D-Wave: computación cuántica mediante *quantum annealing*

Procesamiento de Datos a Gran Escala - Práctica opcional

José Antonio Álvarez Ocete
Francisco Javier Sáez Maldonado

3 de diciembre de 2021

Índice

1. Motivación y objetivos

En esta asignatura hemos expuesto las bases teóricas de la computación cuántica desde el álgebra lineal y la teoría de la computación, para posteriormente explicar los algoritmos más destacados de este nuevo paradigma, como la teleportación cuántica o el algoritmo de Shor.

En esta práctica opcional exploraremos otra forma de implementar computación cuántica, basándonos en evolución adiabática y *quantum annealing* (usualmente mal traducido como *enfriamiento cuántico*). En particular exploraremos los siguientes temas:

- El *quantum annealing* comparte algunas de las bases con la computación cuántica basada en circuitos, describiremos de forma superficial e intuitiva **las bases de la mecánica cuántica** que subyacen a ambos modelos y describiremos dónde residen las diferencias entre ambos.
- Explicaremos en profundidad **óomo funcionan los ordenadores D-Wave**, así como los problemas de optimización que pueden resolver.
- Comentaremos en detalle **el problema de ensamblaje del genoma**, y cómo resolverlo utilizando ordenadores D-Wave.

- Veremos los resultados obtenidos los computadores *D-Wave 2000* y *Advantage* al resolver este problema particular.

Este problema esta basado en el Trabajo Fin de Grado de uno de los componentes de esta pareja de prácticas, José Antonio Álvarez. Dicho documento puede encontrarse en el respectivo repositorio de GitHub.

2. Desarrollo

2.1. El modelo de la mecánica cuántica

Realizaremos una descripción superficial sobre la gran base teórica existente en este respecto. Para más detalle, consúltese el TFG al completo.

La mecánica cuántica esta basado en cuatro "simples" postulados, sobre los que se construye toda la teoría necesaria para la física cuántica y la computación cuántica en particular. Los enunciamos y explicamos a continuación.

1. El primer postulado nos indica **cómo describir matemáticamente un sistema físico**: *Cualquier sistema físico aislado tiene asociado un espacio de Hilbert llamado el* ***espacio de estados****. El estado del sistema queda totalmente descrito por un vector unitario de dicho espacio, y se denomina* ***vector estado****.*

   Dado que todo espacio de Hilbert finito es isomorfo a $\mathbb{C}^n$, donde n es la dimensión de dicho espacio, los vectores estados serán vectores unitarios de $\mathbb{C}^n$ para cierto n. En particular, un *qubit* se define como un vector estado del espacio de estados $\mathbb{C}^2$. Es decir, un vector unitario de $\mathbb{C}^2$:

   $$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle$$

   Donde $|\alpha|, |\beta| = 1$.

2. El segundo postulado nos indica **cómo medir nuestro sistema** en este paradigma. De igual forma que medimos la velocidad de un coche o la altura de una persona, hemos de ser capaces de medir ciertas variables en este sistema. El enunciado formal implica unos operadores de medida que únicamente complicarían este texto, así que nos ceñimos a las implicaciones del mismo.

   La consecuencia más relevante es que **al medir un sistema, lo alteramos**. Esto se traduce en la computación cuántica en el colapso de los qubits tras la medida a uno de los estados. Adicionalmente, esto establece uno de los principales inconvenientes de la computación cúantica: la imposibilidad de conocer exactamente las amplitudes de un estado, lo que desembocará en el **Teorema de no clonación**.

3. El tercer postulado es el más relevante para nosotros pues **crea la distinción entre el modelo de computación cuántica basado en circuitos y el basado en *quantum annealing***.

Describe la evolución del sistema y tiene dos enunciados equivalentes que provocan la distinción mencionada.

En su primera forma, el postulado dicta que la evolución de un sistema cuántico cerrado queda descrita por una **transformación unitaria** que unicamente depende de los tiempos t_1, t_2 en los que se compara el sistema:

$$|\varphi_{t_2}\rangle = U|\varphi_{t_1}\rangle$$

En el modelo de circuitos de la computación cuántica este postulado establece las bases teóricas de las puertas cúanticas: transformaciones unitarias aplicadas al estado del sistema (qubits).

La segunda forma de este postulado viene de la mano de la ecuación de Schrödinger, la cual exploraremos en detalle más adelante.

4. Finalmente, el cuarto postulado nos habla de **la composición de sistemas cuánticos**. Dicta lo siguiente: *El espacio de estados de un sistema físico compuesto es el* ***producto tensorial*** *de los espacios de estados de los subsistemas. Además, el vector de estado del sistema es el producto tensorial de los vectores de estado de los subsistemas:* $|\varphi_1\rangle \otimes \ldots \otimes |\varphi_n\rangle$.

