队伍编号	MC2303283
题号	A

基于 QUBO 模型与量子退火的信用评分卡组合优化问题求解

摘 要

本文根据金融信贷的实际问题，进行数学建模为整数规划模型，针对评分卡的组合优化信用问题，本文运用了量子退火等多种方法，充分发挥了物理学中量子理论在优化问题上的优势，根据题意构建 QUBO 模型，并借助 python 的 pyqubo、neal 等工具包编程求解.

问题一中，要求在 100 个信用评分卡中找出 1 张及其对应的阈值，使最终收入最多，针对该问题进行建模并将模型转为 **QUBO** 形式进行求解. 本文先对原问题进行了化简操作，根据题目建立更加常见的 0-1 规划模型，再通过一系列证明和程序验证，将模型进行 QUBO 建模，在拉格朗日乘数的选择上，我们通过多次程序验证给出了较优的结果，并且在接下来的问题求解中，我们都与传统的求解方法进行了正确性验证.

问题二中，要求给赛题数据中的信用评分卡 1、信用评分卡 2、信用评分卡 3 这三种规则设置其对应的阈值，使最终收入最多，针对该问题进行建模并将模型转为 **QUBO** 形式进行求解. 与第一问不同的是，这一问的通过率涉及到了累乘，我们提出了一些处理 0-1 变量累乘的方法.

问题三中，要求从所给附录中 100 个信用评分卡中任选 3 种信用评分卡并设置合理的阈值，使得最终收入最多，针对该问题进行建模并将模型转为 **QUBO** 形式进行求解. 这个问题与第二问类似，但是次数更高，并且在传统计算机中已经难以得出结论. 我们使用了量子退火进行求解，并与常见的模拟退火算法进行比较验证.

关键词：QUBO 模型；量子退火；信用评分卡；组合优化；整数规划模型；运筹学

目录

一、问题重述

1.1 问题背景

在银行信用卡或相关的贷款等业务中，对客户授信之前，需要先通过各种审核规则对客户的信用等级进行评定，通过评定后的客户才能获得信用或贷款资格．规则审核过程实际是经过一重或者多重组合规则后对客户进行打分，这些规则就被称为信用评分卡．

每个信用评分卡有多种阈值设置，但只有一个阈值生效，这就使得不同的信用评分卡在不同的阈值下，对应不同的通过率和坏账率，一般通过率越高，坏账率也会越高，反之，通过率越低，坏账率也越低．对银行来说，通过率越高，通过贷款资格审核的客户数量就越多，相应的银行获得的利息收入就会越多，但高通过率一般对应着高坏账率，而坏账意味着资金的损失风险．选择不同的信用评分卡，不同的阈值组合，会给银行带来不同的收入与损失，由此决定银行最终收入．由于银行场景的复杂性，往往需要采用选择多个不同的信用评分卡进行组合来实现最佳的风险控制策略．因此，银行的目标是选择最合理的信用评分卡组合以及其阈值，使得银行最终收入最多．

该问题所涉及的组合优化领域是优化领域中最重要的领域之一，在每个行业都有实际应用，也是运筹学、计算机科学和分析学最活跃的研究领域之一．传统方法是由分析师开发一种适合当前问题数学结构的求解算法，虽然这种算法在某些问题环境中产生了良好效果，但实践中出现的应用程序的多样性需要创建多种解决方案技术，每种技术在其最初预期用途之外的应用程序都很有限[1]．目前，利用量子计算原理解决优化问题的前景尤其光明[2][3][4][5][6]，而 QUBO 模型是量子计算的基础，将 QUBO 模型运行在量子计算机硬件上，可以进行毫秒级的加速求解，这种模型和加速方式在未来各行业中将得到广泛的实际应用．

1.2 问题提出

问题 1：在 100 个信用评分卡中找出 1 张及其对应阈值，使最终收入最多，针对该问题进行建模，将该模型转为 QUBO 形式并求解．

问题 2：假设赛题说明 3 目前已经选定了数据集中给出的信用评分卡 1、信用评分卡 2、信用评分卡 3 这三种规则，如何设置其对应的阈值，使最终收入最多，针对该问题进行建模，将模型转为 QUBO 形式并求解．

