**Universidad Internacional de la Rioja (UNIR)**

## Escuela Superior de Ingeniería y Tecnología

**Máster en Computación Cuántica**

# Hacia un algoritmo A* cuántico: incorporación de la heurística en caminatas cuánticas

**Trabajo Fin de Estudios**

**Presentado por:** David Chamizo Sánchez, Eloi Estebanell Pérez y Miguel Ángel Rodríguez García.

**Dirigido por:** Rodrigo Gil-Merino y Rubio.

Ciudad: La Rioja

Fecha: 17 de septiembre de 2025

# Índice general

# Índice de figuras

# Resumen

En el presente Trabajo se explora la posibilidad de formular una versión cuántica del algoritmo A*, ampliamente utilizado en la resolución de problemas de búsqueda y en la resolución del camino más corto/óptimo. Para ello, se propone integrar caminatas cuánticas de Szegedy y la transformada de Doob, con el fin de introducir una heurística en el procedimiento cuántico. La metodología se centra en el análisis teórico y la construcción de un esquema algorítmico que combina registros adicionales y un mecanismo de parada, integrados en un marco cuántico unitario. Los resultados muestran la viabilidad de obtener el camino más óptimo utilizando el algoritmo planteado, demostrando la viabilidad de este y su capacidad de adaptación a distintos grafos. En conclusión, se sientan las bases para un algoritmo A* cuántico, destacando tanto las ventajas como los retos de su implementación en dispositivos de la era NISQ y a futuro.

**Palabras clave:** Algoritmo A*, Caminatas cuánticas de Szegedy, Transformada de Doob, Heurística.

# Abstract

This work explores the possibility of formulating a quantum version of the $A^*$ algorithm, widely used in solving search problems and in finding the shortest/optimal path. To this end, we propose the integration of Szegedy's quantum walks and the Doob transform, with the aim of introducing a heuristic into the quantum procedure. The methodology focuses on theoretical analysis and the design of an algorithmic framework that combines additional registers and a stopping mechanism, all integrated into a unitary quantum model. The results show the feasibility of obtaining the optimal path using the proposed algorithm, demonstrating both its viability and its ability to adapt to different graphs. In conclusion, this work lays the foundation for a quantum $A^*$ algorithm, highlighting both the advantages and the challenges of its implementation in NISQ devices and beyond.

**Keywords:** $A^*$ algorithm, Szegedy quantum walks, Doob transform, Heuristic.

# 1. Introducción

Este Trabajo se basa en la idea de desarrollar un algoritmo cuántico que funcione de manera análoga al algoritmo A* (A estrella) clásico. Dicho algoritmo cuántico debe ser capaz de abordar los mismos problemas que tradicionalmente resuelve A*, pero ofreciendo una mayor eficiencia. La clave está en que el régimen cuántico abre la puerta a métodos más rápidos y efectivos para ciertos casos, lo que hace razonable plantear la creación de un algoritmo A* cuántico.

## 1.1. Motivación del Trabajo

El problema que aborda el algoritmo A*, conocido como el problema del camino más corto (en inglés, *shortest path problem*), es un problema típico en teoría de grafos. Éste consiste en encontrar el camino de coste mínimo que conecte dos nodos de un grafo con aristas ponderadas. Aunque a primera vista pueda parecer sencillo, es fácil notar que la complejidad del problema crece rápidamente en función del número de nodos y aristas que conforman el grafo, lo cual lo convierte en un desafío computacional cuando se trata de grafos de gran tamaño o cuando se requiere una solución en tiempo real.

La búsqueda de caminos óptimos es una necesidad que surge de forma espontánea en la naturaleza, donde podemos ver que algunos organismos han desarrollado estrategias adaptativas que pueden interpretarse como intentos de resolver este problema. Las hormigas, por ejemplo, utilizan feromonas para explorar y reforzar rutas entre una fuente de alimento y su colonia, lo que da lugar a caminos óptimos mediante un proceso descentralizado de optimización colectiva (Colorni et al., 1991). Un caso aún más llamativo es el del moho mucilaginoso *Physarum polycephalum*, que ha demostrado ser capaz de construir redes de túbulos eficientes para conectar diferentes fuentes de nutrientes. En un experimento simulado sobre la geografía de Tokio, este organismo generó una red de conexiones sorprendentemente similar a la red ferroviaria de la ciudad (Tero et al., 2010).

La necesidad de optimizar rutas ha sido también una constante en la historia de las civilizaciones humanas. Desde la planificación de canales y caminos en Mesopotamia y Egipto, hasta la red de calzadas del Imperio Romano, las soluciones al problema del camino más corto han permitido mejorar el comercio, la movilidad y la gestión de recursos. En la actualidad, este mismo problema sigue siendo central en la planificación de infraes-

tructuras, el diseño de redes de comunicaciones y la toma de decisiones automatizadas.

La formalización matemática del problema del camino más corto no llegó hasta bien entrado el siglo XIX, y fue precedida por el desarrollo de la teoría de grafos en el siglo XVIII por Euler (1741), quien propuso su formulación para resolver el célebre problema de los puentes de Königsberg . La primera referencia conocida al problema del camino más corto en sí, aparece en la obra de Wiener (1873), donde se aborda la resolución de laberintos. A pesar de su sencillez aparente, el problema comenzó a atraer atención investigadora en el siglo XX, momento en el que varios investigadores, de forma independiente, desarrollaron algoritmos similares para su resolución (Schrijver, 2012).

Las distintas estrategias que han surgido a lo largo de los años para enfrentarse a distintos problemas de búsqueda, entre los que se encuentra el del camino más corto, pueden clasificarse en dos categorías: los algoritmos no informados y los algoritmos informados. Los algoritmos de búsqueda no informados, también conocidos como algoritmos de búsqueda ciega, son aquellos que exploran el espacio de búsqueda de alguna manera más o menos ordenada, ya que no pueden saber, a priori, si una opción llevará a la solución óptima o no. En el contexto del problema del camino más corto, estos algoritmos garantizan encontrar una solución óptima si existe, aunque su eficiencia puede ser limitada en espacios de búsqueda grandes. Entre los algoritmos no informados más representativos surgidos en el siglo XX se encuentran: la búsqueda en anchura, que explora los nodos por niveles y es óptima cuando el coste de cada acción es uniforme (Russell y Norvig, 2016); y la búsqueda de coste uniforme, una generalización de la anterior que selecciona siempre el nodo de menor coste acumulado desde el inicio. Uno de los algoritmos más influyentes es el de Dijkstra (1959), que resuelve el problema de encontrar el camino más corto hacia todos los vértices desde un vértice inicial. Este último, que destacamos por su importancia histórica, puede verse como un caso particular de la búsqueda de costo uniforme.

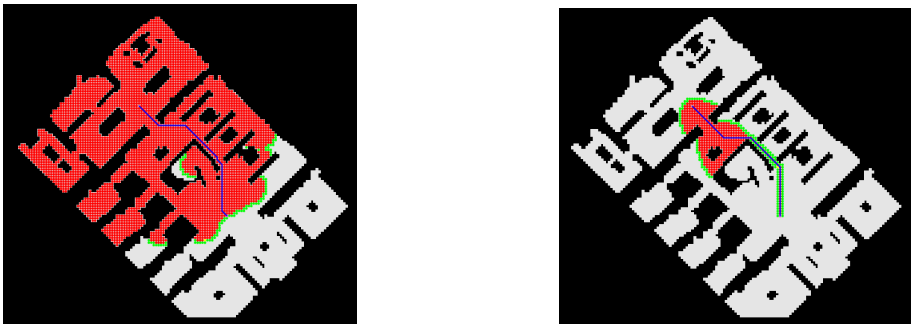
Por otro lado, los algoritmos informados, o heurísticos, son aquellos que cuentan con información sobre el dominio del problema. Esta información se encuentra en el algoritmo en forma de una función de evaluación, la cual puede ser usada para evaluar cómo de prometedores son cada uno de los siguientes nodos que pueden visitar. En este grupo se encuentran los algoritmos voraces, el algoritmo de ascenso de colinas y el mismo A*.

En este contexto, el algoritmo A* surgió como una extensión natural de las estrategias

de búsqueda previas. Fue propuesto en 1968 por Hart et al. (1968) en el contexto del proyecto Shakey, uno de los primeros robots con capacidad de razonamiento sobre sus acciones. En esta publicación introducen formalmente el uso de la información heurística sobre el dominio de un problema en las estrategias de búsqueda, y proporcionan la definición de los conceptos de admisibilidad y optimalidad. Se dice que un algoritmo de búsqueda es admisible cuando garantiza encontrar la solución óptima, y es óptimo cuando la encuentra explorando el mínimo número de nodos necesarios para llegar a dicha solución. Bajo este marco, se demuestra la admisibilidad y la optimalidad del algoritmo A* en términos de dos propiedades que debe cumplir la función heurística utilizada: la admisibilidad y la consistencia. Decimos que una heurística es admisible si nunca sobrestima el coste real, y decimos que es consistente si, para cualquier estado, su valor heurístico es menor o igual que el coste de avanzar a un estado adyacente más el valor heurístico de este último.

Uno de los aspectos clave que muestra la versatilidad del algoritmo A* es su capacidad para trabajar con representaciones atómicas del estado. Tal y como introducen Russell y Norvig (2016), la representación atómica modela un problema como un conjunto de estados y operadores, permitiendo que el algoritmo de búsqueda no dependa de los detalles específicos del dominio. Este enfoque ha permitido extender la aplicación de algoritmos de búsqueda, como A*, a problemas tan diversos como la planificación automática, la resolución de puzzles, el análisis de redes neuronales y la ingeniería de software. En la Figura 1.1 se puede ver una comparación entre un algoritmo no informado, como es el algoritmo de Dijkstra, y un algoritmo informado, como es A*. En rojo se representan los nodos visitados a lo largo de la búsqueda y en verde los que se pueden alcanzar pero aún no han sido visitados. Lo primero que notamos en la imagen es la cantidad inferior de nodos que explora A* en comparación con un algoritmo no informado.

A pesar del hito que representa este algoritmo dentro del campo de la computación y las estrategias de búsqueda, A* presenta ciertas limitaciones cuando se aplica a dominios de búsqueda de gran escala. Su principal inconveniente radica en el elevado consumo de memoria, ya que requiere almacenar todos los nodos generados durante la búsqueda hasta encontrar la solución, lo cual puede resultar prohibitivo en problemas con espacios de estados extensos o altamente ramificados. Además, aunque el uso de una heurística admisible garantiza la optimalidad de la solución, la eficiencia del algoritmo depende en gran medida de la precisión de dicha heurística. En ausencia de una estimación informada, A* se



(a) Dijkstra (b) A*

Figura 1.1: Comparación entre el algoritmo de Dijkstra (a) y el algoritmo A* (b) en una misma instancia de problema. Los nodos visitados se representan en rojo y los nodos en la frontera en verde. Fuente: Sturtevant (2025).

comporta de una forma similar a la búsqueda de coste uniforme, explorando una cantidad considerable de nodos no relevantes. Estas limitaciones han impulsado el desarrollo de variantes y enfoques paralelos diseñados para mejorar su escalabilidad y rendimiento en entornos complejos. Una de las variantes más conocidas que intenta resolver el problema del elevado consumo de memoria es IDA* (Korf, 1985), que en lugar de buscar por todo el grafo desde un inicio, realiza búsqueda iterativas limitadas en profundidad.

La investigación en las últimas décadas en este campo ha estado enfocada a resolver estos problemas. Por un lado, con algoritmos limitados en memoria, como hemos anteriormente. Y, por otro lado, intentando dar con formulaciones paralelas del algoritmo, que si bien no reducen el consumo de memoria, si nos permiten obtener un menor tiempo de ejecución y aprovechar todo el potencial de los ordenadores más modernos. Este segundo acercamiento para mejorar el rendimiento del algoritmo ha sido estudiado por uno de nosotros en un Trabajo Fin de Grado (Rodríguez García, 2025), en el que se introduce el algoritmo A* en profundidad y las principales implementaciones paralelas existentes. Más recientemente, podemos encontrar versiones paralelas pensadas para arquitecturas de procesamiento gráfico (GPU, por sus siglas en inglés) como la de Zhou y Zeng (2015).

Estas dificultades que se presentan al intentar resolver el problema del camino más corto en instancias grandes, ha llevado en los últimos años a buscar algoritmos eficientes en el campo de la computación cuántica, un nuevo paradigma de computación basado en los principios de la mecánica cuántica. Gracias a ciertos fenómenos exclusivos de esta teoría se espera que ciertos algoritmos cuánticos puedan ofrecer ventajas significativas respecto a

sus homólogos clásicos, especialmente en tareas de optimización y búsqueda, como el caso del problema del camino más corto.

## 1.2. Planteamiento del Trabajo

El presente Trabajo tiene como objetivo explorar una versión cuántica del algoritmo de búsqueda A*, ampliamente utilizado en la resolución de problemas de planificación y búsqueda de rutas óptimas sobre grafos. Se busca aprovechar las propiedades fundamentales de los sistemas cuánticos, como la superposición y el paralelismo, con el fin de reducir la complejidad computacional inherente a este tipo de algoritmos en escenarios de gran escala.

El algoritmo A* destaca por combinar una búsqueda informada con el uso de heurísticas para reducir el espacio de búsqueda. No obstante, cuando se enfrenta a grafos complejos o de gran tamaño, su rendimiento se ve limitado por el crecimiento en el número de elementos a explorar. Este hecho motiva el desarrollo de una alternativa basada en computación cuántica, que permita implementar una dinámica de búsqueda más eficiente.

El propósito central de este estudio consiste, por tanto, en analizar y proponer una posible formulación de A* cuántico mediante la integración de herramientas como la búsqueda de Grover, las caminatas cuánticas y técnicas heurísticas adaptadas al paradigma cuántico. Se pretende establecer un marco teórico que justifique la viabilidad de dicha integración y evaluar su aplicabilidad. El Trabajo presentará un conjunto de ejemplos de aplicación del algoritmo propuesto a unas pocas instancias del problema, analizando su eficiencia y coste computacional.

## 1.3. Estructura del Trabajo

Hasta ahora se ha introducido el problema que motiva este Trabajo y se ha planteado la propuesta de explorar una formulación cuántica del algoritmo A*. A partir de este punto, el documento se organiza en una serie de capítulos que siguen un hilo conductor orientado a desarrollar, justificar y evaluar dicha propuesta.

En el Capítulo 2 se presenta el contexto y estado de la técnica, donde se revisan los conceptos fundamentales de la teoría de grafos y del problema del camino más corto,

así como los principales algoritmos clásicos utilizados para resolverlo. Posteriormente, se expone el marco de la computación cuántica y se describen las principales aproximaciones cuánticas aplicadas a problemas en grafos, con especial énfasis en las caminatas cuánticas, que constituirán la base metodológica de este Trabajo.

El Capítulo 3 recoge los objetivos de la investigación, distinguiendo entre los de carácter general, que definen la finalidad global del estudio, y los específicos, que especifican las metas parciales necesarias para avanzar en la construcción de un algoritmo A* cuántico.

El Capítulo 4 constituye el núcleo central del documento, pues en él se desarrolla la propuesta del algoritmo. Se comienza estableciendo las condiciones de aplicabilidad y, a continuación, se describen en detalle los distintos componentes que lo conforman: la caminata cuántica de Szegedy, el kernel de transición, la transformada de Doob, los registros adicionales para representar caminos y longitudes, el mecanismo de parada y la técnica de amplificación de amplitud cuántica. Finalmente, se muestra cómo se integran todos estos elementos en un esquema algorítmico completo.

En el Capítulo 5 se presentan los resultados obtenidos, junto con un análisis crítico y comparativo que permite valorar tanto las ventajas como las limitaciones de la propuesta desarrollada en relación con otros enfoques existentes.

El Capítulo 6 recoge las conclusiones, donde se sintetizan las principales aportaciones, se discute su alcance y se evalúa el grado de cumplimiento de los objetivos planteados.

Por último, en el Capítulo 7 se plantea el trabajo futuro, identificando las posibles líneas de investigación que pueden dar continuidad a esta propuesta y señalando mejoras que podrían contribuir a perfeccionar el algoritmo en contextos más amplios o en escenarios prácticos de mayor complejidad.

# 2. Contexto y estado de la técnica

El problema del camino más corto constituye una de las formulaciones más estudiadas dentro de la teoría de grafos y la optimización combinatoria. Su estudio ha sido fundamental no sólo en matemáticas aplicadas, sino también en campos tan diversos como la informática teórica, la inteligencia artificial, la logística o las redes de comunicación (Gross y Yellen, 2005; Cormen et al., 2022),

En términos generales, este problema plantea la siguiente cuestión: dado un conjunto de elementos interconectados mediante relaciones de coste, ¿cuál es la secuencia óptima de transiciones que minimiza el coste total desde un origen hasta un destino?

Para comprender mejor el funcionamiento de los algoritmos de búsqueda en grafos, vamos a introducir en primer lugar los conceptos necesarios para poder hacer un modelado del problema. A continuación, daremos una descripción del problema del camino más corto y veremos algunos de los algoritmos clásicos más importantes para resolverlo.

## 2.1. Representación mediante grafos

Un grafo es una estructura matemática representada típicamente como un par ordenado $G = (V, E)$, donde $V$ es el conjunto de nodos (o vértices) y $E$ es el conjunto de aristas (o enlaces) que conectan pares de nodos (Bondy y Murty, 2008). Cada elemento de $E$ vendrá representado como un par $e_{ij} = (v_i, v_j) \in E$, que indica que los nodos $v_i, v_j \in V$ están conectados.

Cuando los elementos de $E$ no son pares ordenados, esto es, $e_{ij} = e_{ji}$, hablamos de grafo no dirigido, y en caso contrario hablaremos de grafo dirigido. Cabe mencionar que, en el caso de grafos dirigidos, la existencia de una arista $e_{ij}$ no implica la existencia de la arista $e_{ji}$. En la Figura 2.1 podemos ver ejemplos de ambos tipos de grafo.

Para poder terminar de representar los elementos del problema del camino más corto necesitamos ponderar las aristas. En este caso hablaremos de grafos ponderados (dirigidos o no), en los que cada arista $e_{ij}$ tendrá asociada un costo $w(v_i, v_j) = w_{ij} \in \mathbb{R}$. En la Figura 2.2 podemos ver representado un grafo dirigido con sus aristas ponderadas con pesos positivos.


(a) (b)

Figura 2.1: Ejemplos de grafo no dirigido (a) y dirigido (b) con cuatro vértices. Fuente: elaboración propia.