   La diferencia más relevante en nuestro caso entre el producto vectorial al que estamos acostumbrados y el producto tensorial es que la dimensión de los espacio se multiplica en vez de sumarse: $\mathbb{R}^2 \times \mathbb{R}^3 = \mathbb{R}^5$ mientras que $\mathbb{R}^2 \otimes \mathbb{R}^3 = \mathbb{R}^6$. Es por ello que al añadir qubits a nuestro sistema, la información crece de manera exponencial y no lineal como ocurre en el caso de los bits.

   Como corolario inmediato de este postulado obtenemos que un sistema de n qubits tiene por espacio de estados asociado $\mathbb{C}^{2^n}$.

2.2. Quantum annealing y evolución adiabática

Como se menciono anteriormente, Schrödinger proporcionó una versión equivalente de este postulado. Esta dicta que la evolución temporal de nuestro sistema viene dada por la famosa **ecuación de Schrödinger**:

$$i\hbar\frac{d|\varphi\rangle}{dt} = H|\varphi\rangle$$

Donde $\hbar$ es la constante de Plank, i es la unidad imaginaria y H es un operador hamiltoniano. Aunque esta ecuación es intimidante, es más sencilla de lo que parece. Puesto que $\hbar$ e i son constantes, lo único desconocido es el operador H (una matriz). Con ello conocemos la derivada del vector de estados. Esto es, cómo evoluciona con el tiempo.

La idea que reside bajo el *quantum annealing* es elegir nosotros el hamiltoniano del sistema para hacer que evolucione a nuestro antojo. Utilizaremos la descomposición singular del hamiltoniano:

$$H = \sum_{E} E|v_E\rangle\langle v_E|$$

Donde E son los valores propios de H y v_E los respectivos vectores propios. A los valores propios se les suele denominar *energías*, ya que describen la energía del estado propio asociado. Al estado propio con menor energía se le denomina *estado base* o *ground state*.

En particular, utilizaremos un hamiltoniano que variaremos con el tiempo:

$$H(t) = \sum_{E(t)} E(t)|v_{E(t)}\rangle\langle v_{E(t)}| \quad t \in [t_{initial}, t_{final}]$$

Aqui nos apoyamos del **Teorema de Evolución Adiabática**: un sistema físico se mantiene en un estado propios si las alteraciones realizadas al sistema se producen lo suficientemente despacio y hay un *hueco* lo suficientemente grande entre el valor propio asociado y el resto de valores propios.

En primer lugar, demos una visión intuitiva de este teorema. Supongamos que tenemos un péndulo simple oscilando sujeto en su punto superior con la mano. Si movemos la mano muy despacio, la perturbación que sufre el péndulo es inócua, y este permanecerá en su estado propio. Sin embargo, si movémos rápidamente la mano, el péndulo variará drásticamente su trayectoria.

Para nuestro caso, buscaremos que el hamiltoniano varíe entre dos valores constantes: $H_{initial}$ y H_{final}:

$$H(s) = A(s) \cdot H_{initial} + B(s) \cdot H_{final} \quad \forall s \in [t_{initial}, t_{final}]$$

Donde $A(s)$ disminuirá con el tiempo y $B(s)$ aumentará.

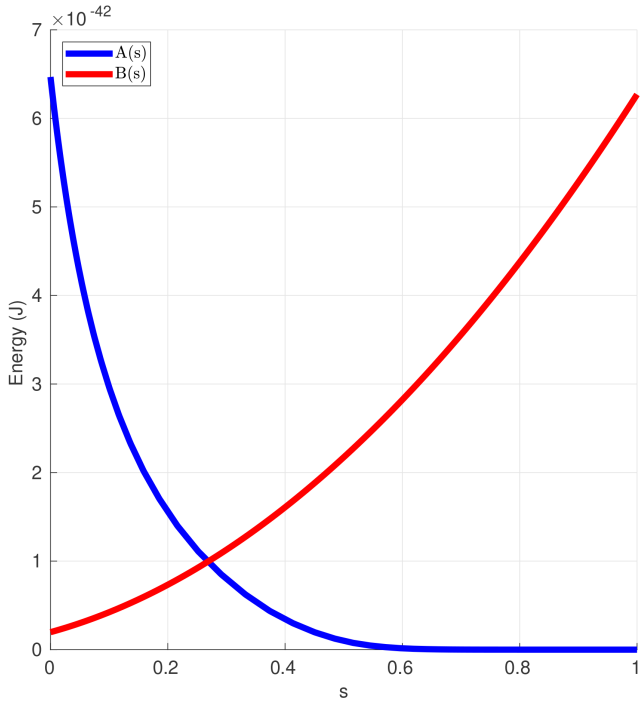


Figura 1: Funciones de *Annealing* utilizadas en los sistemas D-Wave 2X

La codificación inicial, $H_{initial}$, será muy sencilla. De esta forma podremos conocer nuestro sistema a la perfección. De hecho, nuestro estado inicial del sistema será el estado base de $H_{initial}$. Si se cumple el teorema de evolución adiabática (volveremos ahora para ver cuando ocurre), tras la evolución nuestro sistema estará en el estado base de H_{final}.

Dado un problema de optimización, codificaremos H_{final} para que el estado propio de menor energía sea el mínimo de nuestra función objetivo. De esta forma, tras aplicar evolución adiabática obtendremos el estado base. Esto es, la solución que minimiza nuestra función objetivo.

