Universidad de Granada

TRABAJO FIN DE GRADO

INGENIERÍA INFORMÁTICA

Aplicación de Metaheurísticas Cuánticas para la Optimización de Cadenas de Servicios en un Modelo de Red 5G

Autor
Mario Guisado García

Directores
Antonio Miguel Mora García
Alejandro Borrallo Rentero

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE TELECOMUNICACIÓN

—

Granada, junio de 2024

Yo, **Mario Guisado García**, alumno de la titulación Grado en Ingeniería Informática de la Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Firmado: Mario Guisado García

Granada, a 24 de junio de 2024.

D. **Antonio Miguel Mora García**, Profesor Titular del Departamento de Teoría de la Señal, Telemática y Comunicaciones.

D. **Alejandro Borrallo Rentero**, Quantum Computing Scientist en Fujitsu.

Informan:

Que el presente trabajo, titulado ***Aplicación de Metaheurísticas Cuánticas para la Optimización de Cadenas de Servicios en un Modelo de Red 5G***, ha sido realizado bajo su supervisión por **Mario Guisado García**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 24 de junio de 2024.

Los directores:

Antonio Miguel Mora García

Alejandro Borrallo Rentero

Agradecimientos

Este trabajo supone el fin de cuatro años de estudio. Durante este tiempo numerosas personas han servido de apoyo para mantener la motivación y la disciplina necesarias para finalizarlos con éxito. Aunque es imposible nombrarlos a todos, quiero destacar:

A mis directores, D. Antonio Miguel Mora García y D. Alejandro Borrallo Rentero, que me proporcionaron los conocimientos y medios necesarios para este trabajo.

A mis profesores de la Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación, que tuvieron la paciencia y dedicación suficientes para guiarme durante el camino.

A la Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación y la Universidad de Granada, por la oportunidad de formarme.

A mis amigos dentro y fuera de la carrera, por las innumerables anécdotas y momentos compartidos.

Y a mi familia por su apoyo incondicional durante todo este tiempo.

Gracias.

Índice

Índice de figuras

1. Introducción

1.1. Optimización de cadenas de servicios en red

Con las nuevas redes 5G surge la posibilidad de desgranar la cobertura de un servicio requerido por un cliente en múltiples servicios ofrecidos por los diferentes nodos que la componen. La arquitectura de 5G no solo promete mayores velocidades y menor latencia, sino que también habilita la segmentación y especialización de la red, permitiendo diferentes tipos de servicios ser gestionados y optimizados de manera independiente. En este contexto, nace el problema del enrutamiento a través de una red con composición de servicios, donde se pretende optimizar la ruta que deben seguir los datos a través de la red para cumplir con los requerimientos de servicio solicitado [9].

Existen muchos factores a tener en cuenta para resolver dicho problema, como puede ser el ancho de banda entre las conexiones de los nodos de la red, los recursos de estos mismos nodos, las funciones que ofrecen y las funciones requeridas por el cliente o el objetivo a minimizar (coste total de la conexión, número de saltos etc).

1.1.1. Enfoques Tradicionales

El problema en cuestión ha sido abordado por la comunidad científica desde diferentes perspectivas, utilizando una variedad de enfoques. Entre ellos destacan:

Algoritmos Genéticos: inspirados en la evolución natural y utilizan operaciones como selección, cruce y mutación para encontrar soluciones óptimas [5]. Son efectivos para problemas complejos pero pueden requerir mucho tiempo para encontrar una buena solución.

Algoritmos de Colonia de Hormigas: Basados en el comportamiento de las hormigas buscando caminos óptimos para encontrar comida [7]. Sufren el mismo problema que los algoritmos genéticos ya que pueden ser muy lentos para ciertas instancias del problema.

Machine Learning: Técnicas de aprendizaje automático utilizadas para predecir la mejor ruta basada en datos históricos y patrones encontrados [6].

Es una de las opciones más interesantes pero se necesita de una grandísima capacidad de cómputo así como enormes bancos de datos para realizar las continuas pruebas a las que deben ser sometidos.

1.1.2. La Alternativa Cuántica

La computación cuántica abre un mundo de posibilidades hasta ahora desconocido a la hora de resolver problemas de optimización y el problema del enrutado es un candidato ideal para beneficiarse de ella. Gracias a las singulares características de esta tecnología, se pretende aumentar significativamente la velocidad y la eficacia a la hora de obtener las soluciones óptimas de este problema, presentando una alternativa a los enfoques más clásicos que sea viable no solo desde un punto de vista académico sino también práctico y comercial.

1.2. Objetivos

En este trabajo, el objetivo principal es resolver un problema de optimización utilizando la computación cuántica y definiendo una metaheurística para ello. Este objetivo se desglosa en varias metas específicas que abarcan tanto aspectos teóricos como prácticos de la computación cuántica aplicada a problemas de optimización complejos.

1.2.1. Adquisición de Conocimientos Teóricos

Adquirir un conocimiento profundo sobre los principios básicos de la computación cuántica, incluyendo todo lo relacionado con la formulación del problema a resolver desde un punto de vista matemático.

Aprender a expresar y definir una formulación que se adecúe específicamente al problema en cuestión y sus restricciones asociadas es crucial para poder resolverlo. Esto incluye identificar las variables y restricciones relevantes así como la función objetivo a minimizar, y dotar a cada uno de estos elementos un peso acorde para poder realizar una buena valoración de las soluciones. Saber como reformular las expresiones si fuera necesario en función de los detalles específicos de cada instancia del problema, reduciendo el número de variables necesarias y aumentando la eficiencia y eficacia de las resoluciones, demostrando así una adquisición de conocimientos teóricos sólida y completa.

1.2.2. Implementación y Ejecución

Además de los aspectos teóricos, se pretende desarrollar un código en función de lo previamente formulado capaz de ejecutarse tanto en simuladores como en ordenadores cuánticos reales, aprendiendo a utilizar las librerías proporcionadas para tal propósito, como la correspondiente al simulador (que permite emular el comportamiento de un ordenador cuántico en un ordenador clásico) para realizar pruebas iniciales y validar la formulación. Se tiene como objetivo también aprender a solicitar ejecuciones en los sistemas ofrecidos por la plataforma D-Wave Leap, donde encontraremos distintos tipos de ordenadores cuánticos y clásicos de alta potencia disponibles para ejecutar una serie de algoritmos diseñados para resolver este tipo de problemas. Esto incluye entender el proceso de traducción de las expresiones generadas, el envío de la petición y el ajuste de parámetros de ejecución para maximizar la probabilidad de obtener soluciones óptimas. Adquirir las habilidades necesarias para operar con estas herramientas y realizar los ajustes necesarios basados en los resultados obtenidos será requisito indispensable para obtener unos buenos resultados.

Se pretende también obtener los conocimientos necesarios para la interpretación y análisis de los resultados. Documentar dichos resultados de manera clara y concisa y presentarlos de manera que sean comprensibles tanto para expertos en el área de computación cuántica como para aquellos con menos experiencia.

1.2.3. Contribución al Campo de la Computación Cuántica Aplicada

Como objetivo final de este Trabajo de Fin de Grado, se quiere contribuir al campo de la computación cuántica aplicada demostrando la viabilidad de esta tecnología para resolver problemas de enrutamiento complejos. Al alcanzar estos objetivos, el trabajo no solo demostrará la capacidad de utilizar la computación cuántica para resolver problemas complejos de optimización, sino que también proporcionará una base sólida y un punto de referencia valioso para futuras investigaciones y aplicaciones en la industria.