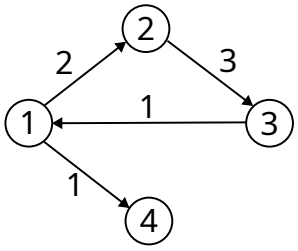



Figura 2.2: Ejemplo de grafo dirigido con aristas ponderadas con pesos positivos. Fuente: elaboración propia.

Esta forma explícita de definir un grafo a partir de sus conjuntos de nodos y aristas atiende a la manera convencional en teoría de grafos, pero cuando se trata con estos problemas desde el punto de vista de la computación, es usual utilizar otra forma equivalente. Se trata de una definición implícita a partir de un conjunto de nodos fuente $S \subset V$ y un operador de vecindad, $\Gamma$, definido sobre el conjunto de elementos $V$ y cuyo valor para cada $v_i$ es un conjunto de pares $\{(v_j, w_{ij})\}$. De esta forma, a partir de un nodo inicial y un operador de vecindad podemos obtener el grafo completo.

## 2.2. El problema del camino más corto

Podemos definir el problema del camino más corto de la siguiente manera: Sea $G = (V, E)$ un grafo (dirigido o no) con sus aristas ponderadas por pesos positivos, un nodo fuente $s \in V$ y un nodo destino $t \in V$, hallar una secuencia de nodos $P = (v_1, v_2 \ldots, v_k)$ tal que:

- $v_1 = s$ y $v_k = t$
- Para todo $1 \leqslant i < k$, $(v_i, v_{i+1}) \in E$

- El coste total del camino $P$, viene definido como $w(P) = \sum_{i=1}^{k-1} w_{i,i+1}$

Esta es la formulación clásica del problema del camino más corto, aunque podemos encontrar modificaciones para resolver el problema entre cualquier par de nodos (en inglés, *all-pairs shortest path problem*) o desde un único nodo fuente hasta todos los demás (en inglés, *single-source shortest path*)(Deo, 2016).

De esta manera, un ejemplo de problema del camino más corto podemos verlo en la Figura 2.3, donde el nodo inicial sería $A$ y el final $F$. En este caso, el camino óptimo entre $A$ y $F$ vendría dado por la secuencia $(A, B, D, F)$, con coste 16.

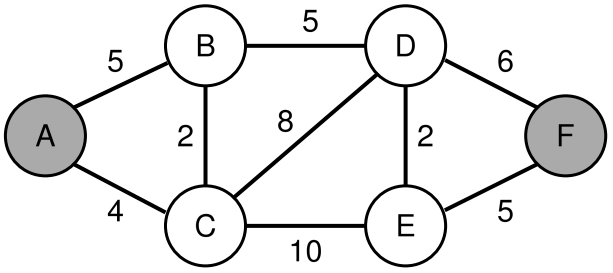



Figura 2.3: Ejemplo del problema del camino más corto. Se trata de un grafo ponderado con 6 nodos y se quiere encontrar el camino óptimo entre $A$ y $F$, que viene dado por la secuencia $(A, B, D, F)$, con coste 16. Fuente: elaboración propia.

## 2.3. Algoritmos clásicos para problemas en grafos

Con el objetivo de comprender cómo se ha abordado este problema desde el marco de la computación clásica, vamos a introducir los algoritmos más relevantes tanto por lo que han significado para investigaciones posteriores como por su eficiencia para resolver el problema.

### 2.3.1. Algoritmo de Dijkstra

Uno de los algoritmos más conocidos en el mundo de la computación clásica es sin duda el algoritmo de Dijkstra (1959). Se trata de un algoritmo que resuelve el problema de encontrar el camino más corto hacia cualquier nodo en un grafo desde un nodo inicial, siempre que los pesos de las aristas sean no negativos. La importancia de este algoritmo

radica no solo en su eficiencia y elegancia, sino en la profunda influencia que tuvo en el desarrollo de algoritmos posteriores.

El trabajo pionero de Dijkstra, publicado en 1959 (aunque concebido en 1956), sentó las bases para lo que luego se conocería como búsqueda de costo uniforme (UCS, por sus siglas en inglés), un algoritmo no informado que generaliza el concepto de búsqueda en amplitud (BFS, por sus siglas en inglés) para entornos con costos variables. De hecho, la búsqueda de costo uniforme puede considerarse una reinterpretación del algoritmo de Dijkstra, donde el enfoque se centra en expandir siempre el nodo con menor costo acumulado desde el origen, garantizando optimalidad en grafos con pesos no negativos. El algoritmo de Dijkstra utiliza una cola de prioridad que selecciona, en cada iteración, el nodo $n$ con menor distancia estimada $g(n)$. El funcionamiento, siendo $s$ el nodo inicial, es el siguiente:

1. Se inicializa $g(v) = \infty$ para todo $v \in V \backslash \{s\}$ y $g(s) = 0$. Se añade $s$ a la cola de prioridad.

2. Se selecciona el nodo $n$ con menor $g(n)$ de la cola de prioridad.

3. Para cada vecino $m$ de $n$ actualizar $g(m) \leftarrow \text{mín}\{g(m),\ g(n) + w(n, m)\}$. Vuelve al paso 2.

Este proceso se repite hasta que todos los nodos han sido visitados o se ha alcanzado el nodo destino, si se busca un camino único. El coste del algoritmo tanto en tiempo de ejecución como en memoria es $\mathcal{O}(b^{1+C^*/\epsilon})$, donde $b$ es el factor de ramificación, $C^*$ es el costo óptimo de la solución y $\epsilon$ es el coste mínimo de cualquier arista en el grafo.

Cabe destacar que, aunque el algoritmo de Dijkstra fue revolucionario, no fue el primer intento de resolver problemas de caminos óptimos. Shimbel (1951) había abordado el problema unos años antes mediante enfoques matriciales, mientras que Ford (1956) desarrolló casi simultáneamente un algoritmo capaz de manejar pesos negativos, que luego evolucionaría en el conocido algoritmo Bellman-Ford. Sin embargo, la claridad conceptual y eficiencia práctica del enfoque de Dijkstra lo consolidaron como referencia fundamental en la teoría de grafos y optimización.

### 2.3.2. Algoritmo de Bellman-Ford

El algoritmo de Bellman-Ford, desarrollado independientemente por Bellman (1958) y Ford (1956), representa un avance fundamental en la teoría de grafos al resolver el problema de caminos más cortos en grafos con pesos arbitrarios, incluyendo aristas negativas. En este tipo de escenarios, algoritmos como el de Dijkstra o UCS pueden llevar a soluciones incorrectas debido a su criterio de selección *greedy*. Esta capacidad lo hace extremadamente importante en aplicaciones en las que ciertos caminos pueden representar ganancias netas, como modelos financieros o de flujo en redes de comunicaciones. El algoritmo opera de la siguiente manera:

1. Se inicializa $g(v) = \infty$ para todo $v \in V \backslash \{s\}$ y $g(s) = 0$. Se añade $s$ a la cola de prioridad.

2. Repetir $|V| - 1$ iteraciones:

   *a*) Para todas las aristas tal que $(n, m) \in E$, actualizar $g(m) \leftarrow \text{mín}\{g(m),\ g(n) + w(n, m)\}$.

3. Para cada arista $(n, m)$:

   *a*) Si $g(m) > g(n) + w(n, m)$ hay ciclo negativo en el grafo.

Esta técnica garantiza encontrar el camino más corto mientras no haya ciclos de peso negativo alcanzables desde el origen. En un último bucle adicional, si alguna distancia todavía puede ser reducida, se detecta la existencia de un ciclo negativo. Su complejidad temporal es $\mathcal{O}(|V| \cdot |E|)$, lo cual lo hace menos eficiente que Dijkstra en grafos densos, pero significativamente más robusto (Cormen et al., 2022). Esto es, soporta una mayor variedad de estructuras diferentes sobre las que es aplicable.

El trabajo de Ford no solo formalizó el problema de flujo en redes, sino que estableció los principios para algoritmos posteriores que integran restricciones de coste y capacidad. Su informe técnico sigue siendo referencia obligada en optimización de redes, evidenciando cómo soluciones teóricas de mediados del siglo XX continúan impulsando aplicaciones modernas.

### 2.3.3. Algoritmo A*

El algoritmo A* es un algoritmo de búsqueda informada que combina las ventajas de la búsqueda de costo uniforme y la búsqueda "primero el mejor". Su objetivo es encontrar el camino de menor costo desde un nodo inicial $s$ hasta un nodo objetivo $t$ en un grafo dirigido ponderado. El elemento diferenciador de este algoritmo es su función de evaluación, que se define como:

$$f(n) = g(n) + h(n) \tag{2.1}$$

donde $g(n)$ representa el coste real acumulado desde un nodo inicial hasta el nodo $n$ por un camino óptimo, y $h(n)$ el coste real del camino óptimo para llegar al nodo objetivo desde dicho nodo. Si solo consideramos la función $g(n)$, el algoritmo A* sería equivalente al algoritmo de búsqueda de costo uniforme y, si solo consideramos $h(n)$, entonces es equivalente al algoritmo "primero el mejor". El hecho de tener en cuenta ambas funciones hace que el algoritmo nos permita no solo encontrar la solución óptima, sino también de manera más eficiente.

El objetivo del algoritmo es encontrar el camino que minimiza el valor de $f$, pero no siempre podemos conocer con exactitud los valores de $h$ y $g$, por lo que tendremos que utilizar una estimación de estos, es decir:

$$\hat{f}(n) = \hat{g}(n) + \hat{h}(n) \tag{2.2}$$

La elección que proponen Hart et al. (1968) para la elección de $\hat{g}(n)$ es el menor coste encontrado hasta el momento actual entre el nodo inicial y $n$. Esta elección implica que $\hat{g}(n) \geqslant g(n)$. La elección de $\hat{h}(n)$ es más complicada y requiere de un conocimiento explícito del dominio del problema. En algunos casos, como por ejemplo, encontrar el camino entre dos puntos de un mapa, la distancia en línea recta sirve como estimación, pero en otros dominios puede no ser una elección trivial. De hecho, uno de los principales problemas que se presentan al querer aplicar este algoritmo en un problema real es determinar esta estimación de $h$.

La manera de proceder del algoritmo será la de aplicar el operador de vecindad al estado actual para obtener sus estados vecinos. Posteriormente, explorará aquel que presente el valor de $\hat{f}$ más pequeño. Para tener en cuenta aquellos estados que ya han sido visitados, y los que quedan por visitar, se utilizan dos estructuras de datos llamadas lista abierta $Q$

y lista cerrada $H$. La lista abierta almacena aquellos estados a los que podemos llegar pero aún no hemos explorado, y la lista cerrada aquellos que ya han sido explorados habiendo sido sus vecinos introducidos en la lista abierta. Decimos que un estado está "abierto" o "cerrado" si está en la lista abierta o cerrada, respectivamente. El algoritmo, tal y como lo definen Hart et al. (1968), opera de la siguiente manera:

1. Marcar $s$ como abierto y calcular $\hat{f}(s)$.

2. Seleccionar el nodo abierto $n$ con el mínimo valor de $\hat{f}(n)$.

3. Si $n = t$, marcar $n$ como cerrado y terminar el algoritmo.

4. Si no, marcar $n$ como cerrado y aplicar el operador $\Gamma$ a $n$. Calcular $\hat{f}(n)$ para cada vecino de $n$ y marcar como abiertos aquellos que no están marcados como cerrados. Remarcar como abiertos los sucesores ya marcados cerrados $n_i$ para los que el valor de $\hat{f}(n_i)$ es menor ahora que cuando fueron marcados cerrados. Volver al paso 2.

Al ser un algoritmo heurístico, la eficiencia de éste dependerá mucho de las características de la función elegida como heurística. Cuanto más parecida sea la estimación al valor real de $h$, menos nodos expandirá el algoritmo. Sabemos que, como mucho, A* expandirá todos aquellos nodos con un valor $f(n) \leqslant C^*$, siendo $C^*$ el coste del camino óptimo. En la literatura más moderna, en relación con la inteligencia artificial, se puede ver la eficiencia del algoritmo expresada en el peor caso como $\mathcal{O}(b^d)$, donde $b$ es el factor de ramificación y $d$ la profundidad del nodo objetivo. En cuanto a la complejidad en espacio, A* guarda todos los nodos generados en la lista cerrada y los nodos por explorar en la lista abierta, y en cada iteración visita un nodo, por lo que la complejidad en espacio es también exponencial con respecto a la longitud del camino $d$.

## 2.4. Paradigma de la computación cuántica

La computación cuántica representa un paradigma completamente distinto al de la computación clásica, basándose en las leyes de la mecánica cuántica para crear un nuevo modelo de computación. Frente a la máquina clásica de Turing y la lógica binaria tradicional, los ordenadores cuánticos manipulan información codificada en sistemas físicos microscópicos como átomos, iones o circuitos superconductores, donde fenómenos como la

superposición y el entrelazamiento permiten un procesamiento distinto del espacio de estados. Esto resulta en una capacidad mucho mayor para la resolución de ciertos problemas que su contraparte clásica.

La idea de aprovechar los fenómenos de la teoría cuántica para la computación surgió en la década de 1980 de la mano de Benioff (1980), quien fue el pionero que propuso el concepto de una máquina cuántica de Turing en su artículo "*The computer as physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines*". Posteriormente, Feynman (1982) presentó "*Simulating Physics with Computers*", donde argumentó que los sistemas cuánticos naturales seguían reglas muy diferentes a la física clásica, y por tanto, solamente un ordenador cuántico podría simular eficientemente fenómenos cuánticos. Más adelante, Deutsch (1985) describió el primer ordenador cuántico universal en su artículo "*Quantum theory, the Church-Turing principle and the universal quantum computer*". En él, propuso que cualquier ordenador cuántico podría simular el funcionamiento de cualquier otro ordenador cuántico, estableciendo los fundamentos teóricos para la computación cuántica universal.

A continuación se proporciona una introducción rigurosa a los principios fundamentales de la computación cuántica, los cuales constituyen la base sobre la que se construirá más adelante el diseño del algoritmo cuántico para el problema del camino más corto.

### 2.4.1. Cúbits, superposición y puertas cuánticas

El cúbit (*qubit* en inglés, abreviatura de *quantum bit*), o bit cuántico, constituye la unidad básica de información en la computación cuántica, análoga al bit clásico en computación clásica. Sin embargo, a diferencia del bit clásico, que solo puede encontrarse en uno de dos estados mutuamente excluyentes (0 o 1), un cúbit puede existir en una superposición coherente de ambos estados base. Matemáticamente, el estado de un cúbit se describe como un vector unitario en un espacio de Hilbert complejo de dos dimensiones, es decir, $\mathcal{H} \cong \mathbb{C}^2$. Cualquier estado puro de un cúbit puede expresarse como una combinación lineal de la base de dicho espacio de Hilbert, $|0\rangle$ y $|1\rangle$, que conforman la llamada base computacional:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \quad \text{con } \alpha, \beta \in \mathbb{C},\ |\alpha|^2 + |\beta|^2 = 1. \tag{2.3}$$

La condición de normalización $|\alpha|^2 + |\beta|^2 = 1$ garantiza que la probabilidad total de medir

el sistema en alguno de los estados base sea igual a la unidad (Nielsen y Chuang, 2010). Una forma común de representar los cúbits es mediante la esfera de Bloch (Figura 2.4).

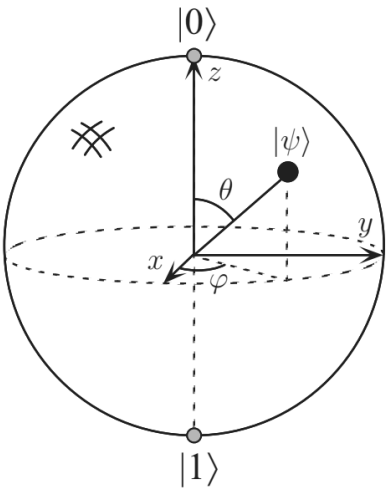



Figura 2.4: Esfera de Bloch. Se trata de una esfera de radio unidad en la que se pueden representar estados arbitrarios de un cúbit tal que $|\psi\rangle = \cos(\frac{\theta}{2})\,|0\rangle + e^{i\varphi}\sin(\frac{\theta}{2})\,|1\rangle$, donde $\alpha = \cos\frac{\theta}{2}$ y $\beta = e^{i\varphi}\sin\frac{\theta}{2}$. Fuente: Nielsen y Chuang (2010).

El fenómeno de la superposición cuántica permite que un único cúbit almacene simultáneamente información sobre ambos valores lógicos. Esta propiedad se traduce en una ventaja computacional cuando se manipulan múltiples cúbits, ya que permite representar exponencialmente más estados con una cantidad lineal de elementos. Por ejemplo, un sistema de $n$ cúbits puede describirse por un vector en $\mathbb{C}^{2^n}$, conteniendo hasta $2^n$ coeficientes complejos. Se permite así representar funciones sobre un dominio de $2^n$ entradas utilizando únicamente $n$ cúbits. Esta codificación posibilita la computación en paralelo característica de la computación cuántica. En ella, una serie de operaciones unitarias aplicadas a un registro en superposición afecta simultáneamente a todas las componentes de dicho registro.

No obstante, la naturaleza de la medición en mecánica cuántica impide acceder directamente a todos los resultados en paralelo. Para extraer información útil del sistema, los algoritmos cuánticos deben explotar la interferencia cuántica. Es decir, la amplificación o cancelación de amplitudes de probabilidad para que los resultados deseados tengan mayor probabilidad de ser observados. Este principio subyace a varios algoritmos cuánticos emblemáticos, como el de Deutsch y Jozsa (1992), el algoritmo de Grover (1996) y el algoritmo de factorización de Shor (1997).

La manipulación de los cúbits para realizar esta serie de operaciones se realiza mediante puertas cuánticas, que corresponden a operadores unitarios sobre el espacio de Hilbert. Estas puertas son similares a las puertas lógicas clásicas, pero con una estructura algebraica mucho más estricta, ya que deben preservar la norma de los estados (i.e., deben ser unitarias). Algunas puertas fundamentales incluyen:

- **Hadamard (H)**: convierte un estado base en una superposición uniforme. Es crucial para inicializar cúbits en estados de superposición:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle).$$

- **Pauli-X (NOT cuántico)**: actúa como una compuerta de negación binaria, intercambiando $|0\rangle \leftrightarrow |1\rangle$. Su representación matricial es:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

- **CNOT (Controlled-NOT)**: es una puerta de dos cúbits donde el segundo cúbit (objetivo) se invierte si y solo si el primero (control) está en el estado $|1\rangle$. Esta puerta introduce correlaciones entre cúbits y permite generar entrelazamiento:

$$\text{CNOT}(|x\rangle \otimes |y\rangle) = |x\rangle \otimes |x \oplus y\rangle, \quad x, y \in \{0, 1\}.$$

Estas puertas, junto con otras como las de fase (S, T), rotaciones $(R_x, R_y, R_z)$ y operaciones multicúbit como Toffoli o SWAP, permiten construir cualquier circuito cuántico, constituyendo un conjunto universal de puertas cuánticas. En la Figura 2.5 podemos ver un ejemplo de la construcción de una puerta Toffoli a partir de otro conjunto de puertas.