La ecuación final utilizada en la práctica por los sistemas D'Wave es la siguiente:

$$H(s) = \underbrace{-\frac{A(s)}{2}\left(\sum_i \sigma_x^{(i)}\right)}_{\text{Initial Hamiltonian}} + \underbrace{\frac{B(s)}{2}\left(\sum_i h_i \sigma_z^{(i)} + \sum_{i>j} J_{i,j}\sigma_z^{(i)}\sigma_z^{(j)}\right)}_{\text{Final Hamiltonian}}$$

Tras esta explicación quedan dos cuestiones por resolver. En primer lugar, ¿Qué es el "hueco" de la segunda hipótesis del teorema de evolución adiabática y cuándo se cumple? Este hueco hace referencia a la distancia entre los valores propios. Durante la evolución temporal que se produce en el sistema, los valores propios también varían. Cuanto más se acerquen dichos valores, más probable es que el sistema "salte" a otro de los estados propios del sistema. Este fenómeno se conoce como *quantum tunneling*.

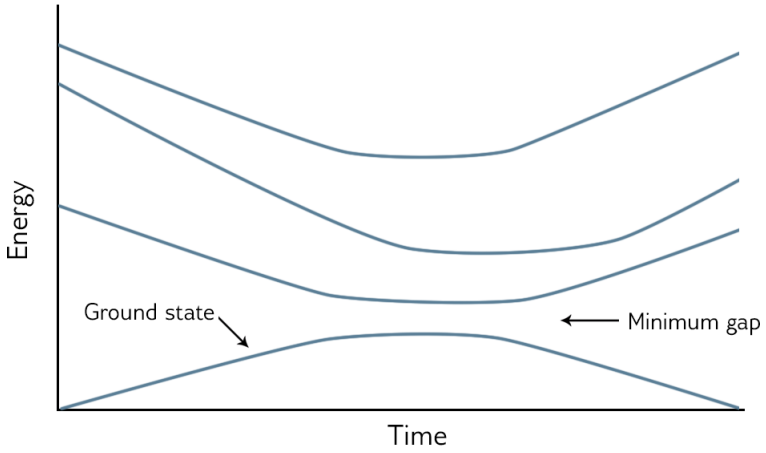


Figura 2: Ejemplo de evolución temporal de valores propios

Dado un problema arbitrario es difícil conocer la evolución de los valores propios temporalmente y mucho menos controlarla. Sin embargo, si la evolución es lo suficientemente lenta, podemos disminuir la probabilidad de dicho salto. Ejecutando el algoritmo en múltiples ocasiones -esto es, utilizando métodos de Monte Carlo- aumentaremos la posibilidad de obtener la solución óptima. Aún así, soluciones cercanas al estado base también obtendrán bajos valores de la función objetivo.

Por otro lado, resta preguntarse qué problemas podemos codificar en un hamiltoniano y cómo hacerlo. Esto serán los problemas QUBO.

2.3. Los problemas QUBO

Los modelos QUBO e Ising pueden codificarse de forma sencilla en un hamiltoniano. En este texto nos centramos en el primero por brevedad, pero el segundo es equivalente. En la práctica, problemas que no son de optimización se transforman en problemas de optimización y posteriormente se codifican como problemas QUBO para ser resueltos utilizando *quantum annealing*. Estudiámos a continuación los problemas QUBO y codificar otros problemas en este formato.

Sea $\mathbb{B} = \{0, 1\}$ y $f_Q : \mathbb{B}^n \longrightarrow \mathbb{R}$ una función polinómica cuadrática sobre variables binaria:

$$f_Q(x) = \sum_{i=1}^{n} \sum_{j=1}^{i} q_{ij} x_i x_j$$

donde $x_i \in \mathbb{B}$ para $i \in \{1, \cdots, n\}$ y los coeficientes $q_{ij} \in \mathbb{R}$ para $1 \leq j \leq i \leq n$. Un problema binario de optimización sin restricciones (***quadratic unconstrained binary optimization* o QUBO**) consiste en encontrar el vector binario x' que minimiza f_Q:

$$x' = \arg\min_{x \in \mathbb{B}^n} f_Q(x)$$

La ventaja de los problemas QUBO es que tienen una descripción matricial sencilla:

$$f_Q(x) = x^T Q x$$

donde Q es una matriz $n \times n$ simétrica que contiene los coeficientes q_{ii} en su diagonal y $q_{ij}/2$ en la posición (i, j) si $i \neq j$. **Esta matriz Q será nuestro hamiltoniano H.** ¿Qué problemas pueden ser codificados de esta forma? A través de una serie de transformaciones sencillas, todo problema cuadrático con restricciones lineales puede ser transformado a un problema de minimización sin restricciones, incluyendo las restricciones del problema original como penalizaciones en nuestra función de pérdida.

En el texto original queda explicado el método de transformación general, así como algunos de los problemas NP-duros más famosos: el problema de corte máximo (Max-Cut), Max SAT-2, el coloreado de grafo y el viajante de comercio.

