题以及模拟通用量子计算机。量子近似优化算法（QAOA）正是基于绝热理论解决优化问题的一类主要算法。

本章主要介绍几种基本的量子算法：Deutsch-Jozsa 算法、Simon 算法、Bernstein-Vazirani 算法和 QAOA 算法。

5.1 Deutsch-Jozsa 算法

5.1.1 量子并行性

量子并行性是许多量子算法的基本特征之一，量子计算的并行性不

同于经典计算的并行性，量子并行性使量子计算机可以同时计算函数 $f(x)$ 在许多不同 x 处的值。

设 $f(x):\{0,1\}\to\{0,1\}$ 是具有单比特定义域和值域的函数，在量子计算机上，计算该函数的一个简单方法是：考虑初态为 $|x\rangle|y\rangle$ 的双量子比特计算机，通过适当的逻辑门序列可以把这个状态转换为 $|x\rangle|y\oplus f(x)\rangle$，其中 $\oplus$ 为模 2 加法。通常，量子比特的集合称为量子寄存器，简称寄存器。在这里，第一个寄存器 $|x\rangle$ 称为数据寄存器，第二个寄存器 $|y\rangle$ 称为目标寄存器，映射 $|x\rangle|y\rangle\longrightarrow|x\rangle|y\oplus f(x)\rangle$ 称为 U_f，且 U_f 为酉的。若 $|y\rangle=|0\rangle$，则第二个量子比特的最终状态为 $f(x)$。输入 $|x\rangle|y\rangle$ 映射为输出 $|x\rangle|y\oplus f(x)\rangle$ 的量子线路如图 5.1 所示。

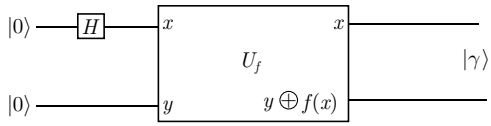

图 5.1　同时计算 $f(0)$ 和 $f(1)$ 的量子线路

在图 5.1 中，实现了 $|x\rangle|y\rangle\longrightarrow|x\rangle|y\oplus f(x)\rangle$。数据寄存器 $|x\rangle$ 输入的是叠加态 $\dfrac{|0\rangle+|1\rangle}{\sqrt{2}}$，再应用 U_f，即

$$\frac{|0\rangle+|1\rangle}{\sqrt{2}}|0\rangle\xrightarrow{U_f}\frac{|0\rangle|0\oplus f(0)\rangle+|1\rangle|0\oplus f(1)\rangle}{\sqrt{2}}\tag{5.1.1}$$

由于 $|0\oplus f(0)\rangle=|f(0)\rangle$，且 $|0\oplus f(1)\rangle=|f(1)\rangle$，故得到状态

$$\frac{|0\rangle|f(0)\rangle+|1\rangle|f(1)\rangle}{\sqrt{2}}\tag{5.1.2}$$

其中，式 (5.1.2) 中不同的项同时包含 $f(0)$ 和 $f(1)$，看起来似乎对 x 的两个值计算了 $f(x)$。经典计算的并行是多重 $f(x)$ 电路同时进行，而这里利用了量子计算机处于不同状态的叠加能力，因此单个 $f(x)$ 线路就可以同时计算多个 x 的值。

利用 Hadamard 变换（H 门），如图 5.2 所示，可以将这个过程推广到任意数目量子比特上的函数计算，该变换是把 n 个 Hadamard 门同时分别应用到 n 量子比特上。例如，当 $n=2$ 时，初态全为 $|0\rangle$ 的情况如下所示。

$$|0\rangle|0\rangle\xrightarrow{H^{\otimes2}}\left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)=\frac{|00\rangle+|01\rangle+|10\rangle+|11\rangle}{2}\tag{5.1.3}$$