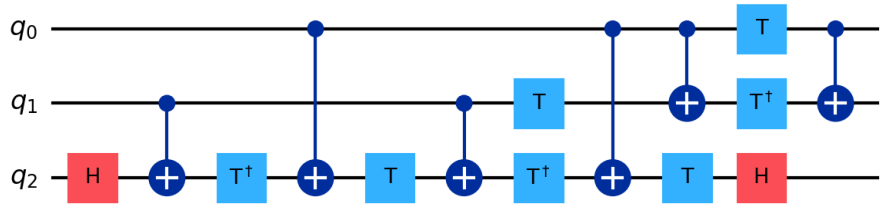



Figura 2.5: Construcción de una puerta Toffoli (izquierda) a partir de un conjunto de puertas, en este caso CNOT, Hadamard, la puerta T y su conjugada (derecha). Fuente: elaboración propia.

### 2.4.2. Entrelazamiento cuántico

Uno de los recursos más distintivos de la computación cuántica es el entrelazamiento (*entanglement*), una forma de correlación que resulta cuando el estado global de un sistema no puede describirse como un producto tensorial de los estados de sus subsistemas. En tales casos, la información del sistema está distribuida globalmente, y no puede ser atribuida a ninguna parte por separado. Esto es, la medición del estado en uno de los subsistemas deriva en el colapso de la función de estado del resto de los subsistemas sujetos al entrelazamiento. Un ejemplo de esta situación es el primer estado de Bell:

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), \tag{2.4}$$

que representa un sistema de dos cúbits perfectamente correlacionados. Este estado es puro y se encuentra máximamente entrelazado. Su estructura impide describirlo como $|\psi\rangle_A \otimes |\phi\rangle_B$. Como consecuencia, una medición sobre uno de los cúbits colapsa instantáneamente el estado del otro, sin importar la distancia que los separe. Este comportamiento fue formalmente discutido en el teorema de Bell (1964) y comprobado experimentalmente en múltiples ocasiones (Aspect et al., 1982; Hensen, 2015), siendo un indicio inequívoco del carácter no local de la mecánica cuántica. El entrelazamiento es indispensable en diversos protocolos cuánticos, como la teleportación cuántica, la corrección de errores y la criptografía cuántica (Steane, 1998).

### 2.4.3. Evolución unitaria y medición

En mecánica cuántica, la evolución temporal de un sistema cerrado (aislado de cualquier entorno externo) está completamente determinada por la ecuación de Schrödinger dependiente del tiempo, una ecuación diferencial lineal de primer orden sobre el espacio de Hilbert del sistema. Dicha ecuación se expresa como:

$$i\hbar\frac{d}{dt}|\psi(t)\rangle = H|\psi(t)\rangle, \tag{2.5}$$

donde $|\psi(t)\rangle$ es el estado cuántico en el tiempo $t$, $\hbar$ es la constante de Planck reducida y $H$ es el hamiltoniano del sistema. La solución general de esta ecuación, bajo la suposición de un hamiltoniano dependiente del tiempo, es una evolución unitaria del estado inicial:

$$|\psi(t)\rangle = U(t)|\psi(0)\rangle, \quad \text{con} \quad U(t) = e^{-iHt/\hbar}, \tag{2.6}$$

donde $U(t)$ es un operador unitario que satisface $U(t)^\dagger U(t) = I$, garantizando la conservación de la norma del estado (Nielsen y Chuang, 2010; Steane, 1998).

En contraste con la evolución reversible procedente de aplicar operadores unitarios o puertas cuánticas al sistema, la medición constituye un proceso fundamentalmente diferente: no es unitario, no es reversible y conlleva una pérdida de coherencia. Al realizar una medición, se proyecta el estado cuántico sobre uno de los vectores de una base ortonormal del espacio de Hilbert, típicamente la base computacional $\{|0\rangle, |1\rangle\}$. Según el postulado de la medida, si el sistema se encuentra en el estado:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \tag{2.7}$$

entonces el resultado de la medición será $|0\rangle$ con probabilidad $|\alpha|^2$ y $|1\rangle$ con probabilidad $|\beta|^2$. Tras la medición, el sistema colapsa al estado correspondiente al valor observado, eliminando cualquier superposición previa. Este proceso de colapso introduce una naturaleza estocástica e irreversible, contrastando con la evolución unitaria.

La medición cuántica también implica limitaciones fundamentales para el procesamiento de información, entre ellas:

- **Teorema de no clonación**: establece que no es posible construir un operador unitario que copie un estado cuántico arbitrario en otro (Wootters y Zurek, 1982).
- **Indeterminación y no determinación del estado**: dado un número finito de copias de un estado cuántico, no es posible determinar completamente su forma mediante mediciones. Este hecho se relaciona con el teorema de Holevo (1973), que limita la cantidad de información clásica que puede extraerse de un estado cuántico.

Estas características subrayan la diferencia radical entre la información cuántica y la información clásica. Mientras que en la computación clásica toda la información está completamente accesible y es determinista, en la computación cuántica el conocimiento del estado y su manipulación están profundamente restringidos por las leyes fundamentales de la mecánica cuántica.

### 2.4.4. Implementación física de ordenadores cuánticos

Con el paso del tiempo, estos avances en la teoría de los ordenadores cuánticos promovieron intentos de implementación física de cúbits mediante diferentes tecnologías, como cúbits basados en circuitos superconductores y trampas de átomos, entre otros (Kjaergaard et al., 2020; Monroe et al., 2021).

Los cúbits basados en superconductores son de los más usados actualmente, siendo empleados por empresas como IBM o Google (Kjaergaard et al., 2020). Estos cúbits son circuitos superconductores a temperaturas criogénicas que implementan uniones Josephson para utilizar el espectro de energías de un oscilador armónico donde los niveles de energía no están equispaciados. Una de las ventajas principales de este sistema es la facilidad de fabricación y la velocidad de aplicación de las puertas. Las desventajas están en la necesidad de obtener temperaturas criogénicas para mantener las propiedades superconductoras del material.

También se tienen ordenadores basados en trampa de iones. Este método consiste en atrapar iones con campos electromagnéticos. La información se podrá obtener a partir de la configuración electrónica de los átomos. Esta técnica resulta muy interesante, ya que tenemos grandes tiempos de decoherencia (Monroe et al., 2021). El problema se encuentra en la escalabilidad de estos sistemas, ya que los iones interaccionan mucho entre sí, de forma que al aumentar de forma significativa el número de átomos, dificulta la capacidad de tenerlos encerrados.

La computación cuántica se encuentra actualmente en lo que se conoce como la era NISQ (siglas del término anglosajón, *Noisy Intermediate-Scale Quantum*), un término acuñado por Preskill (2018) para describir dispositivos cuánticos que pueden realizar ciertas operaciones sin corrección total de errores, pero aún afectados significativamente por ruido y decoherencia.

Los sistemas cuánticos reales son inherentemente abiertos: interactúan de forma inevitable con su entorno, lo que produce pérdida de coherencia cuántica, también denominada decoherencia. La decoherencia destruye las correlaciones de fase entre componentes del estado cuántico y degrada las superposiciones y entrelazamientos necesarios para el cómputo cuántico útil.

Existen múltiples fuentes de ruido en plataformas cuánticas reales:

- **Relajación ($T_1$)**: transición espontánea del estado excitado al fundamental, que lleva a la pérdida de información del sistema.

- **Desfasamiento ($T_2$)**: pérdida de coherencia entre las fases relativas de un estado superpuesto.

- **Errores de compuerta**: desviaciones entre la operación unitaria ideal y la implementada experimentalmente.
- **Errores de lectura**: imprecisiones en el proceso de medición de los cúbits.

Estas fuentes de error limitan la profundidad de los circuitos cuánticos útiles. En ausencia de una corrección activa de errores, los algoritmos cuánticos deben mantener la suficiente profundidad como para ejecutarse antes de que la decoherencia destruya la información. Por esta razón, en la era NISQ se exploran algoritmos resilientes al ruido, como los llamados algoritmos variacionales, que combinan circuitos cuánticos poco profundos con optimización clásica. Asimismo, se desarrollan técnicas de mitigación de errores como extrapolación de ruido o aprendizaje automático, que buscan compensar los efectos del ruido sin necesidad de redundancia cuántica explícita (Temme et al., 2017).

El reto principal de la era NISQ es doble: por un lado, entender los límites computacionales de los dispositivos cuánticos ruidosos; y por otro, encontrar problemas de interés práctico en los que estos dispositivos puedan ofrecer una ventaja cuántica antes de que se logren arquitecturas con corrección de errores completa.

Si bien es cierto, como se verá más adelante, en este Trabajo se opta por una estrategia que puede no ser adecuada para esta era. Esto es una consecuencia de la complejidad necesaria para su implementación.

## 2.5. Algoritmos cuánticos para problemas en grafos

En esta sección se revisan tres enfoques para abordar problemas de optimización en grafos desde una perspectiva cuántica: computación adiabática, algoritmos variacionales y caminatas cuánticas. Estos modelos constituyen las bases teóricas y algorítmicas sobre las que se han desarrollado varias resoluciones a problemas pertenecientes a la teoría de grafos.

### 2.5.1. Computación adiabática

La computación adiabática es una de las principales vías escogidas para resolver problemas que pueden modelizarse como grafos utilizando notación QUBO (Gaitan y Clark, 2014; Stollenwerk et al., 2015; Hua, 2016; Dinneen y Hua, 2017). Por esta razón, siempre

que se pretende resolver un problema en teoría de grafos mediante computación cuántica, es una de las candidatas.

La computación cuántica adiabática se fundamenta en el teorema adiabático de la mecánica cuántica (Nielsen y Chuang (2010)): si un sistema cuántico se prepara inicialmente en el estado fundamental de un hamiltoniano $H_B$ y luego se hace evolucionar lo suficientemente despacio hacia un hamiltoniano problema $H_P$, el sistema permanecerá en su estado fundamental con alta probabilidad (Farhi et al., 2000; Aharonov et al., 2005). Formalmente, la interpolación temporal se escribe como:

$$H(s) = (1-s)H_B + sH_P, \quad s = \frac{t}{T}, \quad t \in [0, T] \tag{2.8}$$

El tiempo total $T$ debe escalar al menos como $\mathcal{O}(1/\Delta^2_{\text{mín}})$, donde $\Delta_{\text{mín}}$ es el gap espectral mínimo entre el estado fundamental y el primer estado excitado. Este enfoque es universalmente equivalente al modelo de circuito cuántico estándar, pero su rendimiento depende críticamente del tamaño del gap y de la presencia de decoherencia (Albash y Lidar, 2018; Farhi et al., 2000). El Hamiltoniano $H_P$ se diseña para codificar la solución del problema del camino más corto como su estado de energía mínima, mientras que $H_B$ permite una preparación inicial conocida y controlable. El éxito práctico de este método depende de la capacidad experimental de mantener la coherencia y controlar las condiciones adiabáticas.

La computación adiabática presenta una serie de ventajas y limitaciones que emergen tanto de fundamentos físicos como de restricciones prácticas de su implementación. Una de sus principales fortalezas radica en su formulación natural para problemas de optimización, ya que prescinde del uso de secuencias discretas de compuertas y se apoya, en cambio, en la evolución continua del estado fundamental de un sistema cuántico. Esta característica permite explotar la estructura de ciertos hamiltonianos cuya solución se encuentra codificada en el estado base, lo que puede conferir cierta robustez frente a errores típicos de las compuertas discretas.

No obstante, el rendimiento del algoritmo adiabático es extremadamente sensible al tamaño del gap espectral mínimo entre el estado fundamental y el primer estado excitado a lo largo de la evolución. Si este gap se cierra exponencialmente con el tamaño del problema, el tiempo necesario para garantizar una evolución adiabática se vuelve excesivamente largo.

Por otro lado, la presencia de decoherencia y el acoplamiento térmico con el entorno inducen transiciones no adiabáticas que afectan negativamente a la fidelidad del estado final. Finalmente, incluso si se alcanza un estado final estacionario, la verificación de que dicho estado corresponde efectivamente a la solución óptima global del problema planteado no es trivial y, en general, constituye una tarea difícil que no puede resolverse mediante una simple validación directa.

El estudio realizado por Padmasola y Chatterjee (2023) contribuye a la aplicación de esta metodología para la resolución del problema del camino más corto, que puede ser de gran interés como base de este Trabajo. Sin embargo, la complejidad asociada a la manipulación del hamiltoniano y la falta de control durante la evolución adiabática del sistema suponen un gran incoveniente, por lo que no ha sido la estrategia que se ha seleccionado para nuestra propuesta.

### 2.5.2. Algoritmos variacionales

Del mismo modo, los algoritmos variacionales son otra de las principales estrategias de resolución de problemas en grafos en computación cuántica. Se pueden modelizar problemas como el *MaxCut*, *k-SAT* o *Vertex Cover* (Farhi et al., 2014; Wang et al., 2018; Zhang et al., 2020; Li et al., 2021).

Son especialmente interesantes en el contexto de la era NISQ. Los algoritmos variacionales constituyen una clase prometedora de esquemas híbridos cuántico-clásicos, diseñados para mitigar los efectos del ruido y la decoherencia mediante circuitos cuánticos poco profundos, y delegando la optimización global del problema a métodos clásicos. Estos algoritmos aprovechan la capacidad del hardware cuántico para preparar estados complejos y calcular funciones objetivo difíciles de evaluar clásicamente, mientras que los parámetros de control se ajustan externamente con técnicas clásicas de optimización (Cerezo et al., 2021; Bharti et al., 2022).

- **Solucionador variacional cuántico de autovalores:** el solucionador variacional cuántico de autovalores (VQE, por sus siglas en inglés) fue propuesto inicialmente para estimar el valor propio más bajo (la energía del estado fundamental) de hamiltonianos cuánticos, con aplicaciones en química cuántica y física de materiales (Peruzzo et al., 2014; Qing y Xie, 2023). Dado un hamiltoniano del sistema $H_P$, se

construye una familia parametrizada de estados cuánticos:

$$|\psi(\theta)\rangle = U(\theta)\,|0\rangle^{\otimes n}\,, \tag{2.9}$$

donde $U(\theta)$ es un *ansatz* (circuito cuántico parametrizado), dependiente de un conjunto de parámetros clásicos $\theta \in \mathbb{R}^k$. La función de coste es la energía esperada del sistema:

$$E(\theta) = \langle\psi(\theta)|\,H_P\,|\psi(\theta)\rangle\,, \tag{2.10}$$

que se estima mediante mediciones cuánticas sobre los observables constituyentes de $H_P$. Un optimizador clásico ajusta iterativamente los parámetros $\theta$ para minimizar $E(\theta)$, convergiendo idealmente al valor mínimo de $H_P$. La calidad de los resultados depende en gran medida del diseño del ansatz.

- **Algoritmo cuántico de optimización aproximada:** en concreto, dentro de los algoritmos variacionales basados en el VQE, el más ampliamente usado es el Algoritmo cuántico de optimización aproximada (QAOA, por sus siglas en inglés). El QAOA, introducido por Farhi et al. (2014), es una técnica variacional especialmente estructurada para problemas de optimización combinatoria. Se ha demostrado que el algoritmo QAOA puede superar a los algoritmos clásicos en instancias del problema Max-Cut en grafos pequeños, particularmente 3-regulares, bajo condiciones ideales de muestreo y optimización de parámetros (Larkin et al., 2020; Lykov et al., 2023; Carlson y Kolla, 2023).

  Considerando un problema cuya función objetivo puede codificarse en un hamiltoniano de coste $H_P$, y un hamiltoniano mezclador $H_B$ que induce exploración del espacio de soluciones, el algoritmo aplica una secuencia alternada de operaciones:

$$U(\gamma,\alpha) = \prod_{j=1}^{p} e^{-i\gamma_j H_P} e^{-i\alpha_j H_B}, \tag{2.11}$$

  donde $p$ es la profundidad del algoritmo y $(\gamma,\alpha)$ son parámetros variacionales. Para $p = 1$, QAOA ya puede producir soluciones aproximadas no triviales, y se ha demostrado que su rendimiento mejora con $p$ bajo ciertas condiciones Zhou et al. (2018).

Los algoritmos variacionales constituyen una de las propuestas más prometedoras para aprovechar el potencial de los dispositivos cuánticos de la era NISQ. Entre sus principales

ventajas destaca el hecho de que requieren circuitos de profundidad relativamente baja, lo que los hace adecuados para arquitecturas cuánticas con tiempos de coherencia limitados. Además, permiten una gran flexibilidad en el diseño del *ansatz*, que puede adaptarse específicamente al hardware disponible, favoreciendo así una implementación eficiente.

Otra fortaleza significativa es su resiliencia frente a ciertos tipos de ruido, que puede mitigarse mediante técnicas específicas como métodos de extrapolación a ruido cero (Giurgica-Tiron et al., 2020). Sin embargo, estos algoritmos también enfrentan desafíos fundamentales. Uno de los más relevantes es la aparición de los denominados *barren plateaus* en el paisaje de optimización, regiones en las que el gradiente de la función de coste se vuelve exponencialmente pequeño, dificultando la convergencia del entrenamiento (McClean et al., 2018).

Además, el rendimiento de los algoritmos variacionales depende críticamente del diseño del *ansatz*: una elección inapropiada puede restringir severamente la exploración del espacio de soluciones relevantes. Finalmente, la carga computacional asociada al cálculo de gradientes puede escalar de manera poco favorable con la complejidad del problema abordado, lo que limita la eficiencia global del enfoque.

Al igual que para la computación adiabática, ya existen trabajos centrados en el problema del camino más corto con esta estrategia (Soltan Aghaei et al., 2008; Fan et al., 2023). Aunque en este caso si se permite el control durante la evolución del sistema, la codificación de una heurística tendría que introducirse en el hamiltoniano. Esto, pese a que puede ser una posible solución al problema, resulta en un gran aumento de la complejidad (restricciones adicionales en el hamiltoniano). Sea este el motivo para utilizar como estrategia en este Trabajo las caminatas cuánticas, pues, como se verá más adelante, la introducción de la heurística resulta más apropiada para este enfoque.