问题 3：从所给附录中 100 个信用评分卡中任选取 3 种信用评分卡，并设置合理的阈值，使得最终收入最多，针对该问题进行建模，并将模型转为 QUBO 形式并求解．

二、问题分析

数据分析：

对所给定的阈值数据集，通过简单的验证，可以总结出以下特征：

1. 数据无缺失值、异常值

2. 通过率和坏账率均随着阈值增大而增大

3. 通过率均为两位小数，坏账率均为三位小数

问题一中，可以进行数学建模，将通过率、坏账率都通过符号表示，并且表达出收入函数作为目标函数．根据约束条件：限制只能选一张卡一个阈值，可以建立整数规划模型，再将其转化为 QUBO 模型．

问题二中，我们需要选择三张银行卡，在实际问题中，银行对客户进行授信的场景非常复杂，往往受到多种因素的影响．根据题意，我们可以将授信流程进行简化．题设要求将坏账率设为三种阈值的坏账率平均，通过率设置为三种通过率相乘．我们可以据此构建新的模型．

问题三中，解空间相比前两问变大了许多，通过穷举已经难以得出结果．于是我们通过传统模拟退火来验证解的正确性，并通过量子退火求解问题．

三、模型假设

1. 假设银行在贷款流程中的贷款资金为定值 M=1000000，银行贷款利息收入为定值 r=0.08%，不受其他因素影响．

2. 假设信用评分卡有 100 种，每张卡有 10 种阈值，最多只能对一张卡选择设置一个阈值．

3. 假设最终收入仅因为信用卡阈值选择进行波动，不考虑其他意外得失的可能性．

四、符号说明

符号	说明
M	贷款资金，题设为 1000000 元
r	利息收入率，题设为 0.08
i	信用评分卡的编号，取值 1 到 100
j	阈值的编号，取值 1 到 10
x_{ij}	0 或 1，表示信用评分卡 i 阈值 j 是否被选择
x	x_{ij} 组成的列向量
t_{ij}	信用评分卡 i 选择 j 阈值时的通过率
h_{ij}	信用评分卡 i 选择 j 阈值时的坏账率
T	所有信用评分卡通过率组成的矩阵
H	所有信用评分卡通过率组成的矩阵
Q	QUBO 模型中的 Q 矩阵，为对称阵

五、问题一的模型建立与求解

5.1 模型准备

QUBO 模型：QUBO 模型：QUBO 模型即二次无约束二进制优化模型．QUBO 模型可以包含工业、科学和政府中发现的各种特殊的重要优化问题．正如 Kochenberger 等人（2014）和 Anthony 等人（2017）等研究所记录那样，通过易于应用的特殊重新配方技术，一旦将 QUBO 求解器放入 QUBO 框架中，就可以利用它们的力量有效地解决许多重要问题．QUBO 模型已经成为量子退火这一量子计算领域的基础，并已经成为神经形态计算的一个研究课题 [1].

QUBO 即二次无约束二值优化模型，形如 $max:x^TQx$．其中 x 是 0 和 1 组成的列向量，Q 是一个对称矩阵．QUBO 模型的基本特征是二值化，也就是说自变量只能取值 0 和 1，以及无约束条件，可以通过增加罚函数的方法，将传统规划模型的约束条件转化为无约束问题.