1.3. Estructura de la memoria

Este trabajo está estructurado en varias secciones distintas y bien definidas:

- Capítulo 1: Esta primera parte a modo de introducción.
- Capítulo 2: Definición formal del problema. Ejemplo de representación gráfica de una instancia.
- Capítulo 3: Se detallarán los conceptos fundamentales necesarios para comprender completamente el contenido del trabajo. Introducción a la computación cuántica y a los modelos QUBO. Slack variables, simuladores cuánticos, solvers y D-Wave.
- Capítulo 4: Exposición del estado del arte en el campo de estudio actual y contribución de este trabajo.
- Capítulo 5: Planificación de la investigación y tareas realizadas. Diagrama con la distribución a lo largo del tiempo de las mismas.
- Capítulo 6: Detalles de la formulación QUBO y sus restricciones.
- Capítulo 7: Datasets empleados para este trabajo y configuración del entorno de trabajo.
- Capítulo 8: Resultados de las distintas pruebas realizadas. Exposición de varias resoluciones para distintos tipos de grafos, número de agentes y condiciones. Análisis de los datos.
- Capítulo 9: Conclusiones finales.

2. Definición del problema

El problema a resolver es el enrutamiento a través de una red con composición de servicios, con el objetivo de minimizar el coste total de la ruta desde un nodo inicial hasta un nodo final. Este enrutamiento debe incorporar una serie de Funciones Virtuales de Red (VNF) previamente definidas, que están distribuidas a lo largo de los nodos del grafo que modelizan dicha red. Estas funciones son en esencia un software ofrecido por el nodo que las sirve, que suelen realizar tareas específicas de red que tradicionalmente son llevadas a cabo por un hardware específico. Las ventajas que ofrece el uso de estas funciones es una mayor flexibilidad y escalabilidad en la red, la posibilidad de desplegar la red en entornos virtualizados y el ahorro de costes al eliminar el hardware específico. A continuación se detallan los componentes y restricciones del problema:

- Minimización del coste de la ruta: La ruta debe ser la más económica posible en términos de un costo definido, que en este trabajo será el número de saltos totales. Cuantos menos saltos entre nodos, mejor será la solución, siempre que cumpla los requisitos para ser considerada como una solución viable.
- Adquisición de funciones: En el camino desde el nodo inicial al nodo final, la ruta debe adquirir las funciones solicitadas, que se encontrarán repartidas a lo largo de los nodos que componen el grafo del problema. Estas funciones no tienen que ser adquiridas en un orden específico, pero todas deben estar presentes en la ruta final. No es necesario pasar por todos los nodos que contengan las funciones solicitadas, pues dichas funciones solo deben ser adquiridas una única vez.
- Restricciones de recursos: Cada nodo en la red tiene una cantidad limitada de recursos (que se puede responder a su CPU, memoria, etc), y el enrutamiento debe tener en cuenta la disponibilidad de estos recursos. Un nodo que se haya quedado sin recursos dejará de estar disponible, y por tanto las rutas que pasen por él serán invalidadas.
- Ancho de banda de los enlaces: Los enlaces entre nodos tienen un ancho de banda limitado, y al igual que pasaba con los recursos de cada nodo, cuando un enlace deje de estar disponible las siguientes rutas establecidas por posibles conexiones futuras no podrán usarlo.

- Conexiones simultáneas: El problema puede considerar desde una sola conexión o ”agente” hasta múltiples conexiones simultáneas (con un máximo de 3), lo cual puede desembocar en soluciones distintas para cada uno de los agentes debido a la modificación en los recursos disponibles tanto de los nodos como de los enlaces que los unen. A nivel teórico, más agentes no implica mayor dificultad, pero a nivel práctico un aumento de las conexiones conlleva un aumento en el número de variables a tener en cuenta y una mayor dificultad a la hora de refinar los parámetros de entrada del ordenador cuántico. Este refinamiento (referido al cálculo de unas constantes ocultas) suele realizarse mediante postprocesados o machine learning, técnicas a las que no se ha tenido acceso en este trabajo, por lo que se ha limitado el número de agentes a una cantidad aceptable para los medios disponibles.
- Evitar ciclos: Un ciclo es un camino que tiene el mismo nodo inicial y final. Como se explicará más adelante en este documento, las peculiaridades de la formulación matemática y el funcionamiento de los ordenadores cuánticos hacen que a veces puedan encontrar soluciones ”imposibles” pero que no se salten ninguna restricción. Uno de estos casos son soluciones con ciclos aislados de distinto tamaño, que implican conexiones imposibles o que no tienen sentido, por lo que se evitará toda solución que forme ciclo o pase más de una vez por un mismo nodo.

La figura 1 muestra gráficamente una instancia de este problema:

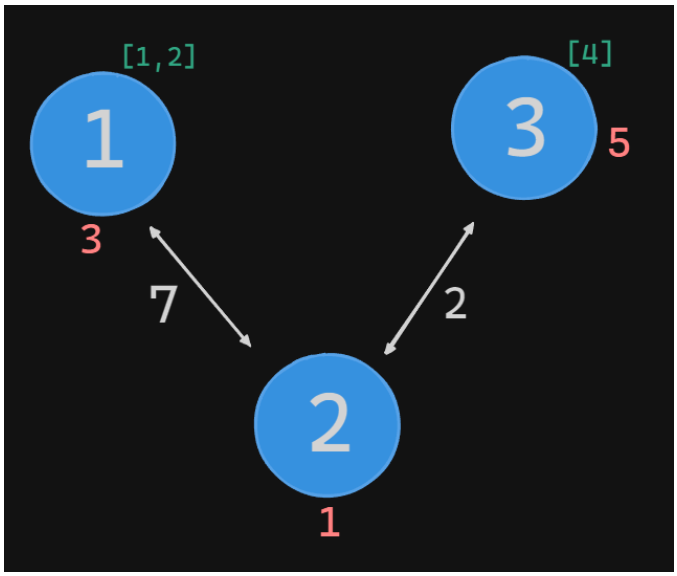

Figura 1: Simplificación de una red.

Donde distinguimos los elementos anteriormente descritos:

- Nodos: Representados en azul, serán los distintos servidores que componen la red y a los que irá saltando el agente.
- Funciones: representadas como una lista de color verde, cada nodo ofrecerá sus propias funciones.
- Recursos: de color rojo, es la capacidad que tendrá el servidor o nodo de atender peticiones o conexiones de cada agente. Cuando un agente se conecte a un servidor -esto es, salte a este nodo-, la capacidad del nodo se verá reducida en una unidad.
- Enlaces: Los enlaces entre nodos están representados como flechas bidireccionales blancas. Esto significa que para todo enlace que una los nodos A y B se podrá saltar desde A hasta B como desde B hasta A.
- Ancho de banda: Cada enlace cuenta con un ancho de banda propio, equivalente a la capacidad de una determinada conexión de manejar peticiones de forma simultánea. Cada vez que una agente utilice un

enlace, su ancho de banda se verá decrementado en la cantidad establecida. Para facilitar la exposición de las resoluciones del problema esta cantidad se ha fijado en 2 unidades.

Todo lo anterior formaría una configuración única sobre la que podemos intentar obtener una solución para unas demandas concretas. Como parte de la definición de la instancia del problema, será necesario indicar:

- Nodo inicial: Indicará el nodo desde el que se origina la conexión.
- Nodo final: Supone el nodo donde finalizará la conexión.
- Funciones requeridas: Es el conjunto de funciones que se quieren solicitar al recorrer el camino. Este conjunto puede ser vacío en caso de no querer adquirir ninguna conexión.

Con estos datos definidos ya se podría obtener la solución óptima al problema, en caso de que la configuración lo permita. En caso contrario, la solución obtenida estaría dentro del espacio de soluciones imposibles o inviables, y sería aquella con un menor coste.