其中，$H^{\otimes2}$ 表示两个 Hadamard 门的并行作用。

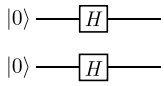

图 5.2　双量子比特上的 Hadamard 变换

更一般地，n 量子比特上的 Hadamard 变换从全 0 出发，即

$$|0\rangle^{\otimes n}\xrightarrow{H^{\otimes n}}\left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)^{\otimes n}\tag{5.1.4}$$

得到状态

$$\frac{1}{\sqrt{2^n}}\sum_x |x\rangle \tag{5.1.5}$$

其中，求和是对 x 的所有可能取值且 x 的长度为 n，并用 $H^{\otimes n}$ 表示这个作用。可以看到，Hadamard 变换产生了所有基态的平均叠加 $\left(\text{所有状态的概率幅全为 } \frac{1}{\sqrt{2^n}}\right)$，仅用 n 个 H 门就可以产生 2^n 个状态的叠加。

对于单比特输入和单比特输出，可以采用下述方法将其扩展为 n 比特输入 x 和单比特输出 $f(x)$，从而实现函数 $f(x)$ 对多个 x 的并行计算。为了制备 $n+1$ 量子比特的状态 $|0\rangle^{\otimes n}|0\rangle$，对前 n 位应用 Hadamard 变换，并连接实现 U_f 的量子线路，这样就产生了状态

$$\frac{1}{\sqrt{2^n}}\sum_x |x\rangle|f(x)\rangle \tag{5.1.6}$$

例如，当 $n=3$ 时，为了制备 4 量子比特的状态 $|0\rangle^{\otimes 3}|0\rangle$，对前 3 量子比特同时应用 Hadamard 变换，得到数据寄存器 x 中的状态为

$$\frac{1}{\sqrt{2^3}}(|000\rangle+|001\rangle+|010\rangle+|011\rangle+|100\rangle+|101\rangle+|110\rangle+|111\rangle) \tag{5.1.7}$$

再连接实现 U_f 的量子线路，产生状态

$$\begin{aligned}&\frac{1}{\sqrt{2^3}}(|000\rangle|f(000)\rangle+|001\rangle|f(001)\rangle+|010\rangle|f(010)\rangle+|011\rangle|f(011)\rangle+|100\rangle|f(100)\rangle\\&+|101\rangle|f(101)\rangle+|110\rangle|f(110)\rangle+|111\rangle|f(111)\rangle)\end{aligned} \tag{5.1.8}$$

在某种意义下，在表面上似乎只进行一次 $f(x)$ 计算，量子并行性却使得 $f(x)$ 的所有可能值同时被计算出来。一般而言，由于对量子态的测量会使量子状态发生坍缩现象，因此测量状态 $\sum_x |x\rangle|f(x)\rangle$ 也类似地只给出了某一个 x 的 $f(x)$ 值，显然利用经典计算就能做到这一点。为了真正地从量子并行性中获得好处，不仅仅在于能够获得一个 $f(x)$ 的值，更为重要的是，能够得到比从一个叠加态 $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$ 中获得一个 $f(x)$ 值更有价值的信息抽取能力，下面介绍的 Deutsch 算法就是这种能力的具体表现。

5.1.2 Deutsch 算法简介

Deutsch 算法是基于量子并行性和量子力学中的干涉性质结合起来实现的，显示了量子相干性的强大计算能力。考虑一个黑箱（Oracle），它可以计算一比特的函数 $f(x):\{0,1\}\to\{0,1\}$，每做一次计算，我们称对黑箱做一次查询。

存在 4 个这样的函数，

$$f_1(x)=x,\quad f_2(x)=\overline{x},\quad f_3(x)=0,\quad f_4(x)=1 \tag{5.1.9}$$

其中，$x\in\{0,1\}$，$\overline{x}$ 是 x 的逻辑非。对于 $f_1(x)$、$f_2(x)$，称它们为平衡的；对于 $f_3(x)$、$f_4(x)$，称它们为常数的。所要解决的问题是：如何用一次计算确定未知的函数 $f(x)$ 是常数的还

