Diseño

19 de febrero de 2024

1. Problemas de optimización combinatoria

Un problema de optimización combinatoria se define con la siguiente función de coste:

$$f(x) = \sum_{\alpha=1}^{m} f_\alpha(x)$$

$$\text{dde } x = x_1 x_2 ... x_n \text{ y } x_i \in \{0, 1\}$$

$$f_\alpha(x) = \begin{cases} 1 \text{ si x satisface } f_\alpha \\ 0 \text{ en otro caso} \end{cases}$$

En este caso el objetivo será encontrar el **mínimo global** de esta función. Para esto mismo se debe encontrar una cadena de bits de tamaño n que cumplan para la mayor cantidad de cláusulas $f_\alpha(x) = 0$.

Para que sean aplicados a algoritmos cuánticos estos problemas no pueden tener restricciones añadidas, por lo que el espacio de resultados posibles será de 2^n combinaciones.

1.1. Añadir restricciones

Si el problema que se está intentando representar tiene restricciones de la forma $A(x) = B(x)$ la forma de añadirlas a la función de coste sería $f'(x) = f(x) + P * (A(x) - B(x))^2$, donde

$$P * (A(x) - B(x))^2 \begin{cases} = 0 \text{ si se cumple la restricción} \\ \geq P \text{ en otro caso} \end{cases}$$

El parámetro P se denomina *modificador de Lagrange* y tiene un valor lo suficientemente grande como para que el castigo en caso de romper una restricción aumente lo suficiente el valor de la función de coste como para que sea mayor que cualquier otro resultado en el que no se rompa. Esto sería: $P > \text{máx}_x f(x)$

2. Circuito de QAOA

El sistema cuántico en el algoritmo se desarrolla sobre un espacio de Hilbert de 2^n dimensiones, donde n es el número de bits de entrada en la función de coste clásica. Esto quiere decir que se tendrán tantos qubits como bits tenga la entrada de *f(x)*.

La base computacional se representa como $\{|x\rangle : x \in \{0,1\}^n\}$.

La idea general de QAOA se basa en preparar un estado $|\psi(\vec{\beta},\vec{\gamma})\rangle$ tal que, con los valores adecuados $(\vec{\beta_{opt}},\vec{\gamma_{opt}})$, el estado $|\psi(\vec{\beta_{opt}},\vec{\gamma_{opt}})\rangle$ encuentre la solución al problema. Los parámetros de dicho estado son:

$$\vec{\beta} = [\beta_0, \beta_1, ..., \beta_{p-1}]$$
$$\vec{\gamma} = [\gamma_0, \gamma_1, ..., \gamma_{p-1}]$$

Donde p es el número de capas del circuito y $\beta, \gamma \in [0, 2\pi]$.

Este estado consta de tres componentes: el **estado inicial** ($|\psi_0\rangle$); y dos operadores denominados **problem hamiltonian** ($U(C,\gamma)$) y **mixing hamiltonian** ($U(B,\beta)$). Estos se combinan de la siguiente forma:

$$|\psi(\vec{\beta},\vec{\gamma})\rangle = U(B,\beta_{p-1})U(C,\gamma_{p-1})U(B,\beta_{p-2})U(C,\gamma_{p-2})...U(B,\beta_0)U(C,\gamma_0)\,|\psi_0\rangle$$

2.1. Estado inicial

El estado inicial del qubit se define como

$$|\psi_0\rangle = \frac{1}{\sqrt{2^n}}\sum_x |x\rangle = (\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle))^{\otimes n} = H^{\otimes n}\,|0\rangle^{\otimes n}$$

Este estado inicial se construye añadiendo operadores de Hadamard a n qubits inicializados a $|0\rangle$, lo que genera un estado equiprobable, es decir, donde la probabilidad de obtener una cadena dada de n bits al medir el estado sería en cualquier caso $\frac{1}{2^n}$.

2.2. Problem hamiltonian y mixing hamiltonian

Los operadores se definen como:

$$U(B,\beta) = e^{-i\beta B}$$
$$U(C,\gamma) = e^{-i\gamma C}$$

La operación $U(X) = e^{-iX}$ sirve para garantizar la unitariedad del operador ya que, como se verá en su definición, B y C no son necesariamente unitarios [1].