3. Conceptos Preliminares

3.1. Entendiendo la Computación Cuántica

La computación cuántica es la redefinición de los conceptos tradicionales de la computación clásica, llevándola a nuevas fronteras gracias a las particularidades de la física cuántica. La principal diferencia entre la computación cuántica y la clásica es el empleo de los llamados qubits, que son los equivalentes cuánticos de los bits clásicos. Mientras que los bits en la computación clásica solo pueden existir en uno de dos estados posibles (0 o 1), los qubits tienen la capacidad de representar un 0, un 1 o una superposición de ambos estados simultáneamente y en distinta proporción, en lo que se describe matemáticamente mediante una combinación lineal de los estados básicos. Esto es en esencia, una ”mezcla” de los estados clásicos en una proporción variable, que dota a la computación cuántica de una inmensa capacidad de representar, manejar y procesar información. En la figura 2 se muestra parte de uno de estos ordenadores cuánticos.

Esta propiedad en la que se pueden mezclar los estados, conocida como *superposición*, confiere a la computación cuántica una potencia exponencial en comparación con la computación clásica. Donde un bit clásico puede representar una única información binaria, un qubit puede representar muchísima más información. Esta capacidad de representar y procesar una enorme cantidad de información en paralelo abre la puerta a la solución de problemas que son inabordables para los ordenadores clásicos y enfoques tradicionales.

Además de la superposición, otra característica crucial de los qubits es el *entrelazamiento* cuántico. Cuando dos o más qubits se entrelazan, el estado de uno de ellos puede determinar el estado del otro, sin importar la distancia que los separa. Esta conexión instantánea permite una coordinación y sincronización entre qubits que es imposible de replicar en sistemas clásicos. El entrelazamiento cuántico es esencial para muchas de las aplicaciones más prometedoras de la computación cuántica, ya que facilita la ejecución de algoritmos muy complejos y costosos.

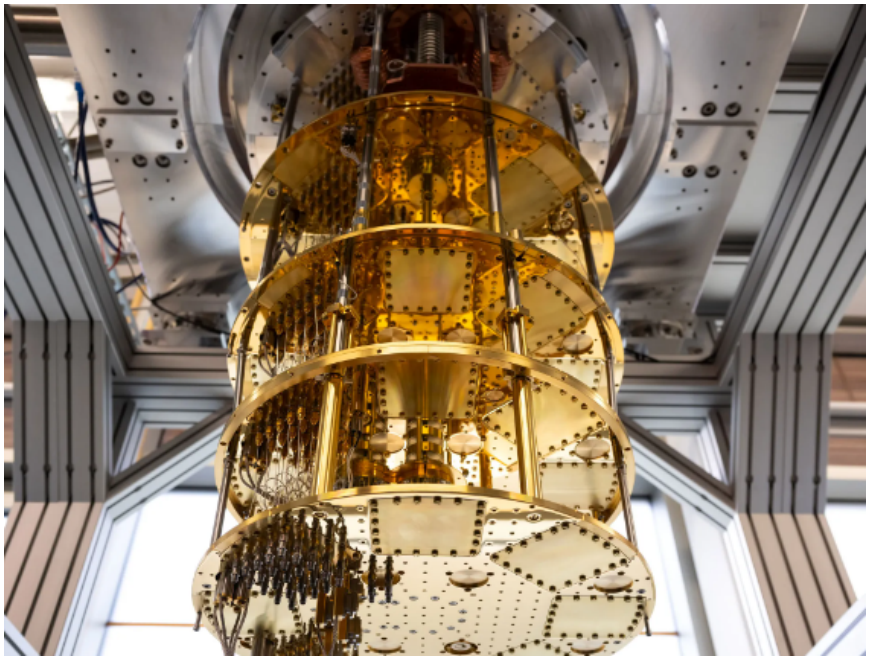

Figura 2: Ordenador cuántico.

El potencial de la computación cuántica se ve maximizado a través del desarrollo de algoritmos cuánticos específicos. En el campo de la optimización, la computación cuántica promete revolucionar el panorama. Muchas aplicaciones industriales y científicas dependen de resolver problemas de optimización complejos, que involucran la búsqueda del mejor resultado entre un vasto número de posibilidades. Los métodos clásicos pueden ser lentos e ineficientes para problemas de gran escala, sin embargo, los algoritmos cuánticos pueden encontrar soluciones óptimas de manera más rápida y eficiente, aprovechando sus características únicas para escapar de mínimos locales y alcanzar el mínimo global de una función de energía.

En resumen, la computación cuántica no solo redefine la computación clásica, sino que la expande a horizontes antes inimaginables. Con su capacidad para procesar información en superposición y aprovechar el entrelazamiento cuántico, junto con algoritmos específicos diseñados para explotar estas propiedades, la computación cuántica se posiciona como una de las tecnologías más prometedoras del siglo XXI. Sus aplicaciones potenciales

en criptografía, optimización, simulación de sistemas cuánticos y más, indican que estamos ante un nuevo campo que revolucionará el conocimiento humano y la industria.

3.2. Computación Cuántica Adiabática

En la computación cuántica adiabática [1] se utiliza un enfoque específico para resolver problemas de optimización, basándose en un concepto físico llamado ”Hamiltoniano”. Esto es de forma simplificada, una herramienta matemática que permite evaluar la cantidad de energía de un sistema. En nuestro problema, el sistema será la expresión matemática del mismo, que incluye todos aquellos posibles saltos, las restricciones asociadas etc. La cantidad de energía será la valoración de una determinada solución para nuestro sistema. En computación cuántica adiabática, cuanta menor sea ese energía, es decir, menor sea la valoración, mejor será la solución que se esté considerando. El Hamiltoniano y las transformaciones que pueda sufrir son manejadas por el ordenador cuántico y se basan en teoría de la mecánica cuántica, por lo que a efectos prácticos no es necesario para el desarrollo y comprensión de este trabajo entender como funcionan de forma profunda, simplemente tener una idea general de su uso. En la figura 3 se muestra un ordenador cuántica que emplea esta tecnología.

Estos problemas son formulados como problemas de Optimización Binaria Cuadrática Sin Restricciones (”QUBO”por sus siglas en inglés). En el contexto de la computación cuántica adiabática, se utilizan dos Hamiltonianos principales, un Hamiltoniano simple y conocido cuyo estado fundamental es fácil de preparar, y el Hamiltoniano del problema, que lo codifica junto con sus restricciones. Su estado fundamental corresponde a la solución óptima del problema.

El sistema cuántico se prepara en el estado fundamental de un Hamiltoniano inicial, que es simple y conocido, y se aplican unas transformaciones de forma gradual sobre el mismo, transformando el Hamiltoniano Inicial en el Hamiltoniano del problema. Si la transformación se realiza de manera suficientemente lenta, según el teorema adiabático, el sistema permanecerá en su estado fundamental durante todo el proceso. Esto asegura que al final del proceso, el sistema se encontrará en el estado fundamental del Hamiltoniano problema, proporcionando así la solución óptima al problema.

Figura 3: Ordenador cuántico adiabático de la empresa D-Wave, primera en comercializar este tipo de ordenadores.

La computación cuántica adiabática es solo uno de los distintos enfoques que existen dentro del creciente campo de la computación cuántica. Su eficiencia a la hora de buscar óptimos y escapar de mínimos locales, y su versatilidad a la hora de adaptarse a diferentes problemas hacen que muchos investigadores se decanten por ella para su aplicación comercial [2]. Esta ha despertado un gran interés en la industria debido a su potencial para resolver problemas complejos que son intratables para los ordenadores clásicos. Algunas áreas de aplicación incluyen:

- Criptografía: La capacidad de resolver problemas más rápidamente que los métodos clásicos podría tener un impacto significativo en la seguridad de la información.
- Optimización de rutas y logística: La optimización de rutas de transporte y la logística de suministro pueden beneficiarse enormemente de la velocidad y eficiencia de la computación cuántica adiabática.
- Inteligencia artificial y aprendizaje automático: La computación cuántica adiabática puede mejorar la eficiencia de los algoritmos de aprendizaje automático y la inteligencia artificial, permitiendo el procesamiento de grandes volúmenes de datos y la optimización de modelos complejos, que de otra manera sería muy costoso con métodos clásicos.