是平衡的？要得到该问题的答案，经典计算机需要对黑箱做两次查询，而量子算法只需要一次。Deutsch 算法的量子线路如图 5.3 所示。

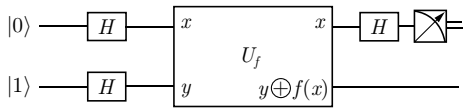

图 5.3　实现 Deutsch 算法的量子线路

由图 5.3 可知，Deutsch 算法利用 Hadamard 门把第 1 量子比特 $|0\rangle$ 制备为叠加态 $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$，同样地，把 Hadamard 门应用到辅助量子比特 $|1\rangle$ 上制备叠加态 $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$。Deutsch 量子算法的步骤如下所述。

算法 5.1　Deutsch 算法

（1）输入状态 $|\varphi_0\rangle=|01\rangle$。

（2）$|\varphi_0\rangle$ 通过两个 Hadamard 门后，输出状态为

$$|\varphi_1\rangle=\left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \tag{5.1.10}$$

（3）容易看出，对于每个 $x\in\{0,1\}$，如果把 $|x\rangle\frac{1}{\sqrt{2}}(|0\rangle-|1\rangle)$ 作为 U_f 的输入，就可以得到 U_f 的输出状态为 $(-1)^{f(x)}|x\rangle\frac{1}{\sqrt{2}}(|0\rangle-|1\rangle)$，即

$$U_f|x\rangle\frac{1}{\sqrt{2}}(|0\rangle-|1\rangle)=(-1)^{f(x)}|x\rangle\frac{1}{\sqrt{2}}(|0\rangle-|1\rangle) \tag{5.1.11}$$

于是，把 $|\varphi_1\rangle$ 作为 U_f 的输入，即

$$U_f\left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)=\frac{1}{\sqrt{2}}(-1)^{f(0)}|0\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}}+\frac{1}{\sqrt{2}}(-1)^{f(1)}|1\rangle\frac{|0\rangle-|1\rangle}{\sqrt{2}} \tag{5.1.12}$$

就得到了两种可能的 U_f 输出，即

$$|\varphi_2\rangle=\begin{cases}\pm\left(\dfrac{|0\rangle+|1\rangle}{\sqrt{2}}\right)\left(\dfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right), & f(0)=f(1)\\ \pm\left(\dfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right)\left(\dfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right), & f(0)\neq f(1)\end{cases} \tag{5.1.13}$$

（4）第 1 量子比特再通过 Hadamard 门，输出变为

$$|\varphi_3\rangle=\begin{cases}\pm|0\rangle\left(\dfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right), & f(0)=f(1)\\ \pm|1\rangle\left(\dfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right), & f(0)\neq f(1)\end{cases} \tag{5.1.14}$$

如果 $f(0)=f(1)$，那么 $f(0)\oplus f(1)=0$；如果 $f(0)\neq f(1)$，那么 $f(0)\oplus f(1)=1$。因此，可将上述结果改写成

$$|\varphi_3\rangle=\pm|f(0)\oplus f(1)\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \tag{5.1.15}$$

（5）测量第 1 量子比特，就可以确定 $f(0)\oplus f(1)$ 的值。

于是，如果函数是常数的，对于第 1 量子的测量就会百分之百地得到 0；而如果函数是平衡的，那么测量结果肯定是 1。因此，应用量子线路只对函数 $f(x)$ 一次计算的情况下，就可以确定 $f(x)$ 是常数的还是平衡的全局性质。完成上述过程的经典计算机至少需要两次计算，而量子计算只需要一次。

上述 Deutsch 量子算法的量子并行性显而易见，它体现了很多量子算法设计的本质特征，即通过精心选择函数和最终变换，从而有效地确定有关函数的有用全局信息，而这在经典计算机上是无法快速得到上述结果的。