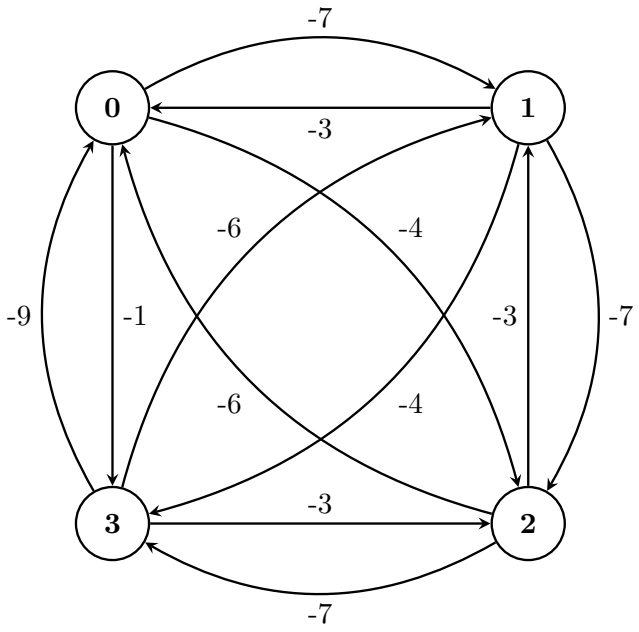
Este último es especialmente relevante para su posterior aplicación en el problema del ensamblaje del genoma. En dicho problema, dado un grafo G con nodos $v_1, \ldots, v_n$, y con pesos asociados a cada lado del grafo $w_{i,j}$ para todo $i, j \in [1, \ldots, n]$, buscamos un camino hamiltoniano (que pasa por todos los nodos del grafo una única vez) con pesos mínimos.

Para ello definimos n^2 variables binarias $x_{i,j}$, que valdrán 1 si y sólamente si nuestro camino pasa por el nodo i en el instante j. Hemos de añadir una serie de restricciones adicionales para que se cumplan las condiciones de ser un camino hamiltoniano por el grafo: unicamente pasamos por un nodo en el instante j, y nunca pasamos dos veces por el mismo nodo. A estas condiciones las denominamos colocación y **repetición** respectivamente.

Añadimos adicionalmente unos pesos negativos en la diagonal de la matriz para, si hiciese falta, premiar el que se asignen nodos. Denominamos a esta condición extra **(self-bias)**. Nuestra función de pérdida final, sin restricciones, es la siguiente:

$$\begin{aligned} \text{Minimizar } & \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} w_{i,j} \sum_{p=0}^{n-1} x_{i,p} x_{j,p+1} \\ & + a \sum_{i=0}^{n-1} \sum_{p=0}^{n-1} x_{i,p} && \text{(self-bias)} \\ & + b \sum_{i=0}^{n-1} \Big(\sum_{p=0}^{n-1} x_{i,p} - 1 \Big)^2 && \text{(repetición)} \\ & + c \sum_{p=0}^{n-1} \Big(\sum_{i=0}^{n-1} x_{i,p} - 1 \Big)^2 && \text{(colocación)} \end{aligned} \tag{1}$$

Donde a, b, c son tres parámetros reales que codifican el peso dado a cada restricción. Hemos transformado nuestro problema original con un grafo de n nodos en un problema QUBO con n^2 variables. Veámos un ejemplo de esta transformación. Dado el siguiente grafo:



Dado que el grafo tiene 4 nodos, definimos $4^2 = 16$ variables binarias y las renombramos:

$$(x_{0,0}, x_{0,1}, x_{0,2}, x_{0,3}, x_{1,0}, x_{1,1}, x_{1,2}, \ldots, x_{3,2}, x_{3,3}) = (x_1, x_2, x_3, x_4, x_5, x_6, x_7, \ldots, x_{15}, x_{16})$$