### 2.5.3. Caminatas cuánticas

Las caminatas cuánticas (*quantum walks* en inglés) constituyen una generalización cuántica de las caminatas aleatorias clásicas. Una caminata aleatoria clásica es una proceso estocástico en el que se produce un desplazamiento siguiendo una distribución de probabilidad definida (Lovász, 1993). Esto es, se realiza un recorrido siguiendo un camino aleatorio determinado por las probabilidades de transición a cada momento. Este conjunto de distribuciones de probabilidad es un caso concreto de cadenas de Markov. Las cadenas

de Markov (1906) son modelos matemáticos que sirven para describir procesos aleatorios paso a paso. En ellas, el estado futuro de un sistema depende únicamente del estado actual, sin importar los estados previos en los que se ha encontrado el sistema (Norris, 1998).

Las caminatas cuánticas conservan la estructura conceptual de las cadenas de Markov y las caminatas aleatorias clásicas, pero la expanden con propiedades cuánticas como la superposición y la interferencia. Esto habilita fenómenos como la propagación paralela y la amplificación selectiva de trayectorias, que constituyen la base de su potencial algorítmico y de sus aplicaciones en la simulación de procesos físicos. Mientras que en el marco clásico la evolución está gobernada por una matriz de transición estocástica, en el ámbito cuántico ésta se reemplaza por un operador unitario, que garantiza que la evolución sea reversible, coherente y conserve la norma del estado (Kempe, 2003). El paso fundamental de la descripción clásica a la cuántica consiste en sustituir las probabilidades por amplitudes de probabilidad complejas. El estado ya no se representa mediante una distribución probabilística sobre los vértices de un grafo, sino mediante una superposición cuántica de estados. Este cambio de marco introduce un recurso ausente en los paseos clásicos: la interferencia. A medida que las amplitudes se propagan por el grafo, pueden reforzarse unas a otras o cancelarse. Esta característica hace que la dinámica de las caminatas cuánticas exhiba patrones de propagación distintos a las caminatas clásicas (Venegas-Andraca, 2012). Es esta diferente dinámica la que explotan los diferentes algoritmos basados en caminatas cuánticas. Gracias a estas propiedades, las caminatas cuánticas se han consolidado como un marco versátil tanto para el diseño de algoritmos cuánticos como para la modelización de procesos físicos, tales como la difusión cuántica o el transporte de energía en sistemas discretos (Childs et al., 2002; Mohseni et al., 2008).

Existen dos formulaciones principales de caminatas cuánticas: la versión en tiempo continuo, gobernada por un hamiltoniano asociado al grafo, y la versión en tiempo discreto, en la que la evolución se construye a partir de operadores de moneda (en inglés, *coin*) y desplazamiento (en inglés, *shift*). Ambas comparten la misma idea general, aunque difieren en su descripción matemática y en las aplicaciones en las que resultan más adecuadas. A continuación, se explican con mayor detalle ambas versiones:

- **Caminatas cuánticas en tiempo continuo:** se introdujeron como una analogía directa con la dinámica de un sistema cuántico gobernado por un hamiltoniano. Sea un grafo no dirigido $G = (V, E)$ con $|V|$ vértices. Asociamos a cada vértice $v \in V$ un

estado base $|v\rangle$. El espacio de Hilbert de la caminata es entonces:

$$\mathcal{H} = \text{span}\{|v\rangle : v \in V\} \cong \mathbb{C}^{|V|}. \tag{2.12}$$

La dinámica viene determinada por un hamiltoniano $H$ que codifica la conectividad del grafo. Una elección común es la matriz de adyacencia:

$$H = A, \quad A_{uv} = \begin{cases} p_{uv} & \text{si } (u,v) \in E, \\ 0 & \text{en otro caso,} \end{cases} \tag{2.13}$$

donde $p_{uv}$ determina la probabilidad de transición entre los vértices correspondientes. La elección de $H$ determina las propiedades de propagación de la caminata (Farhi y Gutmann, 1998). La evolución temporal obedece la ecuación de Schrödinger dependiente del tiempo:

$$i\frac{d}{dt}|\psi(t)\rangle = H|\psi(t)\rangle, \tag{2.14}$$

cuya solución formal es:

$$|\psi(t)\rangle = U(t)|\psi(0)\rangle, \quad U(t) = e^{-iHt}. \tag{2.15}$$

El estado en cualquier instante puede escribirse como una superposición donde los coeficientes son las amplitudes de probabilidad asociadas a cada vértice. Estas amplitudes evolucionan de manera coherente, generando patrones de interferencia característicos de la caminata cuántica.

Este modelo resulta particularmente natural en contextos físicos, ya que reproduce fenómenos de propagación cuántica, como en el transporte de excitaciones en materiales o en fotosíntesis artificial (Mohseni et al., 2008).

Desde el punto de vista algorítmico, se ha mostrado que las caminatas en tiempo continuo permiten acelerar problemas de búsqueda en grafos, logrando ventajas exponenciales frente a caminatas clásicas en ciertos escenarios (Childs et al., 2003).

- **Cáminatas cuánticas en tiempo discreto:** en contraste, las caminatas en tiempo discreto se definen a través de la aplicación iterada de un operador unitario compuesto. Dado que un grafo puede tener vértices de distinto grado, no es suficiente con asociar un estado $|v\rangle$ a cada vértice: es necesario añadir un espacio moneda que

determine la dirección del movimiento (Ambainis, 2003). Así, el espacio de Hilbert total es entonces:

$$\mathcal{H} = \mathcal{H}_{\text{coin}} \otimes \mathcal{H}_{\text{pos}}, \tag{2.16}$$

donde $\mathcal{H}_{\text{pos}}$ codifica los vértices del grafo y $\mathcal{H}_{\text{coin}}$ las posibles transiciones. La evolución a cada paso se implementa en dos fases. En primer lugar, el operador moneda $C$ actúa sobre $\mathcal{H}_{\text{coin}}$, generando una superposición de direcciones. En segundo lugar, un operador de desplazamiento $S$ traslada las amplitudes a lo largo de las aristas del grafo en función de la dirección determinada por la moneda. El operador unitario de un paso completo se escribe como:

$$U = S \cdot (C \otimes I), \tag{2.17}$$

donde $S$ es el operador de desplazamiento e $I$ la identidad en el espacio de posiciones. Esta notación representa la aplicación del operador moneda sobre las posiciones iniciales, sin alterarlas, para su posterior modificación a través del operador desplazamiento.

Esta versión ha dado lugar a múltiples algoritmos cuánticos, entre ellos el algoritmo de búsqueda de Shenvi–Kempe–Whaley (Shenvi et al., 2003), que logra acelerar tareas de búsqueda no estructurada en comparación con métodos clásicos.

Una variante especialmente relevante dentro del modelo discreto es la caminata cuántica de tipo Szegedy, entendida como la generalización cuántica de una cadena de Markov clásica (Szegedy, 2004). Su formalismo es el siguiente.

Sea $P$ una matriz de transición estocástica asociada a una cadena de Markov sobre un conjunto de $N$ estados. Se define un espacio de Hilbert doble:

$$\mathcal{H} = \mathbb{C}^N \otimes \mathbb{C}^N, \tag{2.18}$$

cuyos estados base son $|x\rangle \otimes |y\rangle$, que representan una transición potencial de $x$ a $y$.

A partir de $P$, se construyen vectores normalizados que codifican las distribuciones de transición de cada estado:

$$|\psi_x\rangle = \sum_{y=1}^{N} \sqrt{p_{xy}}\ |x\rangle \otimes |y\rangle\,. \tag{2.19}$$

donde $p_{xy}$ representa la probabilidad de transición de $x$ a $y$. Sea:

$$\mathcal{A} = \text{span}\{|\psi_x\rangle : x = 1, \ldots, N\}, \tag{2.20}$$

el subespacio generado por estos estados. El operador de reflexión asociado es:

$$R = 2\Pi_{\mathcal{A}} - I, \tag{2.21}$$

donde $\Pi_{\mathcal{A}}$ es el proyector sobre $\mathcal{A}$[1]. De manera análoga, se construye un segundo subespacio $\mathcal{B}$, generado por los vectores:

$$|\phi_y\rangle = \sum_{y=1}^{N} \sqrt{p_{xy}}\ |y\rangle \otimes |x\rangle\,, \tag{2.22}$$

y su correspondiente reflexión:

$$R' = 2\Pi_{\mathcal{B}} - I. \tag{2.23}$$

El operador unitario de Szegedy se define entonces como la composición de ambas reflexiones:

$$U_P = R'R. \tag{2.24}$$

Este operador es unitario por construcción y captura de forma cuántica la dinámica inducida por la cadena de Markov (Nielsen y Chuang, 2010). Este operador unitario actúa como doble operador de una caminata cuántica discreta. Generalmente, estos operadores de reflexión son los escogidos para representar las transiciones del sistema a causa de la falta de unitariedad de la matriz $P$.

En resumen, las caminatas en tiempo continuo se apoyan directamente en la dinámica hamiltoniana de la mecánica cuántica y resultan naturales para modelar procesos físicos de transporte. Las caminatas en tiempo discreto, en cambio, introducen un grado adicional de flexibilidad mediante el uso de monedas cuánticas y operadores de desplazamiento, lo que facilita su aplicación en el diseño de algoritmos. Dentro de estas últimas, el modelo de Szegedy destaca por su capacidad de establecer una similitud estructural con las cadenas de Markov clásicas, permitiendo una conversión de técnicas probabilísticas clásicas al contexto cuántico.

[1] El proyector sobre $\mathcal{A}$ se define como: $\Pi_{\mathcal{A}} = \mathcal{A}\mathcal{A}^\dagger$ (Cohen-Tannoudji et al., 1977).

Los trabajos de Montanaro (2018) y Li (2025) son la base para la idea de un algoritmo cuántico inspirado en A* mediante caminatas cuánticas. Montanaro mostró cómo transformar algoritmos de *backtracking* en versiones cuánticas más, eficientes usando caminatas discretas; mientras que Li demostró ventajas exponenciales en problemas de búsqueda de caminos sobre grafos estructurados. Ambos resultados consolidan la motivación de explorar caminatas cuánticas como núcleo de un algoritmo de tipo A*.

Siguiendo todo lo expuesto con anterioridad, se hace notable la practicidad del uso de caminatas cuánticas para el problema que se pretende resolver en este Trabajo. Es por esto que la línea de investigación que se ha decidido seguir busca combinar el uso de caminatas cuánticas y la introducción de una heurística, al igual que hace A*, para resolver el problema del camino más corto.

# 3. Objetivos

Este Trabajo tiene como propósito fundamental sentar las bases teóricas para el desarrollo de una versión cuántica del algoritmo A*, explorando las posibilidades que ofrece la computación cuántica para reducir su complejidad y mejorar su eficiencia. A partir del análisis de los principios cuánticos y de las limitaciones del algoritmo clásico, se busca construir un marco conceptual una implementación cuántica del A*, resaltando así el potencial de la computación cuántica para enfrentar problemas de búsqueda y planificación de manera más eficiente.

El enfoque adoptado pretende vincular los avances recientes en algoritmos cuánticos con las necesidades prácticas de optimización que resuelve el algoritmo A*. A través de una investigación teórica, se espera identificar los elementos clave que conformarán la base para una contribución inicial en la dirección de la creación de un algoritmo cuántico con un funcionamiento similar al A*.

## 3.1. Objetivos generales

- Explorar la posibilidad de diseñar un algoritmo A* cuántico que, aprovechando los principios fundamentales de la computación cuántica, pueda superar en eficiencia a su versión clásica.
- Fundamentar teóricamente el uso de técnicas cuánticas en problemas de búsqueda y planificación.

## 3.2. Objetivos específicos

- Analizar el funcionamiento y la estructura del algoritmo A*, incluyendo su comportamiento heurístico y limitaciones principales.
- Investigar los fundamentos teóricos de la computación cuántica relevantes para algoritmos de búsqueda, con énfasis en fenómenos como la superposición, el entrelazamiento y la interferencia. Asimismo, estudiar algoritmos cuánticos existentes relacionados con búsqueda y optimización, tales como Grover o las caminatas cuánticas.
- Identificar un esquema conceptual para un algoritmo A* cuántico que defina sus componentes, comportamiento esperado y supuestos necesarios.

# 4. Desarrollo del Trabajo

Como se ha mencionado anteriormente, el objetivo de este Trabajo es presentar un algoritmo cuántico, basado en el algoritmo A* y haciendo uso de caminatas cuánticas. Esto con el fin de resolver el problema del camino de menor coste en grafos ponderados. Este enfoque se apoya en la capacidad de las caminatas cuánticas para explorar de manera simultánea un gran número de trayectorias, modulando la exploración para concentrar amplitud sobre las soluciones más prometedoras.

A lo largo de esta Sección 4 se presentarán en detalle las condiciones bajo las cuales este algoritmo puede aplicarse de manera rigurosa. Posteriormente, se describirán de forma separada sus componentes principales: la caminata cuántica de Szegedy; el kernel de transición[1] basado en distribuciones de Boltzmann-Gibbs; la posible aplicación de la transformada de Doob como heurística; la incorporación de registros auxiliares para el camino y el contador de longitud; el mecanismo de parada (en inglés, *halt*).

Finalmente, se mostrará cómo estos elementos se integran para implementar el algoritmo al completo, justificando las elecciones del diseño.

## 4.1. Condiciones de aplicabilidad

El algoritmo propuesto no es universalmente aplicable a cualquier tipo de grafo, sino que requiere el cumplimiento de una serie de condiciones estructurales. El punto de partida es un grafo no dirigido, finito, conexo y ponderado, cuyas aristas están asociadas a costes estrictamente positivos. Estas restricciones son necesarias para asegurar que la matriz de transición derivada de los costes puede interpretarse adecuadamente para un problema definible mediante cadenas de Markov (Nielsen y Chuang, 2010). Esta imposición se debe a, como se verá más adelante, la selección de caminatas cuánticas de Szegedy como componente principal del algoritmo.

Una vez establecida la naturaleza de los grafos problema, se presentan los diferentes elementos que componen el algoritmo.

[1] Cuando se habla de kernel de transición en el contexto de cadenas de Markov, se trata de una función que describe la probabilidad de transición entre dos estados.

## 4.2. Componentes individuales del algoritmo

En esta sección 4.2 se describen de manera detallada los elementos que componen el algoritmo propuesto. Cada componente se presenta de forma independiente, destacando su función, formulación matemática y la justificación de su incorporación.

### 4.2.1. Caminata cuántica de Szegedy

El núcleo del algoritmo es la cuantización de Szegedy (2004) de una cadena de Markov reversible. El formalismo asociado a esta metodología es el siguiente. Dados un conjunto de vértices $V$ y una matriz de transición $P$ sobre ellos, se define el espacio de Hilbert:

$$\mathcal{H}_E = \mathbb{C}^{|V|} \otimes \mathbb{C}^{|V|}, \tag{4.1}$$

tal que:

$$\mathcal{H}_E = \text{span}\{|x, y\rangle : (x, y) \in E\}, \tag{4.2}$$

donde $E$ representa el conjunto de aristas del grafo y los estados base $|x\rangle$ e $|y\rangle$ representan transiciones del nodo $x$ al nodo $y$. Para cada vértice $x$, se construyen los estados:

$$|\psi_x\rangle = \sum_{y \in \Gamma(x)} \sqrt{p_{xy}}\, |x\rangle\, |y\rangle\,. \tag{4.3}$$

donde $\Gamma(x)$ es el conjunto de nodos conectados al nodo $x$ y $p_{xy}$ es el elemento $P(x, y)$ de la matriz de transición y determina la probabilidad de ir del nodo $x$ al nodo $y$, también denominado kernel de transición.

El subespacio $\mathcal{A} = \text{span}\{|\psi_x\rangle : x \in V\}$ genera la reflexión:

$$R = 2\Pi_{\mathcal{A}} - I, \qquad \Pi_{\mathcal{A}} = \sum_x |\psi_x\rangle\langle\psi_x|\,. \tag{4.4}$$

Este operador $R$ constituye el operador moneda característico de una caminata cuántica discreta, esto es, $R \equiv C$. Asimismo, el operador de desplazamiento utilizado en la equivalencia con Szegedy es el llamado en inglés *flip–flop shift*:

$$S\,|x\rangle\,|y\rangle = |y\rangle\,|x\rangle\,, \tag{4.5}$$

que intercambia los dos registros y, por tanto, invierte la orientación de la arista.

Existe también otro desplazamiento de común uso en caminatas cuánticas discretas, en inglés, *moving shift*. Definido como:

$$S_{\text{mov}}\,|x\rangle\,|d\rangle = |y\rangle\,|d\rangle\,, \tag{4.6}$$

donde $d$ denota la dirección elegida desde $x$ hacia $y$. En este caso, el desplazamiento nunca cambia de dirección, siendo especialmente útil para problemas con cierta estructura lineal. Sin embargo, en los grafos problema esta estructura no necesariamente se cumple. Además la equivalencia con Szegedy se rompe, pues $S_{\text{mov}}$ no implementa la simetría necesaria para reproducir $R'$. Por ello, cuando se habla estrictamente de la caminata de Szegedy, se utiliza siempre el *flip–flop shift* y es el implementado en este Trabajo.

Una propiedad fundamental del formalismo de Szegedy es que la reflexión $R'$ asociada al subespacio:

$$\mathcal{B} = \text{span}\left\{|\phi_x\rangle : y \in V\right\}, \qquad |\phi_x\rangle = \sum_y \sqrt{p_{xy}}\,|y\rangle\,|x\rangle\,, \tag{4.7}$$

puede implementarse a partir de la reflexión $R$ y del operador de desplazamiento *flip–flop shift*. Nótese en primer lugar que los estados $|\phi_x\rangle$ se obtienen aplicando $S$ sobre los estados $|\psi_x\rangle$:

$$S\,|\psi_x\rangle = S\left(\sum_y \sqrt{p_{xy}}\,|x\rangle\,|y\rangle\right) = \sum_y \sqrt{p_{xy}}\,|y\rangle\,|x\rangle\,. \tag{4.8}$$

Por ende, $\mathcal{B} = S\mathcal{A}$. Sea $\Pi_{\mathcal{A}}$ el proyector sobre $\mathcal{A}$ y $\Pi_{\mathcal{B}}$ el proyector sobre $\mathcal{B}$. Cumpliendo la igualdad anterior, se obtiene[2]:

$$\Pi_{\mathcal{B}} = S\Pi_{\mathcal{A}}S. \tag{4.9}$$

En consecuencia, el operador de reflexión sobre $\mathcal{B}$ es:

$$R' = 2\Pi_{\mathcal{B}} - I = 2S\Pi_{\mathcal{A}}S - I. \tag{4.10}$$

Esto se puede reescribir como:

$$R' = S(2\Pi_{\mathcal{A}} - I)S = SRS. \tag{4.11}$$

De esta manera se establece que $R'$ no es independiente de $R$, sino que se obtiene mediante la combinación de $R$ con el *flip–flop shift*. El operador completo de Szegedy puede entonces escribirse como:

$$U = R'R = (SRS)R, \tag{4.12}$$

lo que muestra explícitamente que cada paso de Szegedy equivale a la aplicación de dos pasos en una caminata cuántica discreta (Wong, 2017b).