5.1.3 Deutsch-Jozsa 算法简介

对于一个或为常数的或为平衡的 n 比特二进制函数 $f:\{0,1\}^n\to\{0,1\}$，确定它为常数的还是平衡的问题可以描述为 Alice 从 0 到 2^n-1 的数中选一个数 x，以书信的形式把 x 寄给 Bob。Bob 收到信后，根据信中 x 的值计算 $f(x)$ 的值，然后以书信的形式把 $f(x)$ 的值寄回给 Alice。Bob 使用的函数只有两种类型，即对于所有的 x，$f(x)$ 要么是常数的，要么是平衡的。若 $f(x)$ 是平衡的，也就是说，恰好所有可能 x 值的一半使得函数值为 1，另一半使得函数值为 0。Alice 的目的是要用尽可能少的通信确定 Bob 所用的函数 $f(x)$ 是平衡的还是常数的。在经典算法的情况下，Alice 的一封信只能发给 Bob 一个 x 值，在最坏的情况下，Alice 要问 Bob 至少 $2^n/2+1$ 次才能确定函数的类型，而 Deutsch-Jozsa 算法只需要通过对黑箱（Oracle）做一次询问就可以解决这个问题，其量子线路与 Deutsch 算法的量子线路类似，如图 5.4 所示。

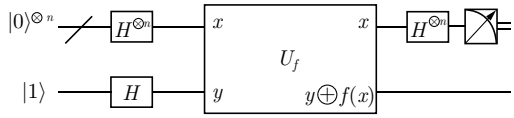

图 5.4 实现 Deutsch-Jozsa 算法的量子线路

在图 5.4 中，带有“/”的直线表示通过此直线的是一组 n 量子比特，Hadamard 门被并行地作用到 n 量子比特上，即

$$H^{\otimes n}=H\otimes H\otimes H\otimes\cdots\otimes H \tag{5.1.16}$$

下面观察线路中的状态变化。输入状态为

$$|\varphi_0\rangle=|0\rangle^{\otimes n}|1\rangle \tag{5.1.17}$$

$|\varphi_0\rangle$ 在通过 Hadamard 门，得到

$$|\varphi_1\rangle=\sum_{x\in\{0,1\}^n}\frac{|x\rangle}{\sqrt{2^n}}\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \tag{5.1.18}$$

接下来，使用 $U_f:|x\rangle|y\rangle\to|x\rangle|y\oplus f(x)\rangle$ 计算函数 $f(x)$, 得到

$$|\varphi_2\rangle=\sum_x(-1)^{f(x)}\frac{|x\rangle}{\sqrt{2^n}}\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \tag{5.1.19}$$

当 $x=0$ 或 $x=1$ 时，我们知道，对单量子比特 $|x\rangle$ 有

$$H|x\rangle=\sum_{z\in\{0,1\}}(-1)^{xz}|z\rangle/\sqrt{2} \tag{5.1.20}$$

于是，对输入 n 量子比特 $|x\rangle=|x_0,x_1,\cdots,x_{n-1}\rangle$ 有

$$\begin{aligned}H^{\otimes n}|x\rangle&=H^{\otimes n}|x_0,x_1,\cdots,x_{n-1}\rangle\\&=\frac{\sum\limits_{z_0\cdots z_{n-1}}(-1)^{x_0z_0\oplus x_1z_1\oplus x_2z_2\oplus\cdots\oplus x_{n-1}z_{n-1}}|z_0,z_1,\cdots,z_{n-1}\rangle}{\sqrt{2^n}}\end{aligned} \tag{5.1.21}$$

上式可以写成更加紧凑的形式，即

$$H^{\otimes n}|x\rangle=\frac{\sum\limits_z(-1)^{x\cdot z}|z\rangle}{\sqrt{2^n}} \tag{5.1.22}$$

其中，$x\cdot z$ 表示对 x 和 z 取模 2 的内积，即

$$x\cdot z=x_0z_0\oplus x_1z_1\oplus x_2z_2\oplus\cdots\oplus x_{n-1}z_{n-1} \tag{5.1.23}$$