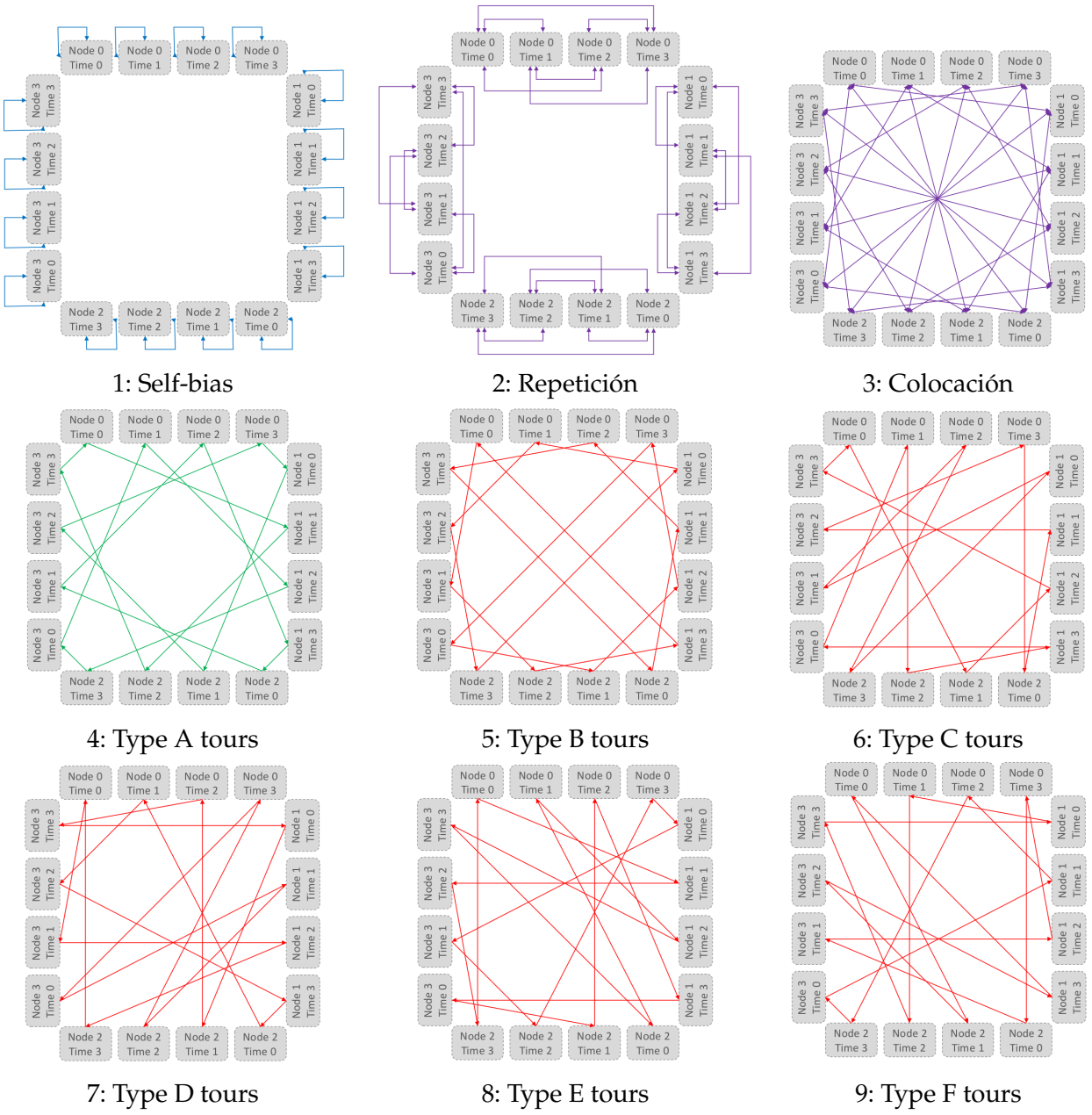
Utilizando los valores $a = 0$ y $b = c = 13$ para los parámetros podemos obtener la expresión de la función de pérdida para nuestro problema, que podemos expresar en forma matricial con la expresión

$$f_Q(x) = x^T Q x.$$

0	13	13	13	13	-14	0	0	13	-8	0	0	13	-2	0	0
13	0	13	13	0	13	-14	0	0	13	-8	0	0	13	-2	0
13	13	0	13	0	0	13	-14	0	0	13	-8	0	0	13	-2
13	13	13	0	-14	0	0	13	-8	0	0	13	-2	0	0	13
13	-6	0	0	0	13	13	13	13	-14	0	0	13	-8	0	0
0	13	-6	0	13	0	13	13	0	13	-14	0	0	13	-8	0
0	0	13	-6	13	13	0	13	0	0	13	-14	0	0	13	-8
-6	0	0	13	13	13	13	0	-14	0	0	13	-8	0	0	13
13	0	0	0	13	-6	0	0	0	13	13	13	13	-14	0	0
-12	13	0	0	0	13	-6	0	13	0	13	13	0	13	-14	0
0	-12	13	0	0	0	13	-6	13	13	0	13	0	0	13	-14
-12	0	-12	13	-6	0	0	13	13	13	13	0	-14	0	0	13
13	-18	0	0	13	0	0	0	13	-6	0	0	0	13	13	13
0	13	-18	0	-12	13	0	0	0	13	-6	0	13	0	13	13
0	0	13	-18	0	-12	13	0	0	0	13	-6	13	13	0	13
-18	0	0	13	-12	0	-12	13	-6	0	0	13	13	13	13	0

Figura 3: Matriz Q final

Esta matriz cuadrada puede verse como una matriz de adyacencia de un grafo con 16 de nodos para obtener una visión intuitiva de las restricciones:



Cuadro 1: Representación gráfica de las penalizaciones entre las distintas intereracciones

2.4. El ensamblaje del genoma

El genoma de un organismo contiene toda su información genética. Dicha información se almacena en el ADN (o en el ARN para ciertos organismos) como cadenas de nucleótidos de entre las cuatro siguientes:

- Adenina (A)

- Timina (T)
- Citosina (C)
- Guanina (G)

Los nucleótidos se colocan en parejas (llamadas *base-pairs* o simplemente bp) en un doble hélice que contiene la información duplicada. Es decir, esta codificación de la información se traduce en una secuencia de letras entre las cuatro posibilidades anteriores. Sin embargo, los dispositivos actuales permiten leer unicamente hasta 1000 bp, teniendo el ADN humano aproximádamente 3 Mbp.