El campo de la computación cuántica adiabática está en constante evolución, con investigaciones en curso que buscan mejorar los algoritmos, aumentar la escalabilidad de los sistemas cuánticos y desarrollar nuevas aplicaciones prácticas. Tiene el potencial de revolucionar múltiples industrias y resolver problemas complejos de manera más eficiente y promete ser una de las innovaciones más importantes del siglo XXI.

3.3. ¿Qué es un QUBO?

QUBO son las siglas de Quadratic Unconstrained Binary Optimization. Esto es, un problema de optimización combinatoria. En él, todas las variables de nuestro problema se expresarán en lo que se conocen como variables binarias. Esto son, variables que pueden tomar únicamente los valores 0 o 1.

Para poder emplear un computador cuántico, es necesario hacer una traducción de las variables de nuestro problema junto con la función a minimizar y restricciones asociadas. Formularemos matemáticamente una función que arrojará un determinado valor según los valores que adquieran las variables binarias definidas, y será trabajo del ordenador explorar las distintas soluciones aprovechando las ventajas en cuanto a eficiencia que nos brinda la computación cuántica, hasta llegar a una solución que considere óptima, siendo esta la solución que arroja el nivel de energía más bajo según la formulación.

De forma simplificada, la formulación seguiría una estructura similar a la siguiente:

$$QUBO = CostFunction + \alpha * (Constrain_1 + Constrain_2...)$$

Donde *Function* es la función de coste a minimizar en nuestro problema, $Constrain_1$ y subsiguientes son las restricciones – aquellos casos que queremos limitar o eliminar de nuestra solución–, y α es un multiplicador asociado a estas restricciones para "amplificar" el efecto negativo que produciría saltárselas. Como se verá más adelante, aunque la anterior es la estructura general, se realizarán pequeños ajustes a la misma para poder abordar mejor nuestro problema.

3.4. Slack Variables

Las *slack variables* son variables adicionales introducidas en problemas de optimización para transformar restricciones de desigualdad en restricciones de igualdad, facilitando así su formulación y resolución. En el contexto de la formulación QUBO, se quiere formular las restricciones del problema de forma que la solución o soluciones óptimas que no rompan dicha restricción den como resultado al evaluarla 0, consiguiendo así no aumentar el valor final de la solución. Se recuerda que en computación cuántica adiabática, que es el enfoque seguido en este trabajo, la forma de medir las soluciones es por su nivel de energía: cuanto más bajo, mejor.

Vamos a ejemplificar el uso de estas variables. En ocasiones, los problemas de optimización vienen con restricciones de desigualdad. Por ejemplo:

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n \leq b$$

Hay muchos casos en los que se puede necesitar expresar una restricción en forma de desigualdad. Utilizando de ejemplo el problema que pretende resolver este trabajo, se podría necesitar que el camino encontrado pase un numero menor o igual a *b* por un determinado nodo, o por el contrario que fuese mayor o igual que este. Para formular este problem en un formato QUBO, es necesario convertir esta restricción de desigualdad en una restricción de igualdad. Aquí es donde entran en juego las slack variables. Se introduciría una slack variable s, que transforma la desigualdad en una igualdad:

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n + s = b$$

Aquí, s es una slack variable que asume el valor excesivo necesario para cumplir la igualdad.

Supongamos que tenemos la restricción:

$$3x_1 + 2x_2 \leq 5$$

Introducimos una slack variable s tal que:

$$3x_1 + 2x_2 + s = 5$$

Donde $s \geq 0$. Esto garantiza que la desigualdad original se cumpla siempre que s sea no negativa. Podemos evaluar el resultado de esta expresión si sustituimos los valores de las variables binarias definidas en la expresión:

$$(3x_1 + 2x_2 + s - 5)^2$$

Donde se ha dejado toda la expresión en un único lado y se ha elevado al cuadrado para evitar posibles valores negativos que resten valor al resultado final y por tanto puedan beneficiar en lugar de perjudicar.

Tal y como se vio antes cuando se explicó la forma general de una formulación QUBO, la integración de esta desigualdad quedaría expresada de la siguiente forma:

$$Minimizar(f(x) + P \cdot (3x_1 + 2x_2 + s - 5)^2)$$

Aquí, P es un coeficiente grande que asegura que cualquier violación de la restricción tenga un alto costo en la función objetivo, forzando así al optimizador a cumplir la restricción.

Entre los beneficios que tiene la inclusión de variables slack destacan:

- **Ampliación de restricciones**: Facilita la conversión de restricciones de desigualdad en igualdades manejables, pudiendo expresar muchas más restricciones más complejas.
- **Compatibilidad con QUBO**: Permite que problemas con restricciones se formulen en el formato QUBO.
- **Flexibilidad**: Las variables de holgura pueden ajustarse para manejar una amplia variedad de restricciones de desigualdad, que de otra manera no podrían expresarse.

Por tanto estas variables serán una herramienta crucial a la hora de formular correctamente nuestro problema y todas sus restricciones, permitiendo así llegar a soluciones óptimas que cumplan todos los requisitos establecidos.

3.5. Simuladores Cuánticos, Solvers y D-Wave

En este trabajo se hará uso en primera instancia de un simulador cuántico para imitar el comportamiento de un sistema cuántico en un entorno clásico. Este simulador es proporcionado en forma de una librería de python que proporcionará los métodos necesarios para evaluar nuestras expresiones matemáticas y generar un resultado similar al que obtendríamos con un ordenador cuántico. La ventaja que tienen es que no se necesita un hardware específico para poder ejecutarlo, y sirven como una buena primera valoración en las fases tempranas del desarrollo del trabajo realizado. Como desventajas encontramos su poca eficacia y eficiencia cuando los problemas se vuelven más grandes y complejos.

En cuando a los solvers, estos son algoritmos diseñados para resolver problemas de optimización utilizando principios de la computación cuántica. El solver utilizado será un solver híbrido, donde una parte del problema se resuelve en un ordenador cuántico y otra parte se realiza en un ordenador clásico con una alta capacidad de cómputo. Este solver será proporcionado por la empresa D-Wave [11] la cual es la primera empresa del mundo en comercializar esta tecnología, y en ofrecer una plataforma abierta al público en la que los desarrolladores pueden solicitar ejecuciones a distintos tipos de solvers que ofrecen. Existe la posibilidad de tener acceso de forma mensual durante 1 minuto a algunos de sus solvers cuánticos, y también cuentan con distintos tipos de tarifas y servicios para empresas que quieran emplear sus servicios. El motivo del uso del solver híbrido en lugar del puramente cuántico es la limitación de tiempo que tiene este último, en el que se limita enormemente la cantidad de tiempo sin interrupciones que un mismo usuario puede emplear la unidad, mientras que en el solver híbrido se puede emplear mucho más tiempo en el refinamiento de la solución.

4. Estado del Arte

4.1. Panorama Actual

La computación cuántica aplicada a problemas de optimización es un campo de investigación relativamente nuevo, aunque en auge. En cuanto a la formulación QUBO, ya existen artículos [3] públicos que facilitan mucho la comprensión del tema, así como una serie de manuales [11] puestos a disposición por la empresa de computación cuántica D-Wave. Estos recursos son buenas opciones como introducción desde un punto de vista técnico a este campo.