因此，我们可以利用式 (5.1.22) 对 $|\varphi_2\rangle$ 的前 n 个量子比特应用并行的 Hadamard 门，得到

$$|\varphi_3\rangle=\sum_z\sum_x\frac{(-1)^{x\cdot z+f(x)}|z\rangle}{2^n}\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right) \tag{5.1.24}$$

对于该输出态 $|\varphi_3\rangle$，在计算基上测量这 n 个量子比特，如果 f 为常数的，则测量以 100% 的概率得到状态 $|000\cdots0\rangle$；如果 f 是平衡的，那么测量到这个态的概率为 0。Deutsch-Jozsa 算法可总结如下。

算法 5.2　Deutsch-Jozsa 算法

输入　对 $x\in\{0,1,\cdots,2^n-1\}$ 和 $f(x)\in\{0,1\}$ 进行变换 $|x\rangle|y\rangle\to|x\rangle|y\oplus f(x)\rangle$ 的 U_f，已知对所有的 x 而言，$f(x)$ 是常数的或者平衡的。

输出　当且仅当 $f(x)$ 是常数的，输出为 0。

运行时间　计算 U_f 一次，总是成功的。

过程

(1) 初始化状态：$|0\rangle^{\otimes n}|1\rangle$。

(2) 使用并行的 Hadamard 门产生叠加态：$\dfrac{1}{\sqrt{2^n}}\sum\limits_{x=0}^{2^n-1}|x\rangle\left(\dfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right)$。

(3) 使用 U_f 计算 $f(x)$：$\dfrac{1}{\sqrt{2^n}}\sum\limits_x(-1)^{f(x)}|x\rangle\left(\dfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right)$。

(4) 进行 Hadamard 变换：$\sum\limits_z\sum\limits_x\dfrac{(-1)^{x\cdot z+f(x)}|z\rangle}{2^n}\left(\dfrac{|0\rangle-|1\rangle}{\sqrt{2}}\right)$。

(5) 测量最终输出 z。

因此，运行一次 Deutsch-Jozsa 算法，对函数 $f(x)$ 仅查询一次就可以确定 $f(x)$ 是常数的还是平衡的。但是在经典计算机上，只有经过 $2^{n-1}+1$ 次查询之后才可以确定函数 $f(x)$ 是否是常数的。但是，对于 Deutsch-Jozsa 算法有几点需要注意。首先，Deutsch 问题不是一个具有实质重要性的问题，它没有已知的应用；其次，在某种角度上，经典算法和量子算法很难相互比较，因为计算函数值的方法差别很大；最后，如果选择概率经典计算机，随机选择一些 x 值计算函数 $f(x)$ 的值，则也可以以非常高的概率确定 $f(x)$ 是常数的还是平衡的，这种概率模型比我们考虑的确定性模型或许更加现实。

5.2 Simon 算法

给定一个映射 $f:\mathbb{Z}_2^n \to \mathbb{Z}_2^m$，其中 $n \geqslant m$，存在一个 $s \in \mathbb{Z}_2^n$ 对所有的 $x, x' \in \mathbb{Z}_2^n$，满足性质 $f(x)=f(x')$ 的充分必要条件是 $x=x'$ 或 $x \oplus x'=s$，其中 $\oplus$ 表示模 2 加法。Simon 算法通过一系列量子计算步骤可以求出满足上述条件的 s。该问题的经典算法复杂度是指数级的，而 Simon 算法可以在多项式时间内求解，这说明了对于特定问题，量子算法比经典算法更快。Simon 算法的量子线路如图 5.5 所示。