也可以将 QUBO 模型展开，形如：

$$max:\sum_{i=1}^{n}x_iw_i+\sum_{j=1}^{n}\sum_{i=1}^{n}x_jx_iw_iw_j$$

整数规划模型：在整数规划中，如果所有变量都限制为整数，则称为纯整数规划．如果仅一部分变量限制为整数，则称为混合整数规划．整数规划的一种特殊情形是 0-1 规划，它的变数仅限于 0 或 1．组合最优化通常都可表述为整数规划问题．两者都是在有限个可供选择的方案中，寻找满足一定约束的最好方案．有许多典型的问题反映整数规划的广泛背景．例如，背袋（或装载）问题、固定费用问题、和探险队问题（组合学的对集问题）、有效探险队问题（组合学的覆盖问题）、旅行推销员问题，车辆路径问题等．因此整数规划的应用范围也是极其广泛的．它不仅在工业和工程设计和科学研究方面有许多应用，而且在计算机设计、系统可靠性、编码和经济分析等方面也有新的应用．

5.2 基本整数规划模型的建立

选定的信用评分卡为 $i(1\le i\le 100)$，阈值设置为 $j(1\le j\le 10)$，h_{ij} 对应 i 卡 j 阈值坏账率，t_{ij} 对应 i 卡 j 阈值通过率，总坏账率 h，总通过率 t．T、H 代表通过率、坏账率组成的矩阵．贷款资金 M，利息收入 r，卡总数 $i_{max}=100$，阈值总数 $j_{max}=10$

根据题目假设，M=100000，r=0.08 为常量．有以下关系式：

收入：$Mrt(1-h)$ 损失：Mth

最终收入:

$$\begin{aligned}Mrt(1-h)-Mth&=Mrt-Mrth-Mth\\&=Mt(r-rh-h)\\&=Mt[r-(1+r)h]\\&=Mrt[1-\frac{1+r}{r}h]\end{aligned}$$

由题意，M、r 为常数．由于在问题中常数项取值对最大值的 i,j 不影响，可省略部分常数，将题目简化为：求收益函数 $f(t,h)=t(1-ah)$ 的最大值，其中 $a=\frac{1+r}{r}$ 为给定值．

针对问题 1，由符号假设，$t_{ij}\in T, h_{ij}\in H$. 因此原问题目标函数是关于 T,H,X 的函数．

定义函数决策变量 $x_{ij}\in\{1,0\}$，其中 x_{ij} 组成向量 x．

$g_{ij}=f_{ij}x_{ij}=f(t_{ij},h_{ij})x_{ij}$，目标函数 $G=\sum_{i=1}^{100}\sum_{j=1}^{10}g_{ij}$，

可针对原问题建立优化模型如下：

$$max: G(T,H,x)=\sum_{i=1}^{100}\sum_{j=1}^{10}x_{ij}f(t_{ij},h_{ij})$$

$$s.t.\begin{cases}\sum_{j=1}^{10}x_{ij}\le 1 & (1-1)\\ \sum_{i=1}^{100}\sum_{j=1}^{10}x_{ij}=1 & (1-2)\end{cases} \tag{1}$$

其中，

$$x_{ij}=\begin{cases}0, & \text{第 i 卡第 j 阈值未被选中}\\ 1, & \text{第 i 卡第 j 阈值被选中}\end{cases}$$

式 (1-1) 表示每个卡最多可取一个阈值，(1-2) 表示 100 张卡中有一个阈值被选择．显然，(1-2) 条件包含了 (1-1) 的情况，也就是说约束条件可以省略 (1-1).

5.3 QUBO 模型建立

因为 x_{ij} 是 0-1 变量，所以 $\forall i,j,$ 有 $x_{ij}^2=x_{ij}$.

$$g_{ij}=x_{ij}f(t_{ij},h_{ij})=x_{ij}^2f(t_{ij},h_{ij}),$$
$$G=\sum_{i=1}^{100}\sum_{j=1}^{10}x_{ij}^2f(t_{ij},h_{ij}) \qquad (1-3)$$

这是一个组合优化问题．可通过以下步骤，将目标函数转化为 QUBO 形式.