En cuanto a la resolución del problema del enrutamiento a través de una red con composición de servicios, uno de los propios directores de este trabajo, D. Antonio Miguel Mora García, publicó su artículo [7] en 2021. Este trabajo se explora la resolución del problema del enrutamiento con cadenas de servicios mediante el algoritmo "Ant-SFC DINAMICO", evolución del algoritmo desarrollado anteriormente en [8]. Como ya se describe en este artículo, la mayoría de investigaciones realizadas que rodean este problema se centran en la aplicación de enfoques como el de la Programación Lineal [9] o técnicas Greedy [10], pero existen muy pocas publicaciones que aborden el tema en cuestión desde el punto de vista de la computación cuántica y con la complejidad en las restricciones que se pretende resolver en este trabajo, donde se quiere conseguir obtener las soluciones óptimas globales, y de forma que la eficiencia de la resolución no escale de forma exponencial.

A destacar, lo realizado por B. Zjawin, B. Pointard, N. Claudet, N. Delorme y R. Ismailati en [12], donde resuelven el problema del camino más corto para un grafo simple empleando la formulación QUBO. Esta publicación, aunque no incluye las complicaciones relacionadas con los anchos de banda de los enlaces, los recursos de cada nodo o las funciones a adquirir, es un buen punto de partida para comprender desde un punto de vista técnico el planteamiento del trabajo descrito. Este estudio proporciona una buena base para entender cómo se pueden aplicar técnicas de computación cuántica a problemas de optimización de red, ofreciendo una perspectiva inicial sobre el potencial de estos métodos.

4.2. Principales Desafíos

En el panorama actual encontramos una serie de publicaciones que resuelven mediante métodos tradicionales el problema que nos ocupa (o una variante del mismo). También encontramos artículos introductorios a la formulación QUBO y la computación cuántica, incluso alguna publicación que utiliza estas técnicas para mostrar una posible resolución de un problema sencillo. Sin embargo, este trabajo se enfrentará a un problema cuyas instancias presentan una complejidad en cuanto a restricciones y parámetros a tener en cuenta que va más allá de lo visto hasta este momento.

Será por tanto el elemento diferenciador de este trabajo la cantidad de factores a tener en cuenta en la resolución del problema, y el acercamiento que ello supone con una aplicación directa sobre situaciones reales y prácticas, lo que lo hará distinto a lo publicado hasta la fecha.

4.3. Aportes de este TFG

Este Trabajo de Fin de Grado supone, por tanto, una ampliación al campo de investigación en cuestión, mostrando una posible resolución del problema del enrutamiento en una red con composición de cadenas de funciones virtuales haciendo uso de la computación cuántica. Se demostrará la viabilidad del enfoque cuántico, llevando más allá los límites de los problemas ya resueltos usando la formulación QUBO, aumentando la complejidad de los mismos y sirviendo como referencia para posibles futuros trabajos.

Lo realizado en este trabajo busca no solo demostrar la capacidad de resolución de problemas de optimización de la computación cuántica, sino además posicionarse como una alternativa válida en el manejo y enrutamiento en red actuales. Demostrando la capacidad de integración en el mundo empresarial de este tipo de soluciones llevando a una reducción sustancial en costes e inversión.

Finalmente se quiere destacar la importancia de la colaboración entre distintas disciplinas, más concretamente en este trabajo se han necesitado combinar conocimientos de redes, optimización y computación cuántica. Para poder aprovechar al máximo esta tecnología y que pueda ser escalable, es necesario que los profesionales que la implementen no solo formen parte

de los diversos campos que pueda requerir su aplicación, sino que mantengan una estrecha colaboración entre ellos. Los resultados obtenidos y las metodologías desarrolladas en este Trabajo de Fin de Grado quieren servir de inspiración y guía para investigaciones futuras, ayudando a acelerar la adopción de la computación cuántica en todas las disciplinas que puedan beneficiarse de ella.

5. Planificación

Resolver un problema como el propuesto requiere de una planificación temporal que divida en subproblemas más sencillos el problema inicial. De esta manera, se abordará cada subproblema por separado facilitando el manejo de su complejidad y garantizando un desarrollo más organizado y eficiente. El proceso puede ser descrito en varias fases, cada una con objetivos específicos y actividades claramente definidas:

- Adquisición de una base teórica: Es el proceso de formación inicial, en él se investigarán los principios básicos de la computación cuántica y sus características, se obtendrán los conocimientos necesarios para realizar una formulación QUBO correcta y acorde con nuestro problema y se aprenderán las diferencias entre el simulador y el solver híbrido.
- Formulación inicial: Con los conocimientos teóricos asentados, se llevará a cabo una primera formulación matemática que plasme los principales puntos a tener en cuenta de nuestro problema. No será necesario definir todas las restricciones, solo las más básicas para poder tener un modelo básico sobre el que trabajar y posteriormente mejorar.
- Implementación: Esta formulación matemática debe ser implementada en código haciendo uso de las bibliotecas pertinentes. Solo después de una implementación totalmente correcta se podrán realizar las primeras ejecuciones y valorar los cambios los resultados, mediante los que se inferirán los cambios oportunos si fuesen necesarios. Si la implementación no es correcta no se obtendrán soluciones válidas.
- Resolución con simulador: Después de asegurar que la implementación es correcta, se ejecutará el simulador para valorar los posibles cambios en la ponderación de los distintos elementos de la formulación, así como el ajuste de los parámetros del propio simulador.
- Reformulación: Llegado a este punto se ampliará la formulación obtenida hasta el momento para incluir ahora sí, todas aquellas restricciones del problema original para expresar la complejidad del mismo.
- Segunda Implementación: Contando con una formulación final, se deberá hacer lo mismo que tras la formulación inicial y plasmar en código el contenido de esta.

- Resolución con D-Wave: Tras la implementación final, se procederá a resolver el problema mediante el solver proporcionado por D-Wave en lugar del simulador, aunque se podrá usar este para contrastar datos en instancias de grafos pequeñas.
- Análisis y comparación: Tras la resolución con el solver de D-Wave se analizarán los resultados obtenidos.
- Revisión final: Tras cerciorarse de que se obtienen los resultados deseados, se revisará y dejará listo el código para su ejecución.
- Redacción de la memoria: Al concluir el trabajo se redactará esta misma memoria.

Finalmente, la distribución de tareas y el tiempo conllevado en cada una ha sido el siguiente:

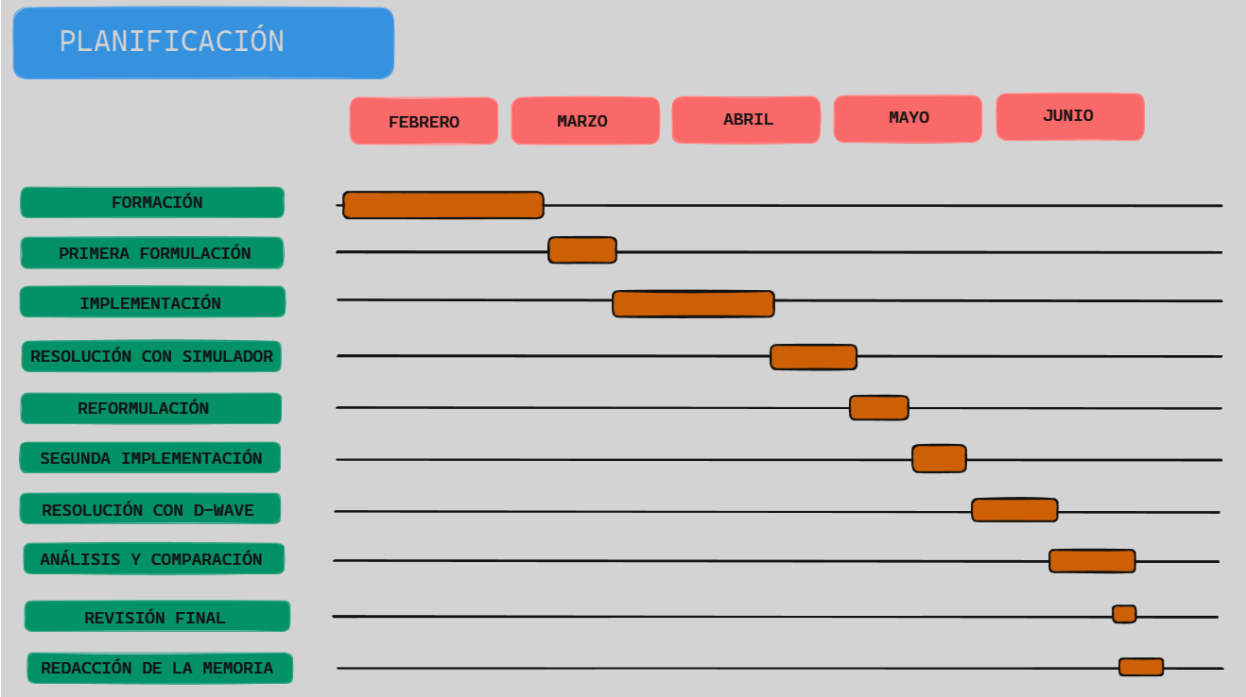

Figura 4: Diagrama de Gantt con la planificación seguida.