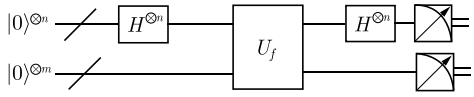

图 5.5 Simon 算法的量子线路

Simon 算法的步骤描述如下。

算法 5.3 Simon 算法

（1）首先将两个量子寄存器初始化为 $|0^n\rangle|0^m\rangle$。

（2）对第一个量子寄存器中的每一个量子比特执行 H 门变换，得到量子态

$$\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle|0^m\rangle$$

（3）对量子态 $\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle|0^m\rangle$ 作酉变换 U_f，得到量子态 $\frac{1}{\sqrt{2^n}}\sum_{i=0}^{2^n-1}|i\rangle|f(i)\rangle$。

（4）测量第二个量子寄存器，假设测量结果为 y，则原来的量子态发生坍缩，坍缩后的量子态为 $\frac{1}{\sqrt{2}}(|x\rangle|y\rangle+|x\oplus s\rangle|y\rangle)$，其中 $y=f(x)=f(x\oplus s)$。

（5）对第一个量子寄存器中的每个量子比特执行 H 门变换，得到量子态

$$\begin{aligned}H^{\otimes n}\left(\frac{1}{\sqrt{2}}|x\rangle+\frac{1}{\sqrt{2}}|x\oplus s\rangle\right)&=\frac{1}{\sqrt{2}}\sum_j\frac{(-1)^{x\cdot j}}{2^{n/2}}|j\rangle+\frac{1}{\sqrt{2}}\sum_j\frac{(-1)^{(x\oplus s)\cdot j}}{2^{n/2}}|j\rangle\\&=\sum_j\frac{(-1)^{x\cdot j}+(-1)^{(x\oplus s)\cdot j}}{2^{(n+1)/2}}|j\rangle\\&=\sum_j\frac{(1+(-1)^{s\cdot j})(-1)^{x\cdot j}}{2^{(n+1)/2}}|j\rangle\end{aligned}\tag{5.2.1}$$

其中，$x \cdot j = x_{n-1}j_{n-1} \oplus x_{n-2}j_{n-2} \oplus \cdots \oplus x_0 j_0$，$s \cdot j$ 的定义类似。如果 $s \cdot j = 0$，则量子态 $|j\rangle$ 的概率幅等于 $\frac{(-1)^{x \cdot j}}{2^{(n-1)/2}}$。反之，如果 $s \cdot j = 1$，则量子态 $|j\rangle$ 的概率幅等于 0。所以第一个量子寄存器中的量子态为 $\sum_{j'} |j'\rangle$，其中每一个 $|j'\rangle$ 均满足 $s \cdot j' = 0$。

（6）对第一个寄存器进行测量，测量的结果 j^1 一定满足 $s \cdot j^1 = 0$。

（7）重复步骤（1）～（6）$O(n)$ 次后，得到一组线性无关的向量组 $j^1, j^2, \cdots j^n$。建立二元域上的线性方程组，由于向量组线性无关，因此方程组的解存在，我们可以通过求解线性方程组得到 s。

下面简单分析 Simon 算法的计算复杂度。Simon 算法有两个量子寄存器，第一个量子寄存器的位数是 n，第二个量子寄存器的位数是 m，其中 $n \leqslant m$，因此量子算法的空间复杂度是 $O(m)$。对第一个量子寄存器的每一位量子比特做 H 门变换，然后对两个量子寄存器作一次酉变换，再对第一个量子寄存器的每一位量子比特作 H 门变换，之后再测量两个量子寄存器，因此查询复杂度是 $O(n)$，而且要重复计算 $O(n)$ 次，因此 Simon 算法的总查询复杂度是 $O(n^2)$。对第一个量子寄存器的每一个量子比特作 H 门变换可以同时进行，因此重复 $O(n)$ 次计算后的总时间复杂度是 $O(n)$，且算法的成功概率是 $O(1)$。Simon 算法的一个重要意义在于它启发了 Shor 算法。