[1] En computación cuántica los operadores lineales deben ser unitarios. Tiene relación con la idea $\sum_{|x\rangle} P(|x\rangle) = 1$, donde $|x\rangle$ itera sobre los vectores de la base computacional del sistema y $P(|x\rangle)$ se refiere a la probabilidad de medir i sobre dicha base.

Tanto la noción de hamiltoniano como la construcción de un operador unitario proceden de la ecuación de Schrödinger de la mecánica cuántica.

2.2.1. Operador *B*

B se construye a partir de puertas Pauli-X (σ^x) y se define de la siguiente forma:

$$B = \sum_{i=0}^{n-1} \sigma_i^x$$

$$U(B,\beta) = e^{-i\beta B} = \prod_{j=0}^{n-1} e^{-i\beta\sigma_j^x} = \prod_{j=0}^{n-1} Rx_i(2*\beta)$$

Como se puede ver únicamente depende de la cantidad de qubits del sistema.

2.2.2. Operador *C*

C debe ser un operador diagonal en la base computacional $|x\rangle$ con los valores de la función objetivo. De esta forma, siendo $|x\rangle$ un vector en la base computacional se cumple que $\forall\, |x\rangle = [x_1x_2...x_n], \langle x|\, C\, |x\rangle = f(x)$. Se define a partir de la función de coste clásica del problema a resolver y se construye utilizando puertas Pauli-Z (σ^z).

Para obtener este operador se parte de una función de la forma $f(x)$, donde $x = x_1x_2...x_n$ y $x_i \in 0, 1$ (como la definida en la sección 1). Para construirlo la función utilizada debe tener como entrada variables con valor $\{-1, 1\}$, en lugar de $\{0, 1\}$, para lo que se debe realizar un cambio de variable:

$$f(x) \to C(z)$$
$$x_i \to \frac{1 - z_i}{2}$$

De esta forma $z_i = \begin{cases} 1 \text{ si } x_i = 0 \\ -1 \text{ si } x_i = 1 \end{cases}$. La forma de pasar de la función *C(z)* al operador C es sustituyendo las variables z_i por puertas σ_i^z.

3. Algoritmo de QAOA

La estructura general del algoritmo es la siguiente:

1. Inicializar $\vec{\beta}$ y $\vec{\gamma}$ a valores arbitrarios.
2. Repetir hasta que se cumpla cierto criterio de convergencia, obteniendo $\vec{\beta_{opt}}, \vec{\gamma_{opt}}$
   2.1. Construir el circuito correspondiente al estado $|\psi(\vec{\beta}, \vec{\gamma})\rangle$

2.2. Medir el estado en la base computacional, lo que devuelve una cadena de bits $\{0,1\}^n$

2.3. Computar el valor esperado de C: $\langle\psi(\vec{\beta},\vec{\gamma})|C|\psi(\vec{\beta},\vec{\gamma})\rangle$. Esto puede ser calculado de manera equivalente evaluando la cadena de bits, obtenida de medir el estado, con la función de coste clásica *f(x)*.

2.4. Hallar nuevos $\vec{\beta_{new}}, \vec{\gamma_{new}}$ usando un algoritmo clásico de optimización y volver al paso 2..

3. Calcular el valor esperado: $\langle\psi(\vec{\beta_{opt}},\vec{\gamma_{opt}})|C|\psi(\vec{\beta_{opt}},\vec{\gamma_{opt}})\rangle$. Este valor aproximará $\text{mín}_x\, f(x)$

Para el estado construido se cumple que para un número de capas $p \to \infty$:

$$\min_{\beta,\gamma}\langle\psi(\vec{\beta},\vec{\gamma})|C|\psi(\vec{\beta},\vec{\gamma})\rangle = \min_{x} f(x)$$

Esto es equivalente a decir que el estado fundamental del sistema corresponde al valor mínimo de la función de coste.

De esta forma, el optimizador clásico llamará en repetidas iteraciones a una función *execute_circuit* con una entrada $(\vec{\beta},\vec{\gamma})$ que devolverá como salida un valor mayor o igual al mínimo de la función de coste. Este resultado es el que el optimizador debe minimizar.

En este caso el optimizador clásico que se utilizará es COBYLA (Constrained Optimization BY Linear Approximation), de la librería de *Python scipy.minimize*.