1. x_{ij} 组成 $100\times 10=1000$ 维的一元列向量 $x=(0,\ldots,1,\ldots,0)$，可将编号 i,j 进行降维处理至 $x_k(1\le k\le 1000)$ 即只使用一个值对 x 的约束，相关的转化方式为：

$$i=\lceil\frac{k}{10}\rceil$$

$$\begin{cases}j=1 & \cdots\text{k 是 10 的倍数}\\ j=k \mod 10 & \cdots\text{k 不是 10 的倍数}\end{cases} \tag{2}$$

2. 通过查阅文献知道，根据约束条件可定义惩罚函数，将问题转化为无约束优化.

$$penalty = -P(\sum_{k=1}^{1000} x_k - 1)^2$$

其中，P 为一个相对目标函数可能取值的一个极大的数，若不满足约束条件，惩罚函数会极大地减小函数值，避免此解被选中为最优解，因此对目标函数起到约束作用.

3. 矩阵 $Q = (f_k)_{1000} = (f_{ij})_{1000}$，由于 (1-3) 只包含 x_{ij} 相关的平方项，因此 Q 矩阵是一个 1000×1000 的对角阵，$Q_k = f(t_k, h_k) = t_k(1 - ah_k)$.

综上所述，可对原优化问题进行初步 QUBO 建模：

$$max : x^T Q x - P(\sum_{i=1}^{1000} x_k - 1)^2$$
$$= \sum_{i=1}^{1000} t_i(1 - ah_i)x_i - P(\sum_{i=1}^{1000} x_k - 1)^2$$

$Q = diag(t_1(1 - ah_1), t_2(1 - ah_2), \ldots, t_{1000}(1 - ah_{1000}))$.

4. 罚系数 P 的选取．P 也可以看做是拉格朗日乘数，是一个自由度较高可以自己定义的常数，P 值的选取是否合理将直接影响到优化问题是否满足约束条件．在此题中要保证选择唯一阈值时目标函数最大，选择 2、3，或更多时结果更小，因此需要满足：在其他多种阈值被选择的情况下，$x^T Q x$ 的作用再大也不大过唯一阈值的情况．这个问题可以通过数学的方式去求解，在这里我们通过多次试验的方式来确定 P 值的合适范围.

通常来说，P 值越大对应的约束条件越强，但可能因此影响结果无法收敛到最优情况.

5.4 求解与验证

由于此问题中仅存在 1000 个可能的变量，可以通过传统计算机进行循环穷举求解.结果为：

信用评分卡号	阈值选择	收入
49	1	61172.0

借助 pyqubo 的工具包可以直接求解 QUBO 模型，编写的代码详见附录 1．需要注意的是，编码时以计算机习惯从 0 开始计数，所以得到的 k 值结果与上述 (2) 式相差 1，计算的结果是 x[480], 也就是 k=481，即 i=49，j=1，与穷举法结果一致.

在此问题中，经过多次试验，我们将拉格朗日乘数选在个位数这个数量级．因为启发式算法每次运行结果可能不同，会在接近最优解处局部收敛，我们通过多次的程序运行，总结了程序得到的结果与惩罚函数的关系，并在此基础上对模型进行改进.

当 P=5 时，12 次运行所得的解集为 $\{(720,58.51519),(201,57.2276),(480,61.1719),(480,61.17199),(481,61.02159),(431,55.5212),(480,61.171),(480,61.171),(130,55.14),(480,61.17199),(250,56.0),(481,61.1719)\}$ 可见其实能正确得到精确结果的情况较少，多数是在接近最优的点附近.