El problema del ensamblaje del genoma consiste en, a partir de un gran conjunto de pequeñas lecturas, recomponer la cadena de ADN original de la que provienen estas lecturas.

Cabe destacar que nos centramos el método denominado *de novo* que no utiliza referencia conocida sobre el ADN de la especie, como si hace el método *ab initio*, añadiendo sesgo. De la misma forma, todo este proceso es equivalente para el ARN aunque las parejas de nucleótidos que lo componen son distintas.

El procedicimiento seguido es transformar este problema en el problema del viajante de comercio, que ya sabemos imbuir en nuestras arquitecturas D-Wave, para posteriormente resolverlo utilizando *quantum annealing*. Para ello creamos un grafo donde cada nodo tiene asignada una cadena. Asignamos pesos a los lados del grafo según lo bien que se solapen dichas cadenas de nucleótidos. Un camino hamiltoniano en este grafo será una ordenación de las cadenas. Si minimizamos los pesos de dicho camino hamiltoniano obtenemos la ordenación que maximiza el pegado entre cadenas. Esto es, la ordenación que recompone el ADN original. A este grafo se le denomina **grafo Overlap-Layout Consensus (OLC)**.

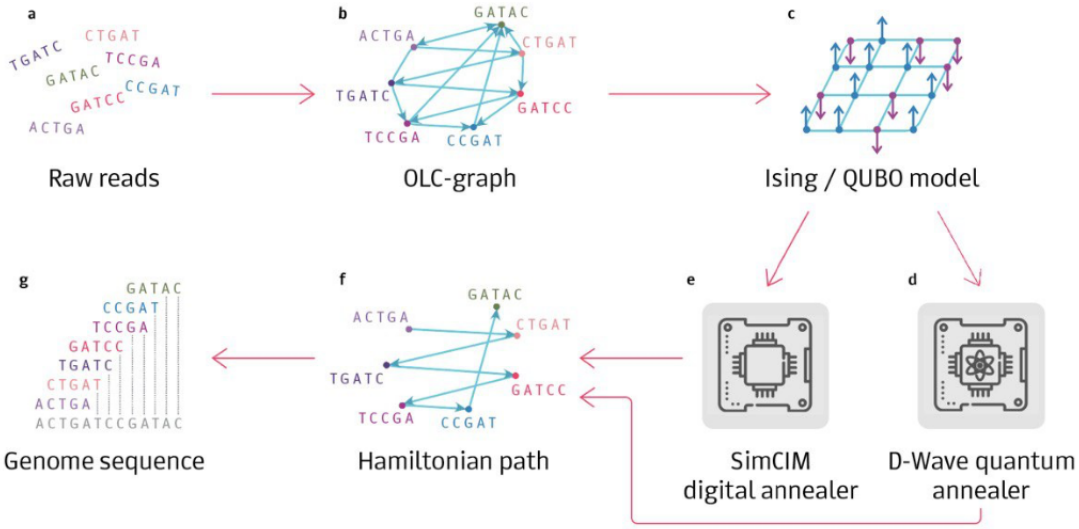


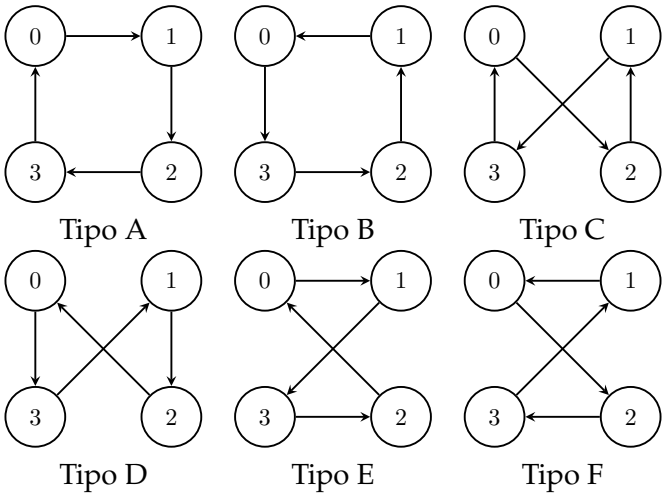
Figura 4: Diagrama del proceso de resolución del ensamblaje del genoma utilizando quantum annealing

2.5. Resultados experimentales

Comentaremos unicamente algunos de los experimentos llevados acabo utilizando los computadores D-Wave. En primer lugar, considerémos un ejemplo con únicamente 4 lecturas:

- $r0 = ATGGCGTGCA$
- $r1 = GCGTGCAATG$
- $r2 = TGCAATGGCG$
- $r3 = AATGGCGTGC$

Donde la solución es unir las lecturas en orden alfanumérico: $r0 \rightarrow r1 \rightarrow r2 \rightarrow r3$. El grafo OLC asociado a este problema es justamente el de la figura **??**. Estudiando este gráfico nos damos cuenta de que hay 6 tipos de ciclo en el mismo, plasmados en la sigueinte figura:



Cuadro 2: Tipos de ciclos de un grafo de 4 nodos

Analizando estos ciclos podemos darnos cuenta de que el tipo de ciclo con menor longitud de pegado (también denominada energía) es el tipo A, como cabría esperar pues esta es la solución. Además, los tipos C, D, E y F tienen todos la misma energía. En último lugar, el ciclo de tipo B tiene la mayor energía de todas.