[2]Téngase en cuenta que el operador $S$ es hermítico: $S = S^\dagger$

La caminata cuántica de Szegedy se ha seleccionado como metodología debido a que los grafos problema pueden modelarse de manera natural como cadenas de Markov reversibles. Este formalismo proporciona un marco adecuado para la cuantización de dichas cadenas, garantizando un operador unitario bien definido. Una ventaja fundamental es que la dinámica inducida por Szegedy permite explorar de manera coherente la totalidad del grafo en superposición cuántica, preservando la estructura estocástica subyacente e incorporando la posibilidad de la propagación de la amplitud en las trayectorias (Portugal, 2016a,b; Wong, 2017a; Portugal y Segawa, 2017).

### 4.2.2. Kernel de transición

El kernel de transición constituye el pilar fundamental a partir del cual se construye la caminata de Szegedy. Esta función determina la probabilidad de transición a partir de los costes presentes en el grafo. Su definición para este caso consta de lo siguiente. Dado un grafo ponderado $G = (V, E)$ con costes positivos $w_{xy} > 0$ en cada arista $(x, y) \in E$, se define una cadena de Markov reversible cuya matriz de transición $P$ asigna a cada vértice $x$ una distribución de probabilidad sobre sus vecinos siguiendo[3]:

$$P(x, y) = \frac{e^{-\beta_w w_{xy}}}{\sum_{z \in \Gamma(x)} e^{-\beta_w w_{xz}}}, \quad y \in \Gamma(x); \tag{4.13}$$

y $P(x, y) = 0$ en caso contrario, esto es, si a $x$ e $y$ no los une ninguna arista.

Esta construcción se inspira directamente en la distribución de Boltzmann–Gibbs de la mecánica estadística, donde la probabilidad de un microestado con energía $E$ está ponderada como $e^{-\beta E}$ (Boltzmann, 1970; Gibbs, 2010). En el contexto de grafos, los costes $w_{xy}$ juegan el papel de energías, y el parámetro $\beta_w$ corresponde al inverso de una temperatura efectiva (Wocjan et al., 2009). Se verá más adelante como este parámetro $\beta$ sirve para modular un sesgo según lo requiera el problema.

Esta estrategia es común en simulaciones clásicas (Metropolis et al., 1953; Goldstein et al., 2020) y se ha demostrado que preparar coherentemente este muestreo con Szegedy da una ventaja cuadrática frente al mezclado clásico (Somma et al., 2008), lo que justifica la selección de esta función.

Por su parte, el parámetro $\beta_w$ introduce un control explícito sobre el equilibrio entre

[3]Nótese la igualdad mencionada previamente: $P(x, y) = p_{xy}$

exploración completa del grafo y explotación de los caminos óptimos de propagación de la probabilidad. El valor de $\beta_w$ afecta a la distribución de forma que se cumple:

- **Cuando $\boldsymbol{\beta_w \to 0}$**, se obtiene:

$$P(x, y) \approx \frac{1}{\deg(x)}, \tag{4.14}$$

  es decir, una distribución casi uniforme sobre los vecinos de $x$. La distribución perfectamente uniforme se alcanza en el valor $\beta_w = 0$. En este régimen la caminata explora el grafo de forma amplia, con poca discriminación en función de los costes. Este sería el enfoque utilizado si el grafo problema no presentase ponderaciones.

- **Para valores intermedios de $\boldsymbol{\beta_w}$**, la caminata favorece aristas de menor coste, pero aún permite explorar transiciones subóptimas. Este comportamiento es análogo al de los algoritmos clásicos de enfriamiento simulado (*simulated annealing*), donde se busca explorar todas las posibles rutas manteniendo un mayor foco en las soluciones más prometedoras (Kirkpatrick et al., 1983).

- **Cuando $\boldsymbol{\beta_w \to \infty}$**, la probabilidad se concentra en la arista de menor coste desde cada vértice. En el límite, la caminata se aproxima a un proceso casi determinista siguiendo siempre la transición de menor coste local.

Idealmente, un valor muy grande de $\beta_w$ es el que maximizaría la probabilidad de seleccionar siempre el camino de coste mínimo. Sin embargo, este escenario presenta inconvenientes. El operador de Szegedy se vuelve extremadamente sesgado, y la amplitud inicial en ramas subóptimas puede ser casi nula, lo que limita la capacidad de explorar todas las ramas derivando en resultados potencialmente inadecuados. Así, la caminata se aproxima a un comportamiento clásico y determinista, desaprovechando las ventajas cuánticas de exploración en superposición.

En la práctica, por tanto, no resulta óptimo elegir $\beta_w$ excesivamente grande, sino modularlo según el problema. De manera que exista suficiente sesgo hacia las aristas de menor coste para que los caminos óptimos acumulen amplitud apreciable y, al mismo tiempo, se mantenga una superposición adecuada de suficiente cantidad trayectorias. En consecuencia, el valor de $\beta_w$ debe calibrarse teniendo en cuenta tanto la estructura del grafo como los recursos computacionales disponibles.

Por otro lado, la definición del kernel implica una normalización local en cada vértice $x$:

$$Z_x(\beta_w) = \sum_{z\in\Gamma(x)} e^{-\beta_w w_{xz}}. \tag{4.15}$$

Este factor de partición depende del grado de $x$ y de los costes asociados a sus aristas. A causa de esto, la probabilidad acumulada de un camino completo $\gamma = (x_0, x_1, \ldots, x_k)$ resulta de un producto de cocientes locales:

$$\Pr(\gamma;\beta_w) = \prod_{t=0}^{k-1} \frac{e^{-\beta_w w_{x_t x_{t+1}}}}{Z_{x_t}(\beta)}. \tag{4.16}$$

Así, incluso cuando un camino tiene coste total bajo, puede perder peso relativo si atraviesa nodos con alto grado, debido al aumento de los valores $Z_{x_t}(\beta_w)$. Este fenómeno introduce lo que se puede entender como una "dilución"de la probabilidad. Es esta la razón que motiva la posterior introducción de la transformada de Doob; que al mismo tiempo funcionará como heurística, basándose en el funcionamiento de A*.

En la caminata cuántica de Szegedy, el kernel determina los estados:

$$|\psi_x(\beta_w)\rangle = \sum_{y\in\Gamma(x)} \sqrt{p_{xy}}\,|x\rangle\,|y\rangle\,, \tag{4.17}$$

que se utilizan en la definición de las reflexiones $R$ y $R'$. La dinámica resultante depende explícitamente de $\beta_w$, y por tanto el operador unitario de Szegedy se convierte en

$$U(\beta_w) = R'(\beta_w)R(\beta_w). \tag{4.18}$$

De esta forma, la probabilidad final de éxito depende en gran medida del valor de $\beta_w$.

### 4.2.3. Transformada de Doob

Uno de los principales problemas del kernel utilizado en las caminatas cuánticas es la "dilución"de probabilidad en grafos con alto grado de ramificación: aunque un camino pueda ser globalmente barato, la probabilidad asociada a recorrerlo disminuye drásticamente si atraviesa nodos con muchos vecinos, ya que la normalización local:

$$Z_x(\beta_w) = \sum_{z\in\Gamma(x)} e^{-\beta_w w_{xz}} \tag{4.19}$$

aumenta y reduce el peso relativo de cada transición. Este efecto dificulta que el algoritmo concentre amplitud en el camino de menor coste.

Para corregir este sesgo, se recurre a la transformada de Doob (1957), introducida originalmente en el contexto de procesos estocásticos condicionados y utilizada más recientemente en algoritmos cuánticos de muestreo y optimización (Chetrite y Touchette, 2014). Su uso en cadenas de Markov está ampliamente estudiado (Faggionato et al., 2009), por lo que su uso en este caso resulta muy oportuno.

La transformada de Doob permite modificar el kernel de transición de forma que se discriminen positivamente las trayectorias que llevan hacia un cierto objetivo, reponderando la dinámica en función de un potencial armónico $h : V \to \mathbb{R}^+$. En el contexto de grafos, un potencial se denomina armónico si satisface una condición de promedio local sobre los vecinos. En particular, para un vértice $x$ distinto del nodo objetivo, se requiere que:

$$h(x) = \sum_{y \in \Gamma(x)} P(x, y)\, h(y), \tag{4.20}$$

donde $\Gamma(x)$ denota el conjunto de vecinos de $x$. Esta relación significa que el valor de $h(x)$ en un nodo intermedio coincide con el valor esperado de $h$ al moverse de $x$ hacia sus vecinos según las probabilidades de transición $P(x, y)$. En consecuencia, el potencial $h$ se comporta como una función dependiente respecto a la dinámica de la caminata (Levin et al., 2008). Transmitiendo así la información sobre la proximidad al nodo objetivo a lo largo de la estructura del grafo.

En este caso, este potencial se ajusta escogiendo:

$$h(x) = e^{-\beta_h d(x, x_\star)}, \tag{4.21}$$

donde $d(x, x_\star)$ es una estimación admisible del coste restante hasta el objetivo $x_\star$, y $\beta_h$ es un parámetro que realiza la misma función que $\beta_w$ sobre las estimaciones, en lugar de sobre los pesos de los caminos. Así, aunque $h$ no es estrictamente armónico respecto al kernel original $P$, desempeña un papel similar: sesga la caminata hacia los estados que minimizan el coste futuro estimado, de la misma forma en que una función armónica en el caso clásico guía las trayectorias hacia los conjuntos objetivos. Este diseño hace que la transformada de Doob funcione como una forma de heurística similar a la que se utiliza en el algoritmo A*. Se consigue de este modo reequilibrar la dinámica que sigue la caminata.

Este diseño conecta directamente con la filosofía del algoritmo clásico A* (Hart et al., 1968), donde la heurística no necesita reproducir fielmente el valor exacto de la distancia restante, sino proporcionar una cota admisible que dirija la búsqueda. De forma análoga,

en la caminata cuántica la elección heurística de $h$ permite balancear la propagación, reduciendo la "dilución"de la amplitud en ramificaciones poco prometedoras y concentrando probabilidad en los caminos más relevantes hacia el objetivo. Así, incluso sin armonicidad exacta, el planteamiento sigue siendo válido y útil para resolver el problema.

En principio, sería posible definir una función $h$ estrictamente armónica, que correspondería a la probabilidad exacta de alcanzar el nodo objetivo. Sin embargo, calcular dicha función requiere resolver un sistema global de ecuaciones en el grafo, lo cual es tan costoso como el propio problema que se pretende resolver. Por este motivo se opta por una aproximación heurística, más sencilla de computar, que aunque no sea armónica reproduce el efecto deseado de guiar la caminata hacia el objetivo.

El nuevo kernel se define como:

$$P^h(x,y) = \frac{P(x,y)\,h(y)}{\sum_{z\in\Gamma(x)} P(x,z)\,h(z)}, \tag{4.22}$$

que es igualmente estocástico y reversible siempre que $P$ lo sea. La interpretación es tal que las probabilidades de transición originales $P(x,y)$ se reescalan por el factor $h(y)$, de manera que se discriminan positivamente los vecinos $y$ con mayor potencial $h$. La introducción del denominador asegura que se mantenga la normalización local.

Aplicar la transformada de Doob significa que los estados de moneda de Szegedy se definen ahora a partir de $P^h$:

$$|\psi_x^h(\beta_w,\beta_h)\rangle = \sum_{y\in V} \sqrt{P^h(x,y)}\,|x\rangle\,|y\rangle\,. \tag{4.23}$$

El operador unitario asociado a la caminata pasa a depender de $h$:

$$U^h(\beta_w,\beta_h) = R'^h(\beta_w,\beta_h)R^h(\beta_w,\beta_h). \tag{4.24}$$

En consecuencia, la dinámica cuántica refleja no solo la estructura estocástica original, sino también la información heurística codificada en $h$.

La utilidad de esta construcción es doble. Por un lado, reduce el impacto de la "dilución"de probabilidad, redistribuyendo amplitud hacia trayectorias que se acercan al objetivo según la heurística. Por otro lado, permite controlar de manera más efectiva la exploración en función de los parámetros $\beta_w$ y $\beta_h$: incluso con valores moderados de $\beta_h$, la transformada de Doob dirige la caminata hacia regiones relevantes del grafo, evitando la necesidad de llevar $\beta_w \to \infty$. Situación que, como se explicó previamente, lleva asociados algunos inconvenientes.

### 4.2.4. Registros adicionales: camino y contador de longitud

En el formalismo estándar de la caminata de Szegedy, la caminata cuántica solo proporciona información acerca del vértice alcanzado al final de la evolución. Sin embargo, en este algoritmo resulta necesario poder conocer el camino completo seguido desde el vértice inicial hasta el objetivo. Para ello se introducen registros adicionales que almacenan cada una de las trayectorias tomadas. Un registro cuántico que guarda todas las posibles trayectorias en superposición, y un registro clásico que sirve como control para la escritura de la información a cada iteración.

Además, para asegurar que todos los caminos simples desde el origen hasta el objetivo pueden explorarse, se fija un parámetro $L_{\text{máx}}$, tal que $L_{\text{máx}} = |V|-1$. Este nuevo parámetro representa la cota superior para el número de desplazamientos necesarios para explorar todo el grafo. Dado que el operador de Szegedy implementa dos pasos en cada instancia, el número de iteraciones de dicho operador será la mitad del valor de $L_{\text{máx}}$[4].

Los dos registros adicionales son: el registro camino (en inglés, *path*) y el registro contador de longitud $\ell$.

- **Registro camino:** este registro se compone de un espacio de Hilbert:

$$\mathcal{H}_{path} = \bigotimes_{j=0}^{L_{\text{máx}}} \mathbb{C}^{|V|}, \tag{4.25}$$

  cuyos elementos guardan la información de los vértices visitados en cada paso de la caminata. Un estado del registro tiene la forma:

$$|path\rangle = |x_0\rangle \otimes |x_1\rangle \otimes \cdots \otimes |x_{L_{\text{máx}}}\rangle, \tag{4.26}$$

  donde $x_0$ es el vértice inicial y los valores subsiguientes $x_j$ representan las transiciones seguidas. Inicialmente:

$$|path_0\rangle = |x_0\rangle \otimes |\varnothing\rangle \otimes \cdots \otimes |\varnothing\rangle, \tag{4.27}$$

  donde el símbolo $\varnothing$ indica que el elemento aún no ha sido marcado. El registro se actualiza en cada paso de la caminata mediante una operación controlada por el con-

[4]Para valores de $L_{\text{máx}}$ impares, puede implementarse la mitad del último operador de Szegedy para tener la cantidad de pasos exactos o bien aplicarlo por completo, ya que la introducción de un paso extra no modifica notablemente el funcionamiento del algoritmo más alla de la complejidad computacional añadida.

tador $\ell$, que copia el vértice actual $x_t$ de la caminata en la posición correspondiente:

$$U_{path,\ell} : |x_t\rangle |path\rangle \mapsto |x_t\rangle |x_0, \ldots, x_{\ell-1}, x_t, \varnothing, \ldots\rangle \tag{4.28}$$

De esta forma, cada trayectoria en superposición queda almacenada de manera coherente en el registro camino. Nótese que este operador actúa con respecto al registro que representa la posición actual en el grafo. Esto es, siguiendo la notación utilizada, el primer registro de los registros del espacio de Hilbert doble que representan los nodos y sus vecinos.

- **Registro contador de longitud:** el registro contador $\ell$ se implementa como un registro clásico compuesto por $L_{\text{máx}}$ bits bajo codificación en caliente (en inglés, *one-hot*) (Harris y Harris, 2012). El registro contador sirve como un conjunto de bits clásicos tal que:

$$\ell = (y_1, y_2, y_3, \ldots, y_{L_{\text{máx}}}) \tag{4.29}$$

En este tipo de codificación, exactamente un bit toma el valor 1 mientras que el resto son 0. Así, el valor del contador para $\ell = 1$ se representa como:

$$\ell = (1, 0, 0, \ldots, 0), \tag{4.30}$$

el valor $\ell = 2$ como

$$\ell = (0, 1, 0, \ldots, 0), \tag{4.31}$$

y en general, para el valor $\ell = j$ como

$$\ell = e_j = (0, \ldots, 0, y_j = 1, 0, \ldots, 0), \tag{4.32}$$

donde el único 1 aparece en la posición $j$.

Inicialmente el registro se prepara en $e_1$ ($\ell = 1$), indicando que la primera escritura en el registro camino se hará en la posición de índice 1. El avance del contador se define mediante la operación determinista

$$U_\ell : e_j \mapsto e_{j+1}, \qquad 1 \leqslant j < L_{\text{máx}}, \tag{4.33}$$

que desplaza el bit activo a la posición inmediatamente siguiente. En la práctica, esta operación puede implementarse como una permutación de bits.

La dinámica completa de un paso de la caminata con la implementación de estos dos registros consiste entonces en la composición de tres operaciones, dos en el ámbito cuántico y una en el clásico:

1. Aplicación de la moneda y el desplazamiento sobre el registro de aristas ($SR$).
2. Escritura del vértice actual en registro camino controlada por el contador ($U_{path,\ell}$).
3. Incremento del contador ($U_\ell$).

Formalmente, si en el paso $t$ el sistema se encuentra en:

$$|x_t, y_t\rangle \otimes |x_0, \ldots, x_{t-1}, \varnothing, \ldots\rangle, \tag{4.34}$$

tras un paso completo de la caminata se obtiene

$$\sum_y \sqrt{P(x_t, y)}\ |y, x_t\rangle \otimes |x_0, \ldots, x_{t-1}, y, \varnothing, \ldots\rangle. \tag{4.35}$$

Así, al cabo de $L_{\text{máx}}$ pasos, cada rama de la superposición cuántica contiene el camino completo recorrido hasta ese punto, registrado en $\mathcal{H}_{path}$. La medición de este registro permite recuperar directamente una trayectoria concreta $\gamma = (x_0, x_1, \ldots, x_{L_{\text{máx}}})$. La implementación de este contador clásico provoca que se esté hablando todo el rato de un algoritmo híbrido, y no de uno puramente cuántico.