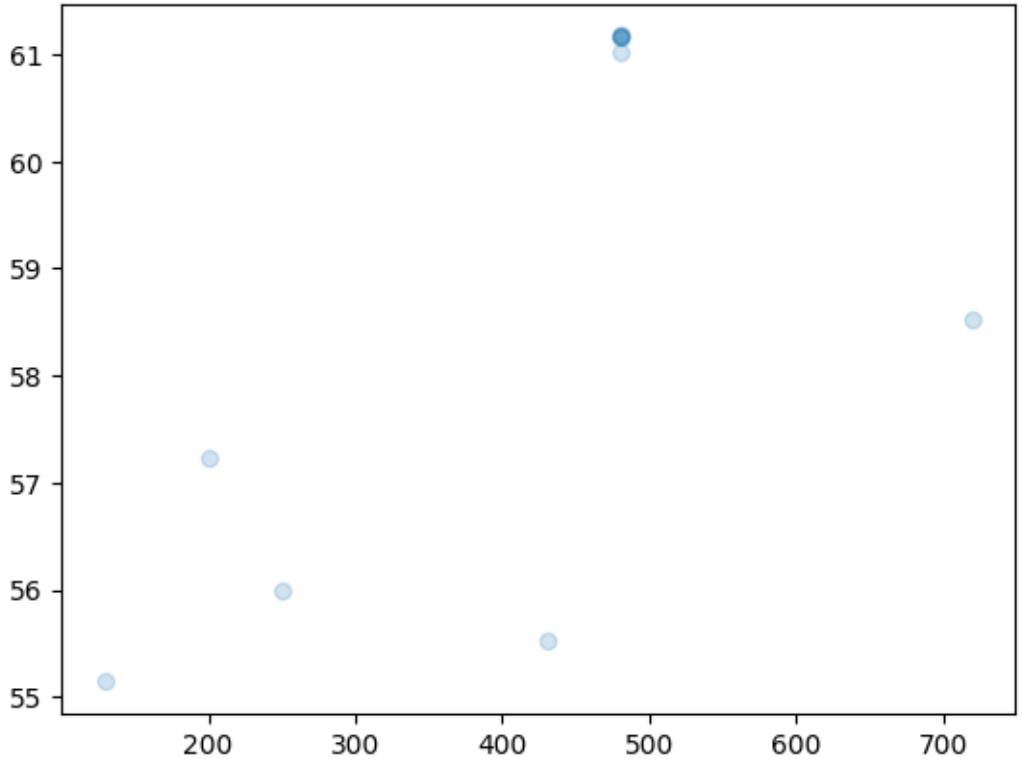

图 1 算法多次运行结果

六、问题二的模型建立与求解

6.1 规划模型建立

与问题一不同的是，这一问需要选择三个信用卡的阈值组合．根据题意，对选中的 i 阈值，通过率 $t = \prod_{i=1}^{3} t_i$, 坏账率 $h = \frac{1}{3}\sum_{i=1}^{3} h_i$,

沿用问题一推导得到的收入函数 $f(t,h) = t(1-ah)$，决策变量 x_{ij}，可以初步建立以下规划模型：

$$max : f(t,h)$$

$$s.t.\begin{cases}\sum_{j=1}^{10} x_{1j}=1\\ \sum_{j=2}^{10} x_{2j}=1\\ \sum_{j=3}^{10} x_{3j}=1\end{cases} \tag{3}$$

(3) 式中约束条件代表信用评分卡 1、2、3 分别有且仅可选择 1 个阈值．但是，值得注意的是，由于决策变量 x_{ij} 可能取值为 0，所以我们不能直接像处理累加那样，将 $x_{ij}t_{ij}$ 直接相乘，这样的结果会恒为 0，这也是这个问题最为棘手的一点．经过一系列的思考，我们提出了一些处理累乘的建模思路．

6.2 累乘的建模处理

6.2.1 转化为指数形式

1. 对 T 矩阵各个值取对数，对下标 i，j 降维至 k 后，T 为 30 维对角矩阵．$T = diag(t_1, t_2, \ldots, t_{30})$, 得到的 $T' = diag(ln(t_1), ln(t_2), \ldots, ln(t_{30}))$

2. $x^T T' = (\ldots, ln(t_{k_1}), \ldots, ln(t_{k_2}), \ldots, ln(t_{k_3}) \ldots))$
   $x^T T x = ln(t_{k_1}) + ln(t_{k_2}) + ln(t_{k_3})$，其中 k_1, k_2, k_3 是选择的阈值．此时我们已经可以得到乘积，因为 $ln(t_{k_1}) + ln(t_{k_2}) + ln(t_{k_3}) = ln(t_{k_1} t_{k_2} t_{k_3})$, 只需要再将其转为指数幂函数即可．