Figura 5: Energía y pegado de los distintos tipos de ciclos.

En primer lugar, utilizamos un Simulated Annealer, también de la empresa D-Wave, para resol-

ver este problema. Para ello basta con configurar correctamente la matriz Q asociada al problema y conectarse a la API de D-Wave para transmitir esta información. Obtenemos los siguientes resultados:

Tipo de ciclo	**Ocurrencias**	**Energía**
Tipo A	3722	-7.9811
Tipo C	1474	-7.4541
Tipo D	1469	-7.4541
Tipo F	1458	-7.4541
Tipo E	1431	-7.4541
Tipo B	446	-6.927

Cuadro 3: Results of experiment 2

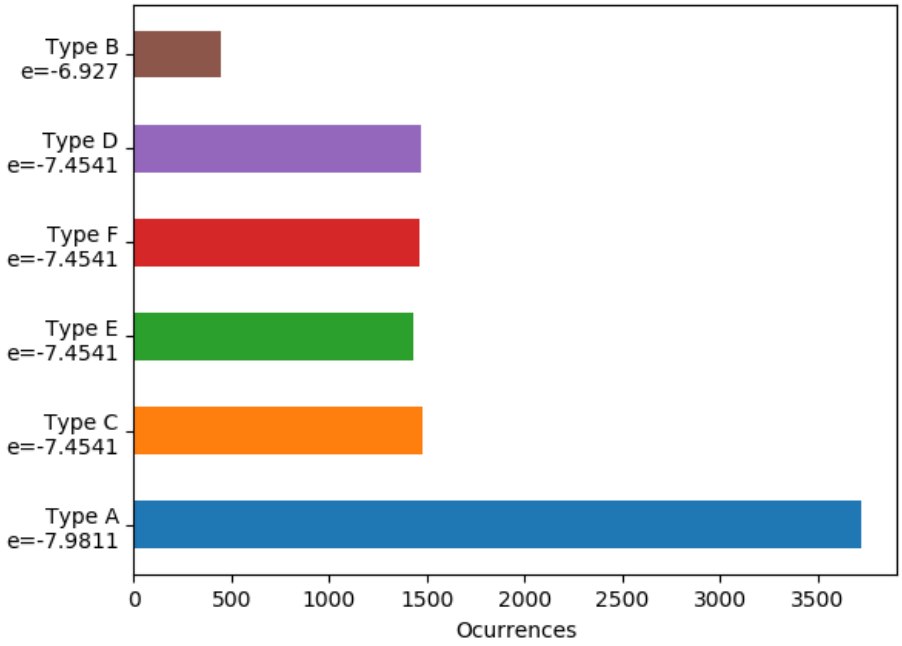


Figura 6: Ocurrencias de cada tipo de ciclo utilizando Simulated Annealing en $10{,}000$ realizaciones.

Realizamos el mismo experimento $10{,}000$ veces y obtenemos la distribución de la figura 6. Estos resultados encajan por completo con el análisis teórico realizado, pues el tipo más frecuente es el A, solución de nuestro problema. Los tipos C, D, E y F obtienen aproximadamente el mismo número de ocurrencias pues tenían la misma energía. Finalmente, el tipo B es el menos común.

Repetimos el experimento utilizando el ordenador cuántico D-Wave 2000. Para ello damos distintos valores a los parámetros. En particular, fijamos $a = -X$ y $b = c = X$ para distintos valores de X. Los mejores resultados se obtuvieron para $X = 1{,}5$ y $1{,}6$, y se muestran a continuación:

Tipo de ciclo	Ocurrencias (X=1.5)	Ocurrencias (X=1.6)	Energía
Tipo A	131	50	-7.9811
Tipo C	124	46	-7.4541
Tipo D	40	91	-7.4541
Tipo F	55	117	-7.4541
Tipo E	112	118	-7.4541
Tipo B	99	64	-7.4541
Inválidos	9221	9194	$>$ -5.6433

Cuadro 4: $10{,}000$ ocurrencias utilizando D-Wave 2000Q.