6. Formulación

Antes de comenzar con el desarrollo del código, se debe definir una formulación QUBO que nos permita expresar, en forma de variables binarias, cual será la función a minimizar y sus restricciones asociadas.
Para el problema de enrutamiento en cuestión, la prioridad será reducir el número de saltos realizados -siempre que no omitan las restricciones asociadas-. Estos saltos serán expresados en forma de matriz, donde el elemento $x_{i,j}$ valdrá 1 en caso de que el camino vaya del nodo *i* al *j*, y 0 en caso contrario. Además, estas matrices contarán con 3 dimensiones, siendo esta última la correspondiente al *agente* en cuestión. Los agentes representarán las distintas conexiones que tendremos en cuenta. Un mismo problema podrá ser resuelto para 1 agente o conexión, lo cual nos dará un camino único y óptimo (si lo hay). El mismo problema podrá ser resuelto para 3 agentes distintos, lo cual resultará en 3 soluciones independientes que podrán ser iguales si la red lo permite, o distintas entre sí si se deben adaptar a los cambios producidos en la red por los otros agentes.

Así pues, se podrá expresar el número de saltos dado de la siguiente forma:

$$CostFunction = \sum_{a=1}^{A} \sum_{i=1}^{N} \sum_{j=1}^{N} x_{i,j,a}$$

Donde *N* será el número de nodos que conforman la instancia del problema y A el número de agentes o conexiones.

Definida nuestra función a minimizar, se deben definir las restricciones asociadas que aseguran que las soluciones alcanzadas sean válidas. Comenzaremos con las restricciones que nos aseguran que del nodo inicial solo se sale, y que al nodo final solo se llega. Para comprobar que el camino sale del nodo inicial, se calcula el número de bits a 1 correspondientes a saltos con nodo *i* el inicial, y de manera similar se calculará el número de veces que se llega al nodo final fijando el nodo *j*. Para evitar ciclos, se requiere que ambos cálculos sean exactamente igual a 1, para asegurar que se sale una única vez del nodo inicial, y se llega una única vez al nodo final.

Matemáticamente:

$$\forall a \in \{1, 2, \dots\dots, A\}, \sum_{j=1}^{N} x_{InitialNode,j,a} = 1$$

Para la restricción del nodo inicial.

$$\forall a \in \{1, 2, \dots\dots, A\}, \sum_{i=1}^{N} x_{i,FinalNode,a} = 1$$

Para la restricción del nodo final.

Llamaremos a la restricción relacionada con el nodo inicial *Constrain*$_1$, y a la correspondiente con el final, *Constrain*$_2$. Al incluir estas restricciones en la formulación QUBO, es necesario hacerlo de tal forma que en caso de que una restricción se incumpla, se aumente el resultado final del cálculo del coste de energía de esa solución. Consecuentemente, las expresiones vistas anteriormente se incluirán en la formulación de la siguiente forma:

$$Constrain_1 = \sum_{a=1}^{A} \left(\sum_{j=1}^{N} x_{InitialNode,j,a} - 1 \right)^2$$

$$Constrain_2 = \sum_{a=1}^{A} \left(\sum_{i=1}^{N} x_{i,FinalNode,a} - 1 \right)^2$$

De esta manera, se asegura que para todas las soluciones en los que el cálculo de saltos desde el nodo inicial o hacia el nodo final no sea 1, la expresión resultante arroje un valor distinto de 0 y positivo, lo que contribuirá a aumentar el valor de energía calculado y por lo tanto a considerarse como una solución menos preferible.

Con estas restricciones sumadas a nuestra función de coste, el resultado obtenido sería el de exactamente dos saltos: uno desde el nodo inicial y otro hacia el final. Para formar caminos es necesario entonces indicar que se deben realizar el mismo número de saltos entrantes y salientes para todo nodo distinto del inicial y el final. Esto es:

$$\forall a \in \{1, 2, \dots\dots, A\}, \forall k \in \{1, 2, \dots, N\} \setminus \{InitialNode, FinalNode\},$$

$$\sum_{i=1}^{N} x_{i,k,a} = \sum_{j=1}^{N} x_{k,j,a}$$

Llamaremos a esta restricción, *Constrain₃*, y de la misma forma que vimos anteriormente, pasaremos las expresiones al mismo lado de la igualdad y elevaremos al cuadrado para asegurarnos sumar un entero positivo al resultado de la expresión en caso de que la restricción se incumpla. Por tanto, la expresión matemática de la restricción será:

$$Constrain_3 = \sum_{a=1}^{A} \sum_{\substack{k=1 \\ k \neq InitialNode \\ k \neq FinalNode}}^{N} \left(\sum_{i=1}^{N} x_{i,k,a} - \sum_{j=1}^{N} x_{k,j,a} \right)^2$$

Para terminar de definir las restricciones que nos permiten encontrar soluciones que formen caminos aceptados, debemos añadir una restricción que se asegure que no formamos ciclos, es decir, caminos con más de un salto en los que el nodo origen es igual que el destino. En la formulación QUBO, será necesario llevarlo más allá y asegurar que no se salta hacia ningún nodo ya visitado. Se puede para ello comprobar que el número de saltos desde un nodo determinado a cualquier otro sea menor o igual a 1, porque en caso contrario significa que habremos pasado más de una vez por dicho nodo desde el que se inicia el salto. Esto es:

$$\forall a \in \{1, 2, \ldots \ldots, A\}, \forall i \in \{1, 2, \ldots, N\}, \sum_{j=1}^{N} x_{i,j,a} \leq 1$$

Para expresar esto en forma de restricción, será necesario hacer uso de lo que se conoce como *"slack variables"*, que serán variables que podrá modificar el computador cuántico en función del valor de la restricción, para poder mantener la igualdad que deseamos. Estas variables podrán tomar valores dentro de un límite previamente establecido. Para el caso en cuestión, se definirá:

$$Constrain_4 = \sum_{a=1}^{A} \sum_{i=1}^{N} \left(\sum_{j=1}^{N} x_{i,j,a} + S - 1 \right)^2$$

Donde $S \in [0, 1]$, lo que significa que S toma valores en el intervalo cerrado de 0 a 1. En el caso de que un determinado $\sum_{j=1}^{N} x_{i,j}$ tome un

valor igual a 0 o 1 -cumpliendo así la restricción-, la variable S tomará el valor opuesto, asegurando que el resultado del paréntesis sea 0, lo que no añade penalización alguna. En caso de que el número de saltos desde un determinado nodo sea mayor que 1, la *slack variable* definida no pueden compensarlo de manera alguna lo que da como resultado un entero positivo que se sumará al resultado en forma de penalización.