3. 对上述式子求指数，我们可以得到 $t = e^{x^T T' x} = t_{k_1} t_{k_2} t_{k_3}$

目标函数可以表示为 $max: e^{x^T T' x}(1 - \frac{a}{3} x^T H x) = e^{\sum_{i=1}^{30} ln(t_i) x_i}(1 - \frac{a}{3} \sum_{i=1}^{30} h_i x_i)$

此时只存在累加，但是涉及到指数运算，可能仍然难以转化为 QUBO 模型．此方法若要进行求解，更加适用于整个目标函数只存在累乘的情况．

6.2.2 用一种简单加和表示累乘的决策变量

定义决策变量组成的函数因子 $(1 - x_i + t_i x_i)$, 观察到，若 $x_i = 0$，函数因子取值为 1，若 $x_i = 1$，函数因子取值为 t_i. 这个方法非常巧妙地使可以用 k 从 1-30 的累乘表达 $t_{k_1} t_{k_2} t_{k_3}$.

即：通过率 $t = \prod_{i=1}^{30}(1 - x_i + t_i x_i)$, 目标函数可以表示为：max：$\prod_{i=1}^{30}(1 - x_i + t_i x_i)(1 - \frac{a}{3} \sum_{i=1}^{30} h_i x_i)$

6.2.3 三阶降维

由于这个问题涉及的因子较少，可以尝试将问题变为三次的，再降维到二次．目标函数可表达为：$max: \sum_{j_1}^{10} \sum_{j_2}^{10} \sum_{j_3}^{10} t_{1j_1} t_{2j_2} t_{3j_3}(1 - \frac{a}{3} \sum_{i=1}^{30} h_i)$ 常见的降维方法如：将 $t_{1j_1} t_{2j_2}$ 乘积表示为一个新的变量 t'.

6.3 建模转换

我们使用上述方法 2 的技巧，可以建立模型

$$max: t = \prod_{i=1}^{30}(1 - x_i + t_i x_i)(1 - \frac{a}{3}\sum_{i=1}^{30} h_i x_i)$$

$$s.t. \begin{cases} \sum_{j=1}^{10} x_{1j} = 1 \\ \sum_{j=2}^{10} x_{2j} = 1 \\ \sum_{j=3}^{10} x_{3j} = 1 \end{cases} \tag{4}$$

为了建立 QUBO 模型，我们可以先定义罚函数，同样根据第一问的规则进行了下标降维：

$$penalty = -P(\sum_{j=1}^{10} x_j - 1)^2 - P(\sum_{j=11}^{20} x_j - 1)^2 - P(\sum_{j=21}^{30} x_j - 1)^2$$

6.4 求解与验证

此问题中可能的情况与上一问相同，也是 $10 \times 10 \times 10 = 1000$ 种可能的组合，因此也可以通过传统计算机进行循环穷举求解.

结果为：

信用评分卡号	阈值选择	总收入
1	8	27914.8
2	1	
3	2	

七、问题三的模型建立与求解

7.1 模型建立

问题三其实和第二问类似，对于我们在上一问建立的模型，有着比较好的一般性，可以直接沿用.

$$max: t = \prod_{i=1}^{1000}(1 - x_i + t_i x_i)(1 - \frac{a}{3}\sum_{i=1}^{1000} h_i x_i)$$

$$s.t.\begin{cases}\sum_{j=1}^{10} x_{ij} \le 1 \\ \sum_{i=1}^{100}\sum_{j=1}^{10} x_{ij} = 3\end{cases} \tag{5}$$

在 QUBO 模型建立中，罚函数定义为：

$$penalty = -P(\sum_{j=1}^{1000} x_j - 3)^2 - P(\sum_{j=1}^{10} x_j + y_j - 1)^2$$

其中，y_j 为引入的松弛变量，也是 0-1 变量.