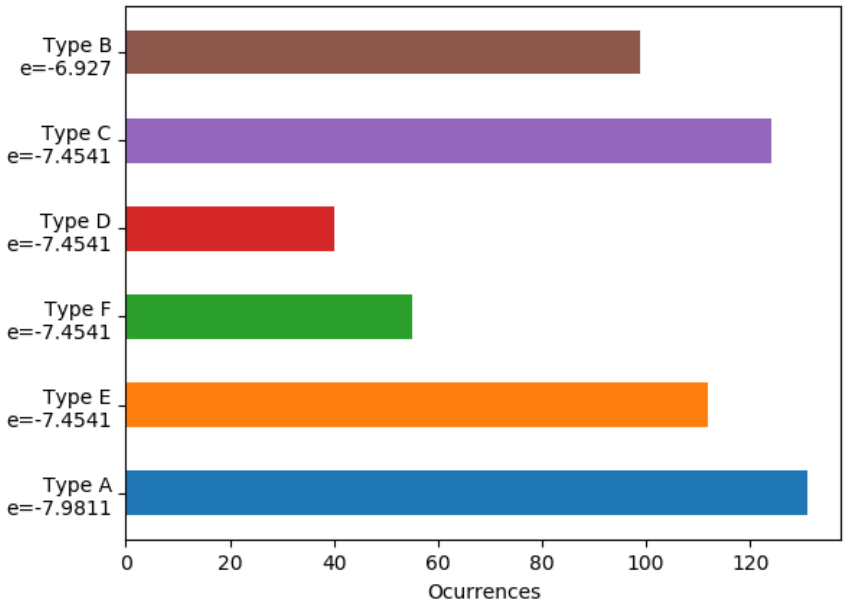


Figura 7: Ocurrencias para cada tipo de ciclo con $10{,}000$ repeticiones utilizando Quantum Annealing, quitando ciclos inválidos, con parámetros QUBO $(-1{,}5; 1{,}5; 1{,}5)$

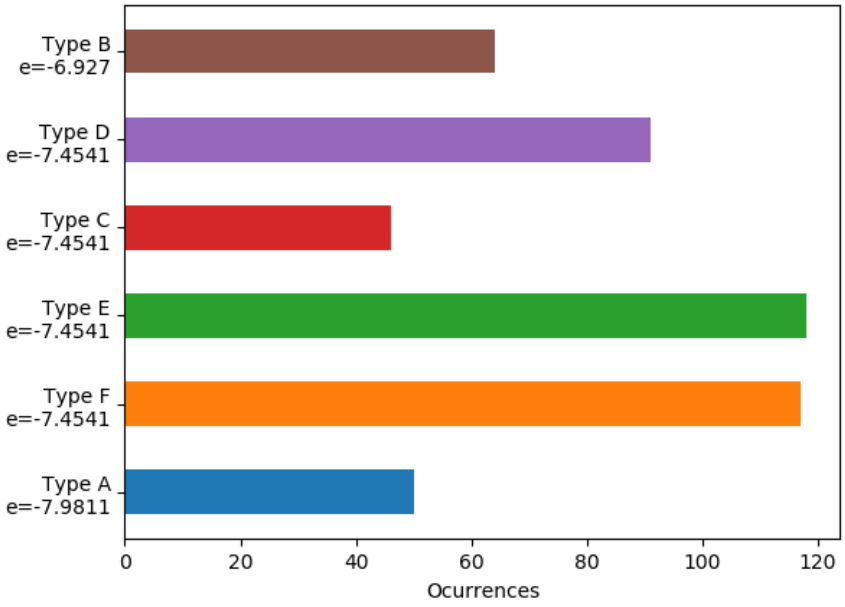


Figura 8: Ocurrencias para cada tipo de ciclo con $10{,}000$ repeticiones utilizando Quantum Annealing, quitando ciclos inválidos, con parámetros QUBO $(-1{,}6; 1{,}6; 1{,}6)$

En primer lugar, el hecho más relevante tras estos experimentos es que obtenemos unicamente un $10\,\%$ de ciclos válidos (esto es, soluciones que representan ciclos que mantienen las restricciones impuestas). En segundo lugar, vemos como no se obtiene una distribución parecida a la esperada. La hipótesis final tras realizar estos experimentos es que el algoritmo no ha tenido suficiente tiempo para realizar el annealing correctamente. Sin embargo, no pudimos realizar más experimentos durante la fase de experimentación, D-Wave provee de únicamente un minuto de tiempo de QPU gratuito al mes.

3. Conclusiones

En este texto hemos expuesto de forma resumida las bases del Quantum Annealing utilizado para los ordenadores D-Wave. Para ello hemos:

1. Expuesto las bases de la mecánica cuántica que subyacen a este modelo, y sus diferencias con el modelo cuántico de circuitos.
2. Explicado en detalle el *quantum annealing* y la evolución adiabática, y el uso que hace D-Wave de estos fenómenos físicos en sus ordenadores.
3. Definido los problemas QUBO, y cómo transformar problemas arbitrarios a este modelo. En particular, hemos codificado el viajante de comercio siguiente este formato.
4. Desarrollado el problema del ensamblaje del genoma y como resolverlo utilizando *quantum annealing*.
5. Expuesto el resultado de diversos experimentos comparando *simulated annealing* y *quantum annealing* para la resolución del problema de ensamblaje del genoma.