En este punto, se puede llegar a pensar que las soluciones ofrecidas que respeten las anteriores restricciones se pueden llegar a considerar válidas, sin llegar a incluir posibles condiciones adicionales como más adelante veremos (ancho de banda de los enlaces, recursos de cada nodo y funciones solicitadas). Sin embargo, existe la posibilidad de que se encuentren algunas soluciones que no sean continuas en el camino trazado, y que presenten un ciclo: las soluciones con ciclos de 2 nodos. Las características de estos ciclos hacen que no se entre a ninguno de los dos nodos más de una vez, y que en ambos se entre y salga el mismo número de veces. Por eso, se debe definir una quinta restricción específica que contemple este caso. Se comprobará entonces que para toda pareja de nodos, el producto de los bits de salto correspondientes es igual a 0, ya que en caso de que se salte desde un nodo *i* a un nodo *j* y viceversa el resultado del producto será 1. Expresado matemáticamente:

$$\forall a \in \{1, 2, ... \ldots, A\}, \forall i \in \{1, 2, \ldots, N\}, \forall j \in \{1, 2, \ldots, N\},$$

$$x_{i,j,a} * x_{j,i,a} = 0$$

Lo que expresaremos en forma de restricción como:

$$Constrain_5 = \sum_{a=1}^{A} \sum_{i=1}^{N} \sum_{j=1}^{N} (x_{i,j,a} * x_{j,i,a})^2$$

Llegado a este punto, las soluciones encontradas serían continuas y sin formar ningún ciclo. Sin embargo, el problema a resolver debe encontrar además caminos que nos permitan solicitar una serie de funciones determinadas distribuidas a lo largo de los nodos del grafo. A esto se le suma que el grafo puede cambiar con el tiempo, dejando algunos de los enlaces o nodos deshabilitados.

Así pues, comenzaremos definiendo la restricción relacionada con la capacidad o ancho de banda de cada enlace. El ancho de banda de cada enlace

será un valor conocido ya que formará parte de la definición del problema a resolver. Debido a las características de la formulación QUBO, solo podemos expresar si un determinado salto ha sido realizado o no (y por tanto si el enlace entre los nodos que lo componen ha sido utilizado), y llevar un recuento entre el número de agentes que lo han realizado. Para poder expresar ese coste que supone utilizar un enlace, y como el uso de un enlace decrementa en esta cantidad su ancho de banda, lo que se hará será dividir las conexiones entre los nodos por este valor, y quedarnos con la parte entera. Esto dará el número de veces que se puede pasar por cada uno. Por ejemplo, si el coste de usar un enlace son 2 unidades, y tenemos un determinado enlace con una capacidad de 7, dividimos 7 entre 2 y nos quedamos con la parte entera, lo cual da 3, que será el número de veces que se puede pasar por este enlace con ese coste de uso. Haremos esta operación para todos los enlaces del grafo. Debemos comprobar que para cada enlace *i-j*, el número de saltos total entre todos los agentes simultáneos sea igual o inferior al número de veces que podemos pasar por ellos. Matemáticamente:

$$\forall i \in \{1, 2, \ldots, N\}, \forall j \in \{i+1, i+2 \ldots \ldots, N\}, \sum_{a=1}^{A} (x_{i,j,a} + x_{j,i,a}) \leq W$$

Donde W corresponderá al número de veces que podemos pasar por cada enlace.

Utilizando las ya vistas *slack variables*, podemos definir la restricción de la siguiente forma:

$$Constrain_6 = \sum_{a=1}^{A} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \left(\sum_{a=1}^{A} x_{i,j,a} + S - W \right)^2$$

Cada vez que un agente pase por un determinado nodo, los recursos de este nodo decrementarán en una unidad. En caso de que estos recursos se hayan reducido hasta llegar a 0, el nodo será inhabilitado y dejará de considerarse como una conexión disponible. De forma similar a la sexta restricción, definiremos una séptima para comprobar que no se pasa por un nodo al que se le han agotado los recursos. En este caso comprobaremos, para cada nodo, el número de veces que se salta hacia él entre todos los agentes, lo cual debe ser menos que la capacidad del propio nodo -asumiendo que cada vez que se pasa por él se reduce dicha capacidad en 1-. Matemáticamente:

$$\forall j \in \{1, 2, \ldots, N\}, \left(\sum_{i=1}^{N} \sum_{a=1}^{A} x_{i,j,a} \leq C \right)$$

Por tanto, expresaremos la séptima restricción de la siguiente forma:

$$Constrain_7 = \sum_{j=1}^{N} \left(\sum_{i=1}^{N} \sum_{a=1}^{A} x_{i,j,a} + S - C \right)^2$$

Donde S será la *slack variable* empleada.

Como última restricción, se debe definir la correspondiente a las funciones solicitadas. En este caso, se debe comprobar que para cada función solicitada, se pasa al menos una vez por alguno de los nodos que la contengan, esto es:

$$\forall a \in \{1, 2, \ldots \ldots, A\}, \forall f \in \{1, 2, \ldots, F\},$$
$$\left(\sum_{i=1}^{N} (\sum_{j=FunctionNodes}^{Nf} x_{i,j,a} + \beta \geq 1) \right)$$

Donde β valdrá 1 en caso de que el nodo inicial contenga la función en cuestión, y 0 en caso contrario. Este valor es necesario debido a que en la restricción se comprueban todos los nodos con función a los que llegamos, pero excluiría el caso en el que la función se encuentra en el nodo del que partimos. La expresión matemática de la restricción será:

$$Constrain_8 = \sum_{a=1}^{A} \sum_{f=1}^{F} \left(\sum_{i=1}^{N} \sum_{j=FunctionNodes}^{Nf} x_{i,j,a} + \beta - S - 1 \right)^2$$

Donde S será la *slack variable* empleada y β la comprobación de la función en el nodo inicial.

Definidas todas las restricciones y la función de coste, se unirán en una única expresión de la forma:

$$QUBO = \alpha_1 * CostFunction + \alpha_2 * Constrain_1 + \alpha_3 * Constrain_2 ...$$

Donde cada α será una constante que se deberá ajustar en función de diversas características de la instancia de nuestro problema, como el tamaño

del grafo o el número de agentes. Esto es así porque al usar variables binarias el máximo valor que pueden tomar es 1. En problemas grandes, saltarse una restricción con la penalización que conlleva, a cambio de reducir la función de coste puede dar lugar a una solución de menor energía. Para evitar esto las constantes asociadas deben ser ajustadas al problema concreto a resolver.

7. Configuración Previa a los Experimentos

7.1. Entorno

Para poder ejecutar el código desarrollado junto con el simulador y las bibliotecas necesarias, se han seguido los siguientes pasos:

- Descargar el instalador de https://www.anaconda.com/downloaddownloads.
- Instalar Anaconda. Anaconda es una distribución gratuita y de código abierto de Python que incluye una gran cantidad de paquetes y herramientas, lo que facilita la gestión de entornos y dependencias.
- Crear un nuevo ”environment” con la version de python apropiada, para ello abrir el Prompt de Anaconda en Windows, o un terminal normal en Linux o Mac y ejecutar la siguiente instrucción: conda create -n “expertcourse”python=3.10.13. Un environment en Anaconda es un entorno aislado que tiene su propia instalación de Python y paquetes específicos lo que facilita el trabajo en diferentes proyectos con diferentes dependencias sin tener conflictos.
- Descargar el fichero dadk-light-3.10.tar.bz2 y colocarlo en la misma ruta que el fichero de requerimientos. Esta biblioteca será la correspondiente al simulador cuántico que se usará en las primeras fases del desarrollo, y no está incluída en Anaconda, por lo que debe descargarse por separado.
- Instalar el fichero de requerimientos con pip install -r ”requirements.txt”. Este fichero contiene una lista de dependencias necesarias para que el proyecto funcione correctamente. Esta acción instalará todas las dependencias en un único comando.