5.3　Bernstein-Vazirani 算法

我们已经介绍过 Deutsch-Jozsa 量子算法，它仅需要运行一次就可以辨别给定函数是常数的还是平衡的，而经典算法则需要运行 $O(N = 2^n)$ 次才能实现该目的。Bernstein 和 Vazirani 运用 Deutsch-Jozsa 算法有效地解决了访问量子数据库的问题。问题具体可以描述为

给定黑箱可以计算的布尔函数 $f_s : \{0,1\}^n \to \{0,1\}$，即

$$f_s(x) = s \cdot x, \tag{5.3.1}$$

其中，$s \in \{0,1\}^n$ 是一个未知的向量，也称隐藏字符串，$s \cdot x = s_1 x_1 \oplus s_2 x_2 \oplus \cdots \oplus s_n x_n$。我们需要找到隐藏字符串 s 的具体值。

如果运行经典算法，则需要调用黑箱 n 次以确定 s 的 n 位数值，算法的查询复杂度为 $O(n)$。事实上，由 $f_s(10\cdots0) = s_1, f_s(01\cdots0) = s_2, \cdots, f_s(00\cdots1) = s_n$ 共 n 次黑箱调用可以确定 $s = (s_1, s_2, \cdots, s_n)$。然而运行 Bernstein-Vazirani 量子算法仅需要调用黑箱一次就可以决定 s。Bernstein-Vazirani 算法的线路图与图 5.4 类似，只需要将函数 f 替换为 f_s 即可。Bernstein-Vazirani 算法的步骤描述如下。

算法 5.4　Bernstein-Vazirani 算法

（1）两个寄存器的初始状态为 $|0^n\rangle|0\rangle$。

（2）对第二个寄存器的量子比特执行 X 门操作，系统状态转化为 $|0^n\rangle|1\rangle$。

（3）对两个寄存器的每个量子比特执行 H 门操作，系统状态转化为

$$\left(\frac{1}{2^{n/2}}\sum_x|x\rangle\right)\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)$$

（4）对两个寄存器的量子比特执行酉变换 U_f，得到量子态

$$\left(\frac{1}{2^{n/2}}\sum_x(-1)^{s\cdot x}|x\rangle\right)\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)$$

（5）对第一个寄存器执行 $H^{\otimes n}$ 操作，系统演化为

$$\begin{aligned}\left(\frac{1}{2^n}\sum_x(-1)^{s\cdot x}\sum_y(-1)^{x\cdot y}|y\rangle\right)\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)&=\left(\frac{1}{2^n}\sum_x\sum_y(-1)^{(s\oplus y)\cdot x}|y\rangle\right)\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)\\&=|s\rangle\left(\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right)\end{aligned}\tag{5.3.2}$$

（6）测量第一个寄存器将以 100% 的概率得到 s 的值。

5.4 QAOA 算法

量子近似优化算法（Quantum Approximate Optimization Algorithm，QAOA）是由 Farhi，Goldstone 和 Gutmann 于 2014 年提出的多项式时间算法，用来近似求解组合优化问题。一般来说，组合优化是从有限数量的对象中寻找使成本函数最小化的目标。组合优化在包括降低供应链成本、车辆路径、作业分配等实际问题中得到了广泛应用。现实中，很多组合优化问题都是 **NP**- 难的问题，不存在多项式时间的经典算法精确求解。往往在实际生活中，有些问题没有必要找到完美精确解，找到一个符合期望的近似解就足够了。于是遗传算法、退火算法和神经网络等近似优化算法（AOA）被提出用于多项式时间求解次优解，而量子优化算法（QAOA）的价值在于展示量子优越性。

QAOA 算法的一个重要应用就是最大切割问题（MAX-CUT）。给定一个由顶点 $i\in V$(顶点也常称为结点) 和边 $(i,j)\in E$ 构成的无向图 G，求解最大切割问题就是要得到两个子集 S_0 和 S_1，使得 $S_0\cup S_1=V$，$S_0\cap S_1=\varnothing$，且图 G 中 $i\in S_0$ 和 $j\in S_1$ 的边 (i,j) 的数量尽可能地多。简单来说就是将图中的顶点分成两组，使得连接这两组中顶点的边的数目最多。两顶点杠铃图如图 5.6 所示。