### 4.2.5. Registro parada

El problema de combinar la caminata de Szegedy con los dos registros adicionales reside en el hecho de que la actualización del camino se da para todas las trayectorias al mismo tiempo. Esto es un problema cuando en algún camino se ha llegado al nodo objetivo previa la finalización del número de iteraciones escogidas. Al mantener la aplicación del operador de Szegedy aunque se haya encontrado una ruta que conecte el nodo inicial y objetivo, la amplitud asociada se ve alterada. Por ello, es esencial añadir a la estructura un mecanismo que permita detectar si se ha llegado al nodo objetivo y, en caso afirmativo, impida la aplicación del resto de operaciones sobre esa rama.

Para evitar que ramas que alcanzan el objetivo sigan evolucionando y modifiquen su camino registrado, se incorpora el registro parada. El registro parada $H$ se introduce para que aquellas ramas de la superposición que alcancen el nodo objetivo $x_\star$ dejen de

evolucionar, preservando inalterado el camino almacenado y deteniendo el contador. Para ello se define un cúbit adicional en el espacio:

$$\mathcal{H}_H = \text{span}\{|0\rangle, |1\rangle\}, \tag{4.36}$$

donde $|0\rangle$ indica que el objetivo aún no ha sido alcanzado y $|1\rangle$ que ya se ha llegado al nodo objetivo.

Entonces, la parada se implementa de manera unitaria utilizando un cúbit $H$ y un cúbit auxiliar $B$. Este cúbit auxiliar se introduce para evitar "sobreescribir" el estado del cúbit $H$. Es decir, una vez se alcance el valor $|1\rangle$, no se modifica para el resto de iteraciones de la caminata. El objetivo es que $H$ se transforme en $|1\rangle$ en el primer paso en el que se alcanza el nodo objetivo $x_\star$, y que permanezca en dicho valor durante el resto de la evolución.

Se considera en primer lugar el espacio de Hilbert para los nodos visto en la Ecuación 4.2. Matemáticamente, el proyector que detecta el objetivo en el primer registro (tras cada aplicación del desplazamiento $S$) es:

$$P_D = \sum_x |x_\star, x\rangle\langle x_\star, x| \quad \subset \quad \mathcal{H}_E. \tag{4.37}$$

Los dos cúbits, $H$ y $B$, usan ambos la base computacional $\{|0\rangle, |1\rangle\}$. Además, se introduce la notación $\Pi_Q^{(b)} = |b\rangle\langle b|_Q$ para los proyectores sobre los estados de un cúbit $Q$, y $X_Q$ para la aplicación de una puerta Pauli-$X$ ($NOT$) en dicho cúbit. Así, el primer paso consiste en el cálculo reversible del cúbit auxiliar. Se construye un operador unitario $U_B$ que modifica el cúbit $B$ si la posición actual del nodo coincide con el objetivo [5]:

$$U_B = (I_{\mathcal{H}_E} - P_D) \otimes I_B \; + \; P_D \otimes X_B. \tag{4.38}$$

A continuación, se implementa la alteración del cúbit $H$. El cúbit $H$ debe convertirse en $|1\rangle$ si y solo si $H = 0$ y $B = 1$. De esta forma, si no estamos en el nodo objetivo o bien ya se ha llegado con anterioridad a dicho nodo, este permanece invariante. Nótese que la inicialización del cúbit $B$ se toma en el estado $|0\rangle$, al igual que el cúbit $H$. Esto se consigue con el operador:

$$U_H = I_H \otimes I_B - \Pi_H^{(0)} \otimes \Pi_B^{(1)} \; + \; X_H \Pi_H^{(0)} \otimes \Pi_B^{(1)}. \tag{4.39}$$

La acción de $U_H$ puede describirse de la siguiente manera: actúa como la identidad fuera del subespacio generado por $H = 0$ y $B = 1$; dentro de dicho subespacio aplica $X_H$,

[5] Considérese la propiedad de los proyectores: $\sum_{(x,y)\in E} |x, y\rangle\langle x, y| = I_{\mathcal{H}_E}$

modificando el valor de $H$; y si $H = 1$, se mantiene invariante incluso en el caso de que $B = 1$.

En el tercer paso se restaura el cúbit $B$ a su valor inicial para mantener unitaria la evolución. Esto se logra aplicando el inverso de $U_B$, que coincide con él mismo por ser hermítico. La actualización unitaria deseada para la parada se obtiene entonces como:

$$U_{halt} = U_B \, U_H \, U_B. \tag{4.40}$$

De esta forma, el cúbit $H$ altera su valor la primera vez que llega al objetivo y no vuelve a tomar el valor $|0\rangle$.

Conforme a la creación de esta nueva operación que detiene el avance de la caminata, se ha de modificar cómo se implementan el resto de operadores para que consideren el valor del cúbit $H$. Los cambios integrados son los siguientes:

- **Modificación del operador moneda y del operador desplazamiento:** sea $R$ la moneda original asociada a Szegedy. La versión influenciada por el registro parada se define :

  $$\widetilde{R} = R \otimes \Pi_H^{(0)} + I \otimes \Pi_H^{(1)}, \tag{4.41}$$

  De forma análoga, para el desplazamiento $S$ se tiene:

  $$\widetilde{S} = S \otimes \Pi_H^{(0)} + I \otimes \Pi_H^{(1)}. \tag{4.42}$$

  Esto asegura que:

  - si $H = 0$, la caminata evoluciona con moneda y desplazamiento estándar.
  - si $H = 1$, la dinámica se detiene en el vértice alcanzado.

Resumiendo, un paso completo de la caminata (medio en la caminata de Szegedy) con registro parada se implementa mediante los operadores[6]:

$$\widetilde{U} = U_{halt} \, U_{path,\ell} \, \widetilde{S}\widetilde{R}, \tag{4.43}$$

y, tras aplicar la operación anterior, se implementa $U_\ell$ sobre el registro clásico.

De esta forma, cada trayectoria al llegar por primera vez al nodo objetivo queda registrada y sin alteraciones posteriores.

[6] Aunque el operador $U_{path,\ell}$ no es necesario modificarlo con la adición del registro parada, puede modificarse para evitar las operaciones redundantes una vez se ha escrito por primera vez el nodo objetivo.

### 4.2.6. Medición y extracción del camino

La última componente del algoritmo es la medición de los registros auxiliares. Tras la evolución de la caminata cuántica se procede a medir el estado global en la base computacional. El registro parada $H$ discrimina entre caminos exitosos y fallidos: solamente en las ramas con $H = 1$ se ha alcanzado el nodo objetivo $x_\star$. El registro camino revela la secuencia de vértices recorrida desde el nodo inicial hasta el objetivo y el contador $\ell$ indica la longitud alcanzada. El diseño del kernel de transición asegura que, dentro del conjunto de caminos exitosos, las amplitudes relativas se distribuyan a favor de los de menor coste esperado. En consecuencia, al realizar la medición final, el camino de coste mínimo es el que aparece con mayor probabilidad, lo que permite extraerlo directamente como salida del algoritmo.

En definitiva, la medición actúa como el mecanismo de selección final: convierte en información clásica la estructura de amplitudes acumulada a lo largo de la caminata, entregando como resultado la trayectoria más eficiente hacia el nodo objetivo.

## 4.3. Integración del algoritmo completo

Una vez descritos los componentes individuales, es posible combinarlos en un procedimiento que describe al algoritmo propuesto. La idea central es partir del nodo inicial $x_0$, propagar amplitud en paralelo sobre todas las trayectorias posibles mediante la caminata de Szegedy y utilizar los registros auxiliares para almacenar la información del camino y detener la evolución en las ramas que alcanzan el objetivo. A continuación, se vuelve a detallar, esta vez en conjunto, como se estructura el algoritmo propuesto al completo.

El sistema se prepara en el estado inicial:

$$|\psi_0\rangle = |x_0\rangle\,|0\rangle \otimes |path = (x_0, \varnothing, \dots)\rangle \otimes |H = 0\rangle \otimes |B = 0\rangle\,, \tag{4.44}$$

donde:

- el primer registro codifica el vértice actual.
- el segundo registro corresponde al vértice vecino.
- el tercero ($path$) almacena la secuencia de vértices recorridos.
- $H$ es el cúbit parada, que discrimina ramas exitosas de fallidas.

- el último ($B$) es el cúbit auxiliar para la actualización del cúbit parada.

El kernel de transición elegido ($P^h$) determina los estados moneda:

$$|\psi_x(\beta_w, \beta_h)\rangle = \sum_{y \in \Gamma(x)} \sqrt{P^h(x,y)}\ |x\rangle\,|y\rangle\,, \tag{4.45}$$

que gobiernan la dinámica de la caminata.

Cada iteración de la caminata (mitad de un iteración de Szegedy) aplica sobre los registros cuánticos la composición unitaria:

$$\widetilde{U} = U_{halt}\, U_{path,\ell}\, \widetilde{S}\widetilde{R}, \tag{4.46}$$

donde:

- $\widetilde{R} = R \otimes \Pi_H^{(0)} + I \otimes \Pi_H^{(1)}$ es la moneda controlada por el registro parada.
- $\widetilde{S} = S \otimes \Pi_H^{(0)} + I \otimes \Pi_H^{(1)}$ es el desplazamiento controlado por el registro parada.
- $U_{path,\ell}$ copia el vértice actual en el registro camino en la posición $\ell$.
- $U_{halt}$ implementa la condición de parada $H \leftarrow H \vee \mathbf{1}_{\{x=x_\star\}}$ con ayuda del cúbit $B$.

Para garantizar que todas las trayectorias simples pueden ser exploradas, se fija $L_{\text{máx}} = |V| - 1$, donde $|V|$ es el número de nodos del grafo. Tras $L_{\text{máx}}$ pasos, se asegura que se han visitado todos los nodos en al menos una trayectoria. Al concluir las iteraciones, el estado global adopta la forma:

$$|\psi\rangle = \sqrt{p}\ |\psi_{\text{s}}\rangle|1\rangle_H + \sqrt{1-p}\ |\psi_{\text{f}}\rangle|0\rangle_H\,, \tag{4.47}$$

donde:

- $|\psi_{\text{s}}\rangle$ es la superposición de caminos completos que alcanzaron el objetivo.
- $|\psi_{\text{f}}\rangle$ corresponde a las trayectorias que no lo alcanzaron.
- $p$ es la probabilidad total de éxito, es decir, de estar en un camino que contenga al objetivo.

Finalmente se mide el sistema:

- el registro $H$ garantiza que se obtenga un camino exitoso con alta probabilidad.
- el registro camino revela el camino completo desde $x_0$ hasta $x_\star$.
- la forma en la que se ha definido el kernel de transición otorga una mayor probabilidad de medir aquellos caminos con menor coste asociado.

# 5. Resultados

En esta sección presentamos un conjunto de grafos de ejemplo con distintas estructuras sobre los aplicaremos de manera analítica el algoritmo propuesto. Se mostrarán los histogramas correspondientes a la probabilidad de medir cada uno de los posibles caminos, mostrando que la probabilidad de medir la cadena de bits correspondiente al camino óptimo es notablemente superior al resto de caminos. También se estudiará el efecto de los parámetros $\beta_w$ y $\beta_h$ en función de las distintas características del problema.

En primer lugar, conviene explicar cómo se han obtenido los resultados. Supongamos que se aplica el algoritmo a un grafo de cuatro vértices. En este caso se necesitarían dos cúbits para codificar los vértices, es decir, cuatro cúbits para los dos primeros registros que representan las aristas. El registro camino necesitaría ocho cúbits, dado que el camino más largo es de longitud cuatro y cada vértice se codifica con dos cúbits. El registro contador de longitud necesitará de dos cúbits y, adicionalmente, necesitaremos dos más para los dos registros de parada. Esto hace necesarios 16 cúbits en total para codificar el problema, lo que se traduce en vectores de estado de $2^{16}$ componentes y operadores de tamaño $2^{16} \times 2^{16}$. Dado que se trata de elementos demasiado grandes para calcularlos con un ordenador clásico, se ha optado por otra vía para simular la evolución del algoritmo.

Fijándonos en las ecuaciones 4.34 y 4.35 podemos ver que tras cada paso de la caminata se multiplica el vector de estado correspondiente a un camino por un factor dependiente de la siguiente arista tomada por ese camino $\sqrt{p_{xy}}$. De esta forma, un camino que empieza en el vértice 0 y pasa por los vértices 1 y 2 tendrá una amplitud $\sqrt{p_{01}p_{12}}$. En cada iteración aparecerán nuevos estados, uno por cada posible intersección en cada uno de los nodos actuales. Debido al problema del tamaño de los operadores y los vectores de estado, se ha decidido calcular estas amplitudes con una función recursiva.

El primer grafo que se ha seleccionado para probar el algoritmo se puede ver en la Figura 5.1. En gris se muestran los nodos inicial (0) y final (3), en negro y sobre las aristas los pesos asociados a cada una de ellas y en gris y junto a los nodos el valor heurístico de cada uno. En este grafo hay cuatro posibles caminos para alcanzar al objetivo desde el nodo inicial. Tal y como se ha planteado en la Sección 4, el camino óptimo tendrá como mucho $|V|$ nodos, o lo que es lo mismo, $|V| - 1$ aristas, por lo que deberemos ejecutar tres

iteraciones del algoritmo para asegurar que se ha encontrado el objetivo.

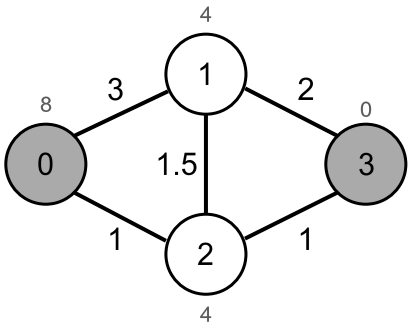
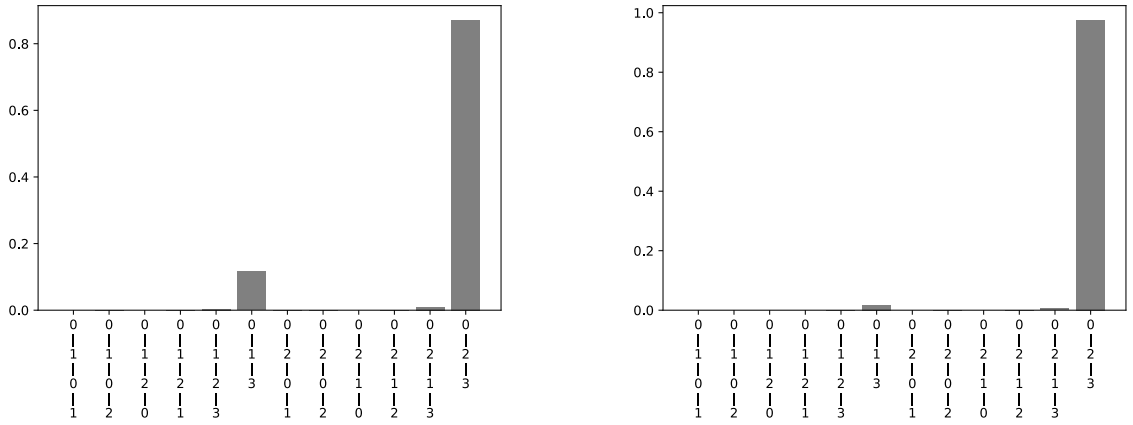



Figura 5.1: Primer grafo de ejemplo. Fuente: elaboración propia.

Con este ejemplo se busca estudiar el comportamiento del algoritmo cuando la heurística no da demasiada información, pues ambas opciones desde el inicio son igual de prometedoras, aunque las aristas no tienen el mismo peso. En la Figura 5.2a se puede ver cómo la probabilidad de medir el camino óptimo (eje de ordenadas) tras tres iteraciones es notablemente superior a la de medir cualquier otro camino, aún cuando ambos parámetros toman el mismo valor. Aun así, la probabilidad de medir un camino no óptimo como $(0, 1, 3)$ aún es superior al 10 %. En la Figura 5.2b se ha aumentado el valor del parámetro $\beta_w$, cuya función es dirigir la búsqueda hacia las aristas de menor peso, y se puede ver que la probabilidad de medir el camino óptimo es de casi el 100 %.




(a) $\beta_w = \beta_h = 1$ (b) $\beta_w = 2,\ \beta_h = 1$

Figura 5.2: Probabilidades de medir el estado correspondiente a cada camino en el grafo de la Figura 5.1 tras tres iteraciones del algoritmo. Fuente: elaboración propia.

A continuación se muestra otro ejemplo de grafo (Figura 5.3), en el que la heurística informa de manera exacta del coste del camino óptimo a través de cada nodo. Además,

hay un camino alternativo que alcanza al objetivo en tan solo un paso, pero sin ser óptimo.

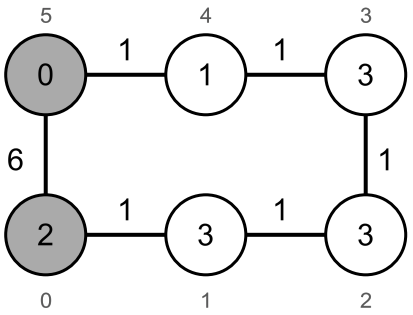
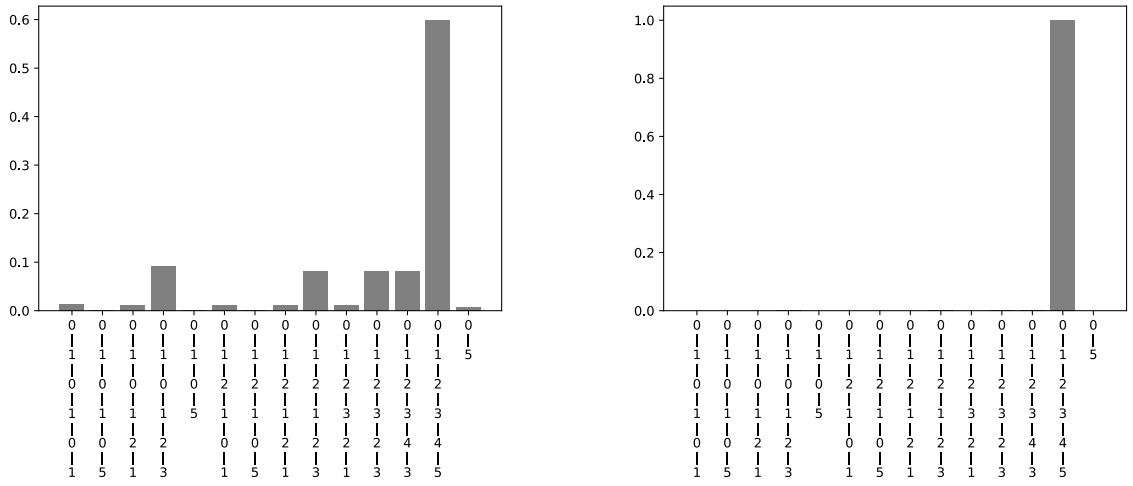



Figura 5.3: Segundo grafo de ejemplo. Fuente: elaboración propia.