7.1.1 使用传统模拟退火求解

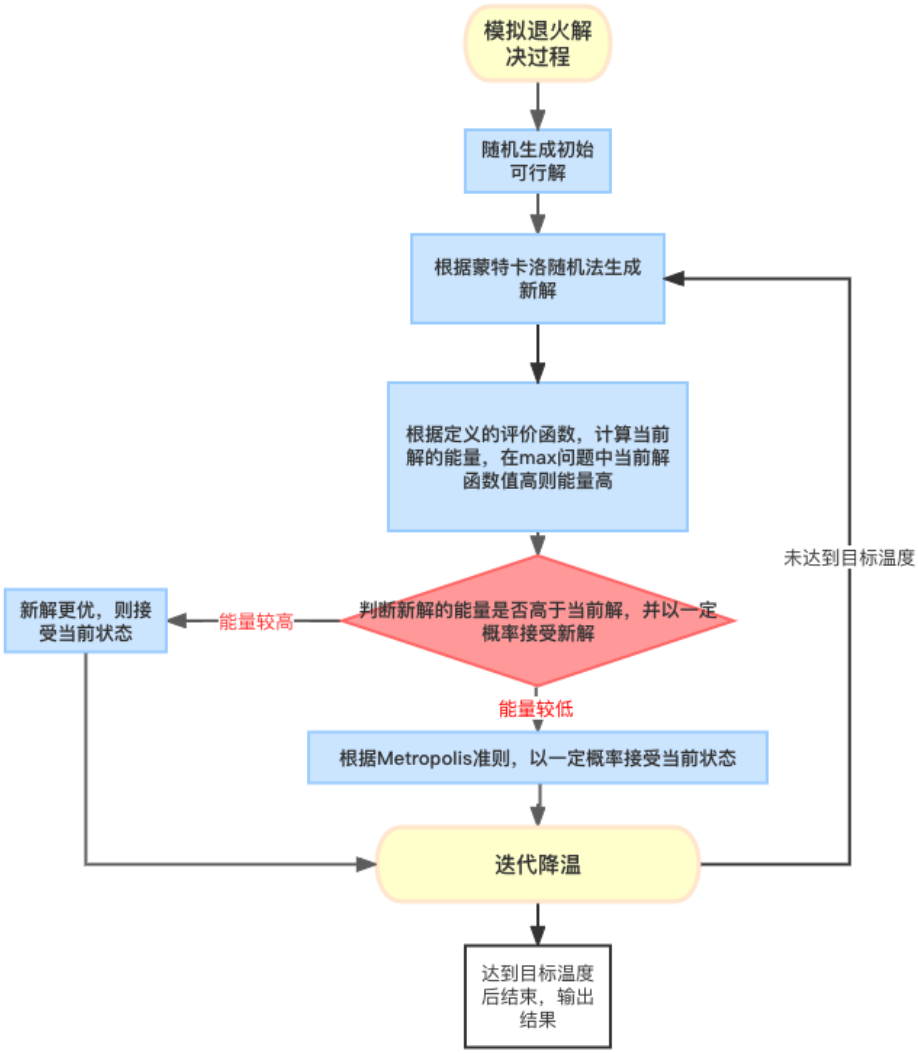

图 2 传统模拟退火流程图

7.1.2 使用量子退火求解

量子退火：量子计算促进了量子力学原理在计算中的应用，从 1982 年起量子计算获得了很大的发展势头．目前存在许多量子算法，它们比著名的经典算法运行得更有效．绝热量子计算是由 Farhi 等人在 21 世纪初提出的．它基于量子力学的一个重要定理，绝热定理．这个定理断言，如果量子系统是由一个从 Hinit 进化到 Hfin 的缓慢变化的哈密顿量驱动的，那么如果系统从 Hinit 的基态开始，系统将最终处于 Hfin 的基态．已经证明绝热量子计算等同于标准量子计算．量子退火基于量子绝热计算范式，被用于解决组合优化问题．量子退火器试图解决问题的方式与使用经典模拟退火解决优化问题的方式非常相似．通过多元函数构建能源景观，使得基态对应于问题的解决方案．量子过程被迭代，直到以足够高的概率找到最优解．“量子退火器”中的“量子”是指使用多量子位隧道．高度并行性是量子退火相对于经典代码执行的优势．量子退火器并行探索所有可能的输入，以找到问题的最佳解决方案 [7]．