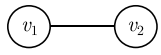

图 5.6 杠铃图

两顶点杠铃图共有 4 种方法二分顶点。我们把集合 S_0 或 S_1 中的顶点表示为 0 或 1，形成一个长度为 n 的比特串。4 种划分可以表示成 $\{00,01,10,11\}$，如图 5.7 所示。其中从左往右数第 1 个比特对应的结点为 v_1，第 2 比特对应的结点为 v_2，图中是以量子比特的形式展现的。只有存在连接不同集合中的顶点时才绘制边，中间的符号 X 表示需要

切割的边。于是，直观上可以看到杠铃图存在两种相同的最大切割，即图 5.7 中的第 2 个和第 3 个分组。更一般地，任意一个 n 顶点的图，其划分的情形总数是 2^n。对于顶点数目较少的情况，能够通过枚举法找到最大切割问题的最优解。然而，一旦顶点数目增加到足够大时，相应的计算时间复杂度也将呈指数级增加，就会形成 NP-难的问题。

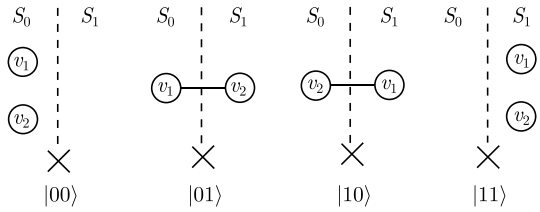

图 5.7　杠铃图的划分情况

接下来，我们将展示如何利用量子近似优化算法（QAOA）求解最大切割问题。出发点是将最大切割的比特串视为编码代价函数的哈密顿量的基态。首先需要引入哈密顿量，哈密顿量用来描述物理系统，以矩阵形式表示，该矩阵的特征值均为实数。哈密顿量的本征向量表示相应物理系统所处的各个本征态，而本征值表示相应本征态对应的能量。对于最大切割问题的哈密顿量形式，可以通过构造经典函数返回值确定。具体来说，如果图 G 中任意一条边所对应的两个结点跨越不同集合，那么返回 1（裁剪的边数或边的权重），否则返回 0。更严格的数学定义为

$$C_{ij} = \frac{1}{2}\left(1 - z_i z_j\right) \tag{5.4.1}$$

其中，如果顶点 i 属于集合 S_0，则 z_i 的值为 $+1$，否则 z_i 的值为 -1。于是，图 G 的最大切割值等于各条边切割贡献之和，即

$$\text{MAX-CUT} = \sum_{ij} C_{ij} \tag{5.4.2}$$

就杠铃图而言，存在 4 个分组，把它看成一个系统的 4 个孤立状态 $|00\rangle$、$|01\rangle$、$|10\rangle$、$|11\rangle$，每种划分对应的切割值可以看成是系统处于该状态时的能量。因此，杠铃图系统处于 $|00\rangle$ 态的能量为 0，处于 $|01\rangle$ 态的能量为 1，处于 $|10\rangle$ 态的能量为 1，处于 $|11\rangle$ 态的能量为 0。杠铃图的哈密顿量的矩阵形式表达为

$$H = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \frac{I - \sigma_1^z \sigma_2^z}{2} \tag{5.4.3}$$

其中，σ_1^z 表示对第 1 个量子比特位作用 Pauli Z 矩阵，对其余量子比特位作用单位矩阵，即 $\sigma_1^z = \sigma_z \otimes I$，类似地，$\sigma_2^z = I \otimes \sigma_z$。推广到更复杂的最大切割系统，由于其切割值为所包含的杠铃图系统切割值之和，所以对应于该复杂系统的哈密顿量表示为

$$H = \sum_{(i,j)\in E} \frac{I - \sigma_i^z \sigma_j^z}{2} \tag{5.4.4}$$