Al igual que en el caso anterior, cuando se inicializan ambos parámetros a la unidad se obtiene una probabilidad de medir el camino óptimo mayor que la de cualquier otro camino (Figura 5.4a), aunque en este caso es tan solo del 60 %. De la misma manera, ajustando los parámetros $\beta_w$ y $\beta_h$ se ha conseguido aumentar dicha probabilidad hasta prácticamente un 100 % (Figura 5.4b).



(a) $\beta_w = \beta_h = 1$ (b) $\beta_w = 4,\ \beta_h = 4$

Figura 5.4: Probabilidades de medir el estado correspondiente a cada camino en el grafo de la Figura 5.1 tras tres iteraciones del algoritmo. Fuente: elaboración propia.

Finalmente, se utiliza un grafo algo más complicado que los anteriores (Figura 5.5), con mayor un grado en la mayoría de los nodos. En este caso la heurística da una buena estimación de la distancia real, aunque no exacta, y contamos con una mayor cantidad de caminos posibles entre el nodo inicial y el objetivo.

En la Figura 5.6 se pueden observar los resultados obtenidos para este grafo. En esta

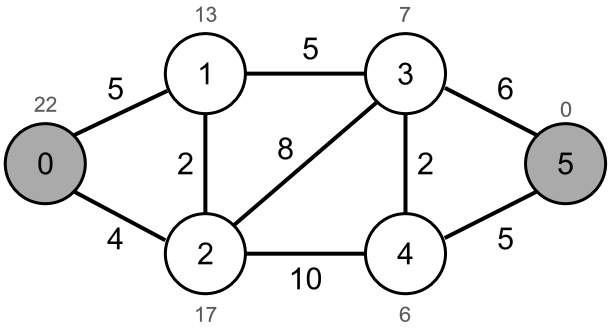
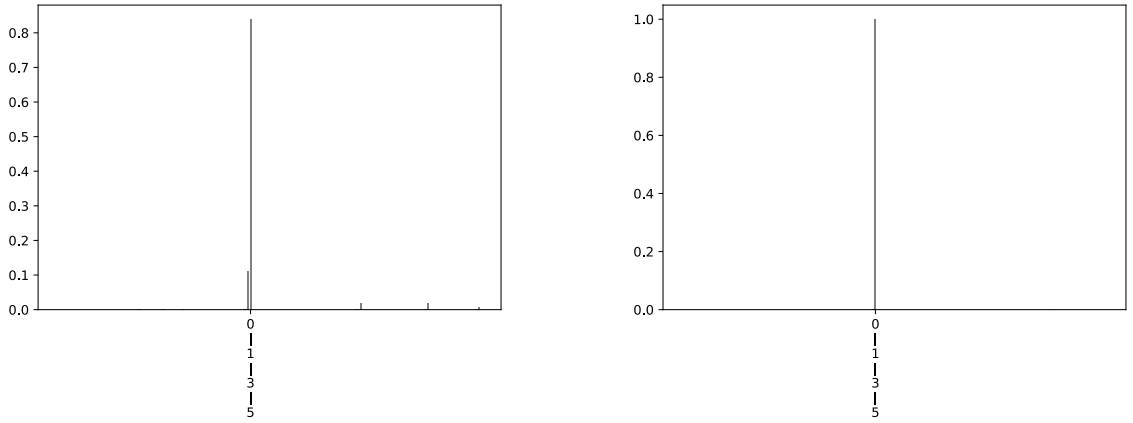



Figura 5.5: Tercer grafo de ejemplo. Fuente: elaboración propia.

figura solamente se muestra el camino óptimo en el eje horizontal para una mejor visualización, dado que para este ejemplo existen 199 posibles caminos. La probabilidad de medir el camino óptimo tras cinco iteraciones del algoritmo con ambos parámetros inicializados a la unidad es superior al 80 % (Figura 5.6a). En este caso, se puede aumentar dicha probabilidad hasta prácticamente un 100 % favoreciendo las aristas con menor heurística (Figura 5.6b).




(a) $\beta_w = \beta_h = 1$ (b) $\beta_w = 1,\ \beta_h = 2$

Figura 5.6: Probabilidades de medir el estado correspondiente a cada camino en el grafo de la Figura 5.1 tras tres iteraciones del algoritmo. Fuente: elaboración propia.

Una vez se ha comprobado que el algoritmo es capaz de resolver el problema objetivo, se procede a analizar su complejidad.

## 5.1. Complejidad del algoritmo

En esta sección se analiza la complejidad, en anchura y profundidad, del algoritmo propuesto en este Trabajo. Se estudia, en primer lugar, la complejidad en anchura, esto es, la cantidad de cúbits (y bits) necesarios para la implementación. En segundo lugar, se obtiene la complejidad relativa a la profundidad del algoritmo. La profundidad hace referencia al conjunto de operaciones en el circuito necesarias para completar la adquisición de resultados. Para cada una de las partes del análisis se descompone la contribución de cada una de las componentes al término total de la complejidad. Detallando y explicando la complejidad individual de cada elemento individual del algoritmo.

Antes de comenzar con el análisis, conviene fijar algunos parámetros que permiten caracterizar los órdenes de magnitud de las cantidades que aparecerán más adelante. Considérese un grafo ponderado $G = (V, E)$, donde $V$ es el conjunto de vértices y $E$ el conjunto de aristas. Denotando $|V| = n$ como el número de vértices, que mide el tamaño del grafo en términos de nodos, y $|E| = m$ como el número de aristas. Además, denominando $d$ al grado máximo, que corresponde al mayor número de vecinos posibles en alguno de los nodos del grafo. Estos parámetros sirven como base para acotar complejidades y facilitar la interpretación de los resultados.

### 5.1.1. Complejidad en anchura del algoritmo

En este apartado se justifica, término a término, la expresión de la anchura total explicando la codificación que se asume en cada registro y la justificación de las cotas. A saber, todas las longitudes de registros cuánticos se encuentra en cúbits, y para los registros clásicos en bits. Asimismo, todos los logaritmos se encuentran en base 2.

De esta forma, la complejidad aportada por cada componente es:

- **Registro $\mathcal{H}_E$:** este registro, correspondiente a la representación de los nodos y sus posibles vecinos (aristas), viene representado por el espacio de Hilbert doble de la ecuación 4.1. Cada registro contiene así $n$ posibilidades, requiriendo para su representación al menos $\log n$ cúbits. Esto como consecuencia de la posibilidad de codificar hasta $2^n$ estados posibles mediante $n$ cúbits. Asimismo, dada la necesidad del registro $\mathcal{H}_E$ de contener a los nodos y sus vecinos, la cantidad de cúbits es, por tanto, doble. Así, este registro presenta una complejidad en anchura de $2\log n$ cúbits.

- **Registro camino:** el algoritmo almacena, tras cada iteración, el vértice actual en una celda del registro de camino. Si se realizan $L_{\text{máx}}$ desplazamientos y se almacena el vértice inicial, el número de celdas a reservar es $L_{\text{máx}} + 1$. Cada celda debe poder codificar un vértice de entre $n$, luego cada una necesita de $\log n$ cúbits. Por tanto, se requieren $(L_{\text{máx}} + 1) \log n$ cúbits para poder almacenar la información relativa a cada trayectoria.

- **Cúbit $H$ y cúbit auxiliar $B$:** en ambos casos, solo se precisa de dos cúbits, uno para cada componente. Por ende, su contribución a la complejidad es constante de la forma $\mathcal{O}(1)$ y, en consecuencia, se puede tomar como despreciable para el cálculo del orden total en anchura.

- **Contador $\ell$:** el contador $\ell$ contiene información de la celda del registro camino en la que debe escribirse el vértice actual. Debe representar $L_{\text{máx}}$ valores posibles si se empieza en $\ell = 1$ y se incrementa tras cada iteración. De aquí, se tiene que el contador necesita de $L_{\text{máx}}$ bits para representar todos los posibles valores mediante codificación *one-hot*.

Dada la expresión $L_{\text{máx}} = n - 1$, se desarrolla el cálculo. De este modo, la complejidad en anchura o número de cúbits (y bits) totales se toma como una función $N(n)$ que cumple:

$$N(n) = \underbrace{2 \log n}_{\text{aristas}} + \underbrace{n \log n}_{\text{camino}} + \underbrace{\mathcal{O}(1)}_{H,B} + \underbrace{n - 1}_{\text{contador}} . \tag{5.1}$$

En definitiva, considerando el término de mayor peso y su contribución, se puede aproximar:

$$N(n) = \underbrace{\mathcal{O}(n \log n)}_{\text{cúbits}} + \underbrace{\mathcal{O}(n)}_{\text{bits}} . \tag{5.2}$$

### 5.1.2. Complejidad en profundidad del algoritmo

El análisis de la profundidad requiere descomponer cada medio paso de la evolución unitaria modificada $\tilde{U}$ e identificar el número de capas de puertas que no pueden emplearse en paralelo. La profundidad no mide el número total de puertas sobre el circuito, sino el número de capas mínimo para ordenar el circuito bajo operaciones simultáneas (Nielsen y Chuang, 2010).

A continuación, se describe la complejidad de cada uno de los bloques:

- **Operador moneda $\tilde{R}$:** el coste de la reflexión se encuentra dominado por el operador que caracteriza la preparación de los estados moneda. Por ejemplo, usando rotaciones uniformemente controladas para obtener un estado arbitrario que permita crear el operador moneda adaptado a cada nodo. Con esta metodología se obtiene una cota máxima para la profundidad de $\mathcal{O}(n)$ (Möttönen et al., 2005; Bergholm et al., 2005; Plesch y Brukner, 2011).

- **Operador desplazamiento $\tilde{S}$:** esta operación consiste en intercambiar dos registros de tamaño $\log n$ mediante la operación $|x\rangle|y\rangle \mapsto |y\rangle|x\rangle$. En hardware con conectividad completa, los intercambios entre distintos pares pueden ejecutarse en paralelo, resultando en profundidad constante $\mathcal{O}(1)$. Sin embargo, en arquitecturas con conectividad restringida, dependientes de la topología del procesador cuántico, el coste de implementar un intercambio entre registros arbitrarios requiere una mayor cantidad de operaciones. Estas operaciones añadidas sirven como mecanismo para conectar cúbits que siguiendo la topología del procesador no interactúan entre sí. Este exceso se ha estudiado en la literatura y se puede traducir en un factor de $\mathcal{O}(\log n)$ (Beals et al., 2012).

- **Operador camino $U_{path,\ell}$:** se requiere copiar el vértice actual en la posición señalada por el contador $\ell$. Con la codificación *one-hot*, cada valor del contador corresponde a un único bit activo, de modo que basta con asociar un control por cada bit clásico: únicamente el bit activo habilita las CNOT necesarias para copiar el vértice en la posición correspondiente del registro camino. En términos de complejidad, el registro *one-hot* implica disponer de $L_{\text{máx}}$ controles clásicos, uno por cada posible posición del contador, lo que supone un coste espacial lineal en $L_{\text{máx}}$. Una vez activado, la copia del vértice en la posición seleccionada se implementa mediante un conjunto de CNOT que, bajo conectividad total, pueden ejecutarse en paralelo con profundidad $\mathcal{O}(1)$. En arquitecturas con conectividad restringida, este coste en profundidad aumenta hasta $\mathcal{O}(\log n)$ (Beals et al., 2012).

- **Operador parada $U_{\text{halt}}$:** requiere comparar el vértice actual con el vértice objetivo en cada iteración de la caminata. Dicha comparación equivale a evaluar el valor del nodo actual a través de operaciones controladas asignadas según el nodo objetivo en cada caso. Es decir, se tiene una operación tal que se controla el valor del cúbit auxiliar $B$ según los valores para el vértice actual. Esto se consigue conociendo de

antemano cuál es el nodo objetivo y definiendo qué conjunto de valores para los cúbits de control han de convertir el estado de $B$ de $|0\rangle$ a $|1\rangle$. La forma de lograr esto es mediante una NOT multicontrolada. Esta puerta se debe ajustar para que en aquellos cúbits que en el correspondiente nodo objetivo tendrían valor 1 actúen como controladores; y, para aquellos que valdrían 0, lo hagan como "anticontroladores". Con este último término, se hace referencia a controles que permitan que se realice la operación si y solo si el cúbit de control vale 0. Se comportan de forma contraria a un cúbit de control típico. Este tipo de controladores se pueden crear sin más que añadir una puerta NOT previa y posteriormente al control. La complejidad asociada a esta NOT multicontrolada aumenta linealmente con la cantidad de cúbits de control, así para este caso es del orden de $\mathcal{O}(\log n)$ (Saeedi et al., 2013).

Por otro lado, la actualización requiere únicamente de un par de puertas CNOTs. Primero, una CNOT del cúbit $H$ hacia el cúbit $B$ y posteriormente en el sentido contrario. Esto permite que el cúbit $B$ no sobreescriba $H$ una vez se ha llegado al nodo objetivo. Al ser un número constante de puertas, su complejidad asociada es del orden de $\mathcal{O}(1)$.

- **Incremento del contador $U_\ell$:** se busca aumentar el valor que codifica el contador hacia el siguiente a cada iteración. Dado que se trata de un registro clásico y siguiendo la codificación *one-hot* adoptada, esto se consigue de forma generalista mediante un intercambio entre los bits adyacentes correspondientes en cada iteración. Por tanto, siendo un registro clásico esta operación tiene una cota máxima en $\mathcal{O}(1)$.

Reuniendo cada uno de los componentes de un paso de la caminata se tiene una complejidad por iteración de:

$$D(n) = \underbrace{\mathcal{O}(n)}_{\tilde{R}} + \underbrace{\mathcal{O}(\log n)}_{\tilde{S}} + \underbrace{\mathcal{O}(n\log n)}_{U_{path,\ell}} + \underbrace{\mathcal{O}(\log n) + \mathcal{O}(1)}_{U_{halt}} + \underbrace{\mathcal{O}(1)}_{U_\ell}, \tag{5.3}$$

donde se ha utilizado la equivalencia $L_{\text{máx}} = n - 1$.

Se puede ver cómo el término de mayor peso es el asociado al almacenamiento del camino seguido. Se puede considerar, por ende, que la complejidad de cada iteración es aproximadamente de orden $\mathcal{O}(n \log n)$. Sin embargo, para obtener la complejidad total queda añadir el coeficiente asociado al número de iteraciones a realizar de cada paso. De modo que la complejidad en profundidad total del algoritmo resulta:

$$D_{\text{total}}(n) = (n-1)D(n) \approx \mathcal{O}(n^2 \log n). \tag{5.4}$$

## 5.2. Comparativa con algoritmos clásicos

Una comparación completa entre el algoritmo cuántico presentado y los algoritmos clásicos como Dijkstra o A* exigiría el diseño de comparaciones concretas y un análisis experimental detallado que incluya diferentes topologías de grafos, medidas de memoria efectiva, profundidad física de los circuitos y tasas de error. Sin embargo, en la era NISQ resulta difícil llevar a cabo este tipo de medidas: la fidelidad limitada, la decoherencia y la escasa conectividad de los procesadores hacen que las métricas teóricas (profundidad $\mathcal{O}(n^2 \log n)$ y anchura $\mathcal{O}(n \log n)$) se vean rápidamente superadas por los errores acumulados. En contraste, los algoritmos clásicos aceptan costes de $\mathcal{O}(m + n \log n)$ en tiempo y $\mathcal{O}(n)$ en memoria para Dijkstra y el peor caso de A* (Dijkstra, 1959; Hart et al., 1968; Cormen et al., 2009). Magnitudes muy asumibles incluso en grafos densos con decenas de miles de nodos. Esto hace que, bajo criterios puramente de eficiencia, este algoritmo cuántico no represente en la actualidad una opción ventajosa.

No obstante, la construcción del algoritmo tiene un valor importante en sí misma. Por un lado, ofrece un marco formal para trasladar estructuras clásicas de búsqueda de caminos a un lenguaje cuántico, lo cual es útil como punto de partida para futuras generalizaciones. Por otro, muestra de manera explícita cuáles son los cuellos de botella que deben abaratarse para alcanzar una ventaja cuántica real: principalmente la escritura en el registro camino $U_{path,\ell}$, que escala como $\mathcal{O}(n \log n)$ por iteración. Avances en esta dirección resultan imprescindibles si se quiere concebir un algoritmo cuántico competitivo para el problema del camino más corto. En este sentido, aunque la propuesta no mejore hoy en día a los clásicos, sí establece una base metodológica para algoritmos híbridos y cuánticos inspirados en sus contrapartes clásicas.

# 6. Conclusiones

En el presente Trabajo se ha estudiado el problema del camino más corto entre dos vértices de un grafo, y se ha propuesto un algoritmo híbrido basado en caminatas cuánticas para resolverlo. Este algoritmo propuesto incorpora la heurística para guiar la caminata cuántica hacia las regiones más prometedoras del espacio de búsqueda, de la misma forma en que algoritmos clásicos como A* guían la búsqueda a través del grafo. Para ello, primero se ha hecho una introducción a los distintos algoritmos clásicos que se centran en resolver este problema, para posteriormente hacer una introducción a los algoritmos cuánticos. Finalmente, se ha expuesto el algoritmo planteado con sus correspondientes resultados de implementación y complejidad.

Se ha podido comprobar como el algoritmo planteado funciona correctamente, y como funciona independientemente de la forma de la estructura del grafo. En las gráficas mostradas, se ha observado que la probabilidad de medir el camino óptimo tras $|V|-1$ iteraciones de la caminata es notablemente superior a la de medir cualquier otro camino, llegando a conseguirse probabilidades de prácticamente un 100 % cuando se ajustan correctamente los parámetros $\beta_w$ y $\beta_h$. En cuanto a estos últimos se ha observado el efecto que cada uno de ellos tiene sobre la búsqueda, siendo $\beta_w$ el encargado de favorecer la búsqueda en la dirección de las aristas de menor peso y $\beta_h$ el equivalente hacia los nodos con menor heurística.

También se ha podido estudiar su complejidad, demostrando que, aunque funcione correctamente, no lo hace de una forma tan eficiente como los algoritmos clásicos mencionados a lo largo de todo el Trabajo. Esto se debe principalmente a la complejidad tanto en anchura como en profundidad que requiere el algoritmo. Aún así, presenta un buen punto de partida para estudiar la incorporación de la heurística en las caminatas cuánticas y de la implementación física de este tipo de algoritmos.