八、模型评价与改进

在这几天的研究中，我们发现 QUBO 模型和量子退火都是一项很新颖的技术，在解决 NP-hard 的组合优化问题上大放异彩，这无疑是非常令人振奋的．与此同时，也说明这项技术存在着极大的可优化空间，就现在来说可解决的问题还有一定的局限性．我们总结了一些建模上发现的难点．

1. 如何选择决策变量，QUBO 和 ising 等模型的特点是决策变量只能取二值变量．
2. 对大规模问题计算速度会较慢，比如问题一的 1000 维矩阵，实际上计算速度和结果与传统计算机相比是没有优势的．
3. 如何对目标函数进行 QUBO 建模，这也是本次数学建模问题中最难最重要的一点，很多时候目标问题并不能直接转化为 QUBO 所要求的二次函数．

针对我们建立的模型，我们认为还存在非常多的不足，第一问中散点图可以看出来收敛效果并不好，很难得到最优解，原因是建立的矩阵维数过高，第三问也是如此．我们建立的模型已经超出了题目的需要，题目只需要三个数相乘，而我们的目标函数也可以表示为更多的数相乘，因此我们认为这并不代表着最优．

参考文献

[1] Fred Glover, Gary Kochenberger, Yu Du. Quantum Bridge Analytics I: A Tutorial on Formulating and Using QUBO Models[J]. 4OR-Q J Oper Res, 17:335-371, 2019.

[2] Perdomo-Ortiz A, Dickson N, Drew-Brook M, Rose G, Aspuru-Guzik A. Finding low-energy conformations of lattice protein models by quantum annealing[J]. Scinetific Report, 2, 571, 2012.

[3] Sarkar A.Quantum Algorithms: For Pattern-Matching in Genomic Sequences[D]. Technische Universiteit Delft, 2018.

[4] Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N, Lloyd S.Quantum machine learning[J]. Nature, 549, 195, 2017.

[5] Papalitsas C, Karakostas P, Andronikos T, Sioutas S, Giannakis K. Combinatorial GVNS(General Variable Neighborhood Search) Optimization for Dynamic Garbage Collection[J]. Algorithms, 11, 38, 2018.

[6] Rebentrost P, Schuld M, Wossnig L, Petruccione F, Lloyd S. Quantum gradient descent and Newton’s method for constrained polynomial optimization[J]. NewJ.Phys, 21, 073023, 2019.

附　录

附录 1：源代码

问题一：

```
import numpy as np
from pyqubo import Array, Constraint, solve_qubo

# 定义二进制变量 x
x = Array.create("x", shape=1000, vartype="BINARY")

# 定义原始目标函数
f = sum(t[i] * (10 - 135 * h[i]) * x[i] for i in range(1000))

# 定义约束条件
constraint = Constraint((sum(x[i] for i in range(1000)) - 1) ** 2, label="
    constraint")
# 定义拉格朗日乘数
lagrange_multiplier = 10000
# 定义带有约束的目标函数
g = -f + lagrange_multiplier * constraint
# 编译目标函数，生成QUBO
compiled_qubo = g.compile()
# 解决QUBO问题
qubo, offset = compiled_qubo.to_qubo()
solution = solve_qubo(qubo)
# 创建SA求解器
sa = neal.SimulatedAnnealingSampler()
# 采样
sampleset = sa.sample_qubo(qubo)
# 解码
decoded_samples = model.decode_sampleset(sampleset)
best_sample = min(decoded_samples, key=lambda x: x.energy)
# 输出最优解
best_sample.sample
print("解决方案:",solution)

#在结果字典中寻找value为1的键值对
def get_key_by_value(d, target_value):
    for key, value in d.items():
        if value == target_value:
            return key
    return "Value not found in the dictionary"
get_key_by_value(best_sample.sample,1)
```