# 7. Trabajo futuro

A lo largo de este Trabajo se han mostrado las ventajas que ofrece la computación cuántica y, en particular, el enfoque basado en caminatas cuánticas de Szegedy con la transformación de Doob. Sin embargo, aún queda un largo camino por recorrer, y el algoritmo propuesto presenta diversas limitaciones que abren interesantes líneas de investigación futuras.

En primer lugar, se ha observado que el algoritmo planteado solo funciona bajo ciertas condiciones: el grafo debe ser no dirigido, finito, conexo y con costes positivos. Por ello, un avance significativo sería el desarrollo de un algoritmo más generalista que pudiera aplicarse a grafos con menos restricciones. De esta forma, se ampliaría el rango de problemas resolubles, permitiendo representar y resolver una mayor variedad de cuestiones.

En segundo lugar, en la actual era NISQ no sería posible implementar de forma directa el algoritmo propuesto, dado que la fidelidad de los dispositivos cuánticos actuales no es lo suficientemente buena, y los errores acumulados afectarían de forma crítica a su desempeño. En este contexto, resulta interesante estudiar posibles modificaciones o aproximaciones del algoritmo que lo hagan viable en *hardware* cuántico ruidoso, evaluando los casos en los que su rendimiento se mantenga competitivo a pesar de las limitaciones tecnológicas.

Otra línea de investigación podría centrarse en la implementación explícita del algoritmo mediante la definición de las puertas lógicas necesarias para modelizar el circuito cuántico correspondiente. Este enfoque permitiría estimar con mayor precisión tanto el coste como la eficiencia del algoritmo en términos de profundidad del circuito, número de cúbits requeridos y tolerancia al ruido. Con esta información sería posible realizar comparaciones más fiables frente a los algoritmos clásicos existentes, y evaluar en qué escenarios el enfoque cuántico logra una verdadera ventaja computacional.

Finalmente, los resultados obtenidos muestran que la amplitud asociada al camino de menor coste suele ser significativamente mayor que la de los demás caminos, aunque esta diferencia depende del número de trayectorias con costes similares. Esto abre una línea

de investigación orientada a la estimación del número óptimo de repeticiones necesarias para ejecutar el algoritmo. Si efectivamente la amplitud del camino deseado se mantiene significativamente superior a la de las alternativas, podría ser suficiente con un número reducido de repeticiones para alcanzar una alta probabilidad de éxito, lo que redundaría en una mejora de la eficiencia global del método.

# 8. Organización del Trabajo

El desarrollo del Trabajo ha consistido en dos partes fundamentales y bien diferenciadas. La primera de ellas consistió en la búsqueda de la mayor cantidad de información posible sobre resolución del problema del camino mas corto, independientemente del tipo de algoritmo implementado. En este periodo trabajamos los tres de forma independiente, reuniéndonos cada semana para compartir información. Esta parte culminó con la decisión de utilizar caminatas cuánticas tipo Szegedy para resolver el problema del camino más corto. La segunda parte del desarrollo de este Trabajo se centró en entender más en profundidad ese tipo de caminatas cuánticas y de estudiar la incorporación de la heurística a las mismas. Nos organizamos de tal forma que nos dividimos el trabajo en dos partes: por un lado, en el desarrollo escrito de la cuestión del arte, introducción y objetivos; y por otro lado, en la búsqueda de la implementación de las caminatas cuánticas con heurística.

Una vez sabíamos como podíamos enfrentarnos al problema, se desarrolló el algoritmo teóricamente mientras hacíamos la implementación, para posteriormente realizar un estudio de la complejidad del algoritmo.

# Bibliografía


Aharonov, D., van Dam, W., Kempe, J., Landau, Z., Lloyd, S., y Regev, O. (2005). Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM Journal on Computing*, 37(1):166–194.

Albash, T. y Lidar, D. A. (2018). Adiabatic quantum computation. *Reviews of Modern Physics*, 90(1):015002.

Ambainis, A. (2003). Quantum walks and their algorithmic applications. *International Journal of Quantum Information*, 1(4):507–518.

Aspect, A., Dalibard, J., y Roger, G. (1982). Experimental test of bell's inequalities using time-varying analyzers. *Physical Review Letters*, 49(25):1804–1807.

Beals, R., Brierley, S., Gray, O., Harrow, A., Kutin, S., Linden, N., Shepherd, D., y Stather, M. (2012). Efficient distributed quantum computing. *arXiv preprint*.

Bell, J. S. (1964). On the einstein podolsky rosen paradox. *Physics Physique Fizika*, 1(3):195–200.

Bellman, R. (1958). On a routing problem. *Quarterly of Applied Mathematics*, 16(1):87–90.

Benioff, P. (1980). The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. *Journal of Statistical Physics*, 22(5):563–591.

Bergholm, V., Vartiainen, J. J., Möttönen, M., y Salomaa, M. M. (2005). Quantum circuits with uniformly controlled one-qubit gates. *Physical Review A*, 71:052330.

Bharti, K., Cervera-Lierta, A., Kyaw, T. T., Haug, T., Alperin-Lea, S., Anand, A., Degroote, M., Heimonen, H., Kottmann, J. S., Menke, T., Mok, W. K., Sim, S., Kwek, L.-C., y Aspuru-Guzik, A. (2022). Noisy intermediate-scale quantum algorithms. *Reviews of Modern Physics*, 94(1):015004.

Boltzmann, L. (1970). *Weitere Studien über das Wärmegleichgewicht unter Gasmolekülen*, volume 67, pages 115–225. Vieweg+Teubner Verlag, Wiesbaden.

Bondy, J. A. y Murty, U. S. R. (2008). *Graph Theory*. Springer.

Carlson, C. y Kolla, A. (2023). A quantum advantage over classical for local max cut. *arXiv preprint arXiv:2304.08420*.

Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S. C., Endo, S., Fujii, K., McClean, J. R., Mitarai, K., Yuan, X., Cincio, L., y Coles, P. J. (2021). Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644.

Chetrite, R. y Touchette, H. (2014). Nonequilibrium markov processes conditioned on large deviations. *Annales Henri Poincaré*, 16(9):2005–2057.

Childs, A. M., Cleve, R., Deotto, E., Farhi, E., Gutmann, S., y Spielman, D. A. (2003). Exponential algorithmic speedup by quantum walk. *Proceedings of the 35th ACM Symposium on Theory of Computing*, pages 59–68.

Childs, A. M., Farhi, E., y Gutmann, S. (2002). Example of the difference between quantum and classical random walks. *Quantum Information Processing*, 1(1):35–43.

Cohen-Tannoudji, C., Diu, B., y Laloë, F. (1977). *Quantum Mechanics, Volume 1*. Wiley, New York.

Colorni, A., Dorigo, M., Maniezzo, V., et al. (1991). Distributed optimization by ant colonies. In *Proceedings of the first European conference on artificial life*, volume 142, pages 134–142. Paris, France.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., y Stein, C. (2009). *Introduction to Algorithms*. MIT Press, 3rd edition.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., y Stein, C. (2022). *Introduction to Algorithms*. MIT Press, 4 edition.

Deo, N. (2016). *Graph theory with applications to engineering and computer science*. Courier Dover Publications.

Deutsch, D. (1985). Quantum theory, the church–turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 400(1818):97–117.

Deutsch, D. y Jozsa, R. (1992). Rapid solution of problems by quantum computation. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 439(1907):553–558.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.

Dinneen, M. J. y Hua, R. (2017). Formulating graph covering problems for adiabatic quantum computers. In *Australasian Computer Science Week Multiconference (ACSW '17)*, pages 18:1–18:10. ACM.

Doob, J. (1957). Conditional brownian motion and the boundary limits of harmonic functions. *Bulletin de la Société Mathématique de France*, 85:431–458.

Euler, L. (1741). Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, pages 128–140.

Faggionato, A., Gabrielli, D., y Ribezzi Crivellari, M. (2009). Non-equilibrium thermodynamics of piecewise deterministic markov processes. *Journal of Statistical Physics*, 137:259–304.

Fan, Z., Xu, J., Shu, G., DIng, X., y Lian, H. (2023). Solving the shortest path problem with qaoa. *SPIN*, 13(01):2350002.

Farhi, E., Goldstone, J., y Gutmann, S. (2014). A quantum approximate optimization algorithm. *Nature Communications*, 5:4213.

Farhi, E., Goldstone, J., Gutmann, S., y Sipser, M. (2000). Quantum computation by adiabatic evolution. Technical Report arXiv:quant-ph/0001106, MIT Center for Theoretical Physics.

Farhi, E. y Gutmann, S. (1998). Quantum computation and decision trees. *Physical Review A*, 58(2):915–928.

Feynman, R. P. (1982). Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6):467–488.

Ford, L. R. J. (1956). Network flow theory. Technical Report P-923, RAND Corporation, Santa Monica, California.

Gaitan, F. y Clark, L. (2014). Graph isomorphism and adiabatic quantum computing. *Physical Review A*, 89(2):022342.

Gibbs, J. W. (2010). *Elementary Principles in Statistical Mechanics: Developed with Especial Reference to the Rational Foundation of Thermodynamics*. Cambridge Library Collection - Mathematics. Cambridge University Press.

Giurgica-Tiron, T., Hindy, Y., LaRose, R., Johri, S., y Pooser, R. C. (2020). Zne: Zeno-based noise extrapolation for quantum-classical algorithms. *Quantum*, 4:P237.

Goldstein, S., Lebowitz, J. L., Tumulka, R., y Zanghì, N. (2020). Gibbs and boltzmann entropy in classical and quantum mechanics. In *Statistical mechanics and scientific explanation: Determinism, indeterminism and laws of nature*, pages 519–581. World Scientific.

Gross, J. L. y Yellen, J. (2005). *Graph Theory and its Applications*. Chapman and Hall/CRC, 2 edition.

Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC)*, pages 212–219.

Harris, D. y Harris, S. (2012). *Digital Design and Computer Architecture*. Morgan Kaufmann, 2 edition.

Hart, P. E., Nilsson, N. J., y Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107.

Hensen, Bas, e. a. (2015). Loophole-free bell inequality violation using electron spins separated by 1.3 kilometres. *Nature*, 526:682–686.

Holevo, A. S. (1973). Bounds for the quantity of information transmitted by a quantum communication channel. *Problemy Peredachi Informatsii*, 9(3):3–11. English translation: Problems of Information Transmission, 9(3), 177–183.

Hua, R. (2016). Adiabatic quantum computing with qubo formulations. Master's thesis, University of Auckland. M.Sc. thesis.

Kempe, J. (2003). Quantum random walks: an introductory overview. *Contemporary Physics*, 44(4):307–327.

Kirkpatrick, S., Gelatt, C., y Vecchi, M. (1983). Optimization by simulated annealing. *Science (New York, N.Y.)*, 220:671–80.

Kjaergaard, M., Schwartz, M. E., Braumüller, J., Krantz, P., Wang, J. I.-J., Gustavsson, S., y Oliver, W. D. (2020). Superconducting qubits: Current state of play. *Annual Review of Condensed Matter Physics*, 11:369–395.

Korf, R. E. (1985). Depth-first iterative-deepening: An optimal admissible tree search. *Artificial intelligence*, 27(1):97–109.

Larkin, J., Jonsson, M., Justice, D., y Guerreschi, G. G. (2020). Evaluation of qaoa based on the approximation ratio of individual samples. *arXiv preprint arXiv:2006.06744*.

Levin, D. A., Peres, Y., y Wilmer, E. L. (2008). *Markov chains and mixing times*. American Mathematical Society.

Li, J. (2025). Exponential speedup of quantum algorithms for the pathfinding problem. *Quantum Information Processing*, 24(3):67.

Li, J., Alam, M., y Ghosh, S. (2021). Large-scale quantum approximate optimization via divide-and-conquer. *arXiv preprint arXiv:2102.13288*.

Lovász, L. (1993). *Random walks on graphs: A survey*, volume 2. János Bolyai Mathematical Society.

Lykov, D., Wurtz, J., Poole, C., Saffman, M., Noel, T., y Alexeev, Y. (2023). Sampling frequency thresholds for the quantum advantage of the quantum approximate optimization algorithm. *npj Quantum Information*, 9(1):91.

Markov, A. A. (1906). Extension of the law of large numbers to quantities that depend on each other. *Proceedings of the Physics and Mathematics Society at Kazan University*, 15(2):135–156.

McClean, J. R., Boixo, S., Smelyanskiy, V. N., Babbush, R., y Neven, H. (2018). Barren plateaus in quantum neural network training landscapes. *Nature Communications*, 9(1):4812.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., y Teller, E. (1953). Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092.

Mohseni, M., Rebentrost, P., Lloyd, S., y Aspuru-Guzik, A. (2008). Environment-assisted quantum walks in photosynthetic energy transfer. *The Journal of Chemical Physics*, 129(17):174106.

Monroe, C., Campbell, W. C., Duan, L.-M., Gong, Z.-X., Gorshkov, A. V., Hess, P. W., Islam, R., Kim, J., Pagano, G., Potter, A. C., Revelle, M., Shen, C., Saffman, M., y Yao, N. Y. (2021). Programmable quantum simulations of spin systems with trapped ions. *Reviews of Modern Physics*, 93(2):025001.

Montanaro, A. (2018). Quantum walk speedup of backtracking algorithms. *Theory of Computing*, 14(15):1–24.

Möttönen, M., Vartiainen, J. J., Bergholm, V., y Salomaa, M. M. (2005). Transformation of quantum states using uniformly controlled rotations. *Quantum Information & Computation*, 5(6):467–473.

Nielsen, M. A. y Chuang, I. L. (2010). *Quantum Computation and Quantum Information*. Cambridge University Press, 10th anniversary edition.

Norris, J. (1998). *Markov Chains*. Cambridge University Press.

Padmasola, V. A. y Chatterjee, R. (2023). Optimization on large interconnected graphs and networks using adiabatic quantum computation. *International Journal of Quantum Information*.

Peruzzo, A., McClean, J., Shadbolt, P., Yung, M.-H., Zhou, X.-Q., y O'Brien, J. L. (2014). A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5:4213.

Plesch, M. y Brukner, Č. (2011). Quantum-state preparation with universal gate decompositions. *Physical Review A*, 83:032302.

Portugal, R. (2016a). Establishing the equivalence between szegedy's and coined quantum walks using the staggered model. *Quantum Information Processing*, 15(4):1387–1409.

Portugal, R. (2016b). Staggered quantum walks on graphs. *Physical Review A*, 93(6):062335.

Portugal, R. y Segawa, E. (2017). Connecting coined quantum walks with szegedy's model. *Interdisciplinary information sciences*, 23(1):119–125.

Preskill, J. (2018). Quantum computing in the nisq era and beyond. *Quantum*, 2:79.

Qing, M. y Xie, W. (2023). Use vqe to calculate the ground energy of hydrogen molecules on ibm quantum. *arXiv:2305.06538*.

Rodríguez García, M. A. (2025). Estrategias de paralelización del algoritmo A* en entornos de memoria compartida. Trabajo de Fin de Grado en Ingeniería Informática. Mención en Computación. Universidad de Murcia.

Russell, S. J. y Norvig, P. (2016). *Artificial intelligence: a modern approach*. pearson.

Saeedi, M., Pedram, M., y Wille, R. (2013). Linear-depth quantum circuits for n-qubit toffoli gates with no ancilla. *Physical Review A*, 87:062318.

Schrijver, A. (2012). On the history of the shortest path problem. In Grötschel, M., editor, *Documenta Mathematica Series*, volume 6, pages 155–167. EMS Press, 1 edition.

Shenvi, N., Kempe, J., y Whaley, K. B. (2003). Quantum random-walk search algorithm. *Physical Review A*, 67(5):052307.

Shimbel, A. (1951). Applications of matrix algebra to communication nets. *The bulletin of mathematical biophysics*, 13(3):165–178.

Shor, P. W. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509.

Soltan Aghaei, M. R., Zukarnain, Z. A., Mamat, A., y Zainuddin, H. (2008). A hybrid algorithm for the shortest-path problem in the graph. In *International Conference on Advanced Computer Theory and Engineering (ICACTE)*, pages 251–255.

Somma, R. D., Boixo, S., Barnum, H., y Knill, E. (2008). Quantum simulations of classical annealing processes. *Physical Review Letters*, 101(13).

Steane, A. M. (1998). Quantum computing. *Reports on Progress in Physics*, 61(2):117–173.

Stollenwerk, T., Lobe, E., y Tröltzsch, A. (2015). Discrete optimisation problems on an adiabatic quantum computer. In *Conference of Simulation and Software Technology, German Aerospace Center (DLR)*.

Sturtevant, N. R. (2025). Moving ai lab. [Disponible online a septiembre de 2025: https://www.movingai.com].

Szegedy, M. (2004). Quantum speed-up of markov chain based algorithms. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 32–41.

Temme, K., Bravyi, S., y Gambetta, J. M. (2017). Error mitigation for short-depth quantum circuits. *Physical Review Letters*, 119(18):180509.

Tero, A., Takagi, S., Saigusa, T., Ito, K., Bebber, D. P., Fricker, M. D., Yumiki, K., Kobayashi, R., y Nakagaki, T. (2010). Rules for biologically inspired adaptive network design. *Science*, 327(5964):439–442.

Venegas-Andraca, S. E. (2012). Quantum walks: a comprehensive review. *Quantum Information Processing*, 11:1015–1106.

Wang, Z., Hadfield, S., Jiang, Z., y Rieffel, E. G. (2018). Quantum approximate optimization algorithm for maxcut: A fermionic view. *Physical Review A*, 97(2):022304.

Wiener, C. (1873). Ueber eine aufgabe aus der geometria situs. *Mathematische Annalen*, 6(1):29–30.

Wocjan, P., Chiang, C.-F., Nagaj, D., y Abeyesinghe, A. (2009). Quantum algorithm for approximating partition functions. *Physical Review A*, 80(2).

Wong, T. G. (2017a). Coined quantum walks on weighted graphs. *Journal of Physics A: Mathematical and Theoretical*, 50(47):475301.

Wong, T. G. (2017b). Equivalence of szegedy's and coined quantum walks. *Quantum Information Processing*, 16(9):215.

Wootters, W. K. y Zurek, W. H. (1982). A single quantum cannot be cloned. *Nature*, 299(5886):802–803.

Zhang, J., Leichenauer, S., Sankaran, K., Sundaram, A., Zlokapa, A., Martinis, J. M., Neven, H., y McClean, J. R. (2020). Qaoa for max-k-sat: A quantum oriented design. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2438–2445.

Zhou, L., Wang, S.-T., Choi, S., Pichler, H., y Lukin, M. D. (2018). Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. *Quantum*, 2:759.

Zhou, Y. y Zeng, J. (2015). Massively parallel a* search on a gpu. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29.