Llegado a este punto se puede utilizar el editor de código de preferencia (recomendado VS Code) asegurando cambiar el environment en cuestión. El resto de bibliotecas que pudiesen ser necesarias se instalarán usando pip con la forma: pip install *biblioteca*.

7.2. Datos

Los problemas a resolver y sus características se encuentran representados en una serie de ficheros .DAT.

En ellos, se representa la información básica de la instancia del problema con el que se trabajará. Concretamente, para las conexiones se empleará una matriz cuyos elementos valdrán 0 en caso de no existir conexión entre los elementos de la fila y columna correspondientes, u otro valor entero positivo para indicar la capacidad del enlace entre dichos elementos. Esta matriz será simétrica puesto que se considerará que los enlaces son bidireccionales.

```
0  5  7  0  0  0
5  0 10  5  0  0
7  10 0  0  2  0
0  5  0  0  8  6
0  0  2  8  0  3
0  0  0  6  3  0
```

Figura 5: Conexiones del grafo. El elemento *i,j* representa el ancho de banda del enlace entre el nodo *i* y el *j*.

En cuanto a las funciones de cada nodo, el archivo correspondiente contará con una fila de enteros positivos separados por espacios indicando las funciones almacenadas por el nodo de la fila correspondiente.

```
1 5 0 0 0 0
2 4 0 0 0 0
3 6 0 0 0 0
1 6 0 0 0 0
2 3 4 0 0 0
5 0 0 0 0 0
```

Figura 6: Funciones del grafo. Cada fila *i* representa las funciones del nodo *i*.

Finalmente, para los recursos de cada nodo se contará con un fichero formado por una única fila de enteros positivos indicando los recursos de cada nodo de forma ordenada, comenzando por el nodo 1 y acabando en el último nodo del grafo en cuestión.

```
4 5 5 5 4 5
```

Figura 7: Recursos del grafo. El elemento *i* representa los recursos del nodo *i*.

Concretamente en este trabajo, y, excluyendo los grafos iniciales desarrollados como prueba, se han definido en total 6 archivos para formar 2 instancias del problema, una de 6 nodos y otra de 19 nodos:

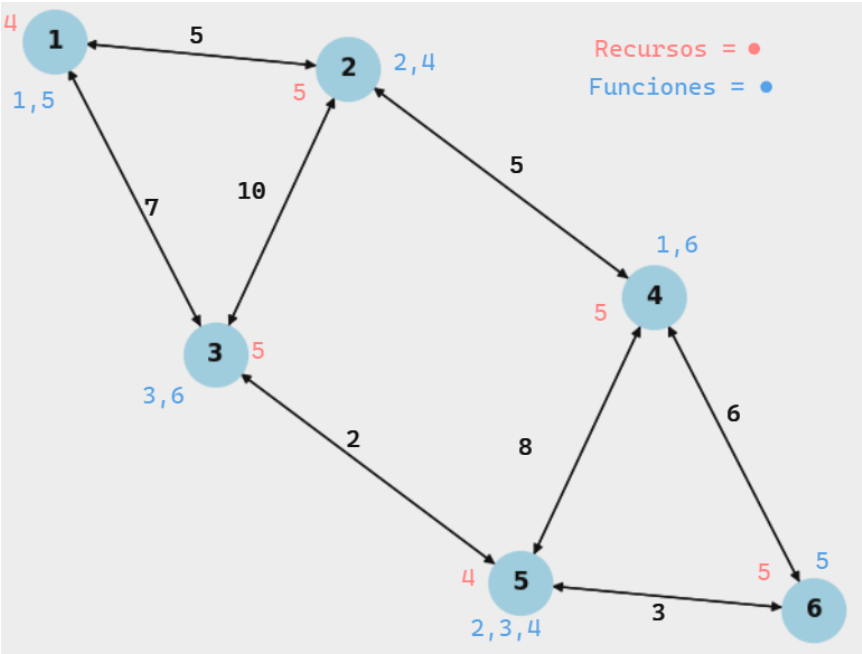

Figura 8: Grafo de 6 nodos.

Este grafo de 6 nodos será el grafo inicial con el que se harán las pruebas con la implementación de la primera formulación y con el simulador, aunque más tarde será resuelto también con la formulación e implementación final, y con el solver híbrido de D-Wave.

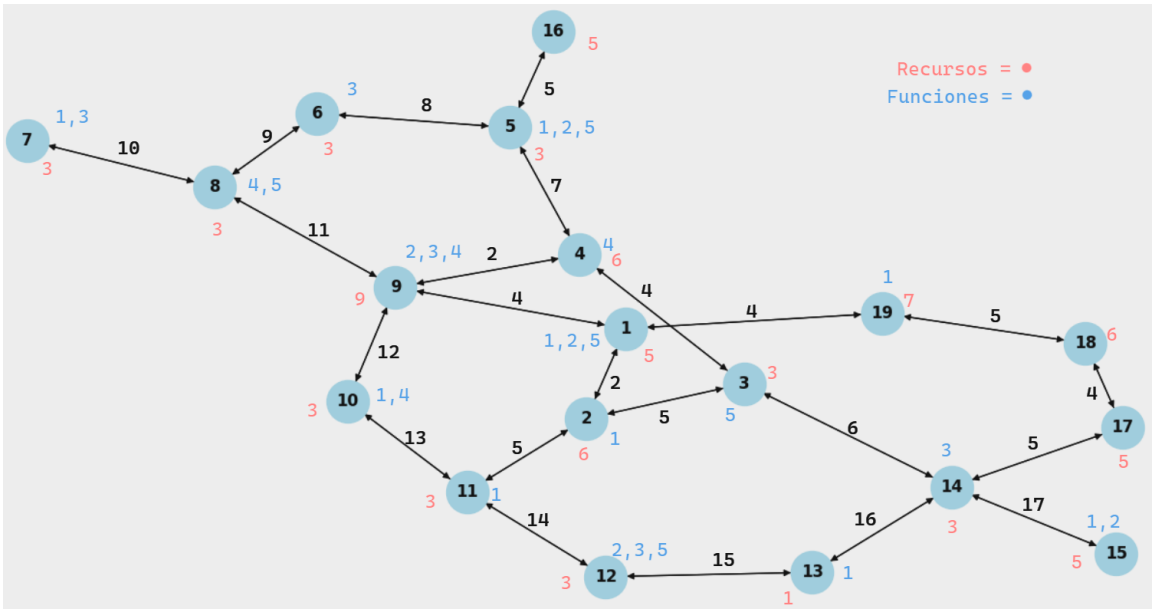

Figura 9: Grafo de 19 nodos.

El grafo de 19 nodos servirá para aumentar el tamaño del problema tras asegurar que lo desarrollado para el grafo anterior es correcto. Este aumento en el tamaño implica una mayor complejidad relacionada principalmente con el cálculo de las constantes asociadas a las restricciones y la función de coste, pues aunque las expresiones matemáticas no se modifiquen, un problema más grande, y por tanto soluciones con un valor de energía por lo general más alto, implican un aumento en las penalizaciones que supone saltarse alguna de estas restricciones.

8. Resultados

Para poner a prueba la formulación y el código desarrollados, se presentan problemas de distinto tamaño y características. Para todos los casos, cada conexión a un nodo restará en 1 la capacidad del mismo. Por ejemplo, si un nodo *a* tiene una capacidad inicial de 5, cuando se realice una conexión al nodo este rebajará su capacidad a 4 para las posibles futuras conexiones que se establezcan, hasta poder llegar a reducirse a 0 y quedar inhabilitado. De forma similar que con los recursos de los nodos, el ancho de banda de los enlaces también se verá decrementado cada vez que una conexión los emplee pudiendo llegar a quedar inhabilitados, aunque en este caso el decremento será de 2 unidades.

8.1. Grafo de 6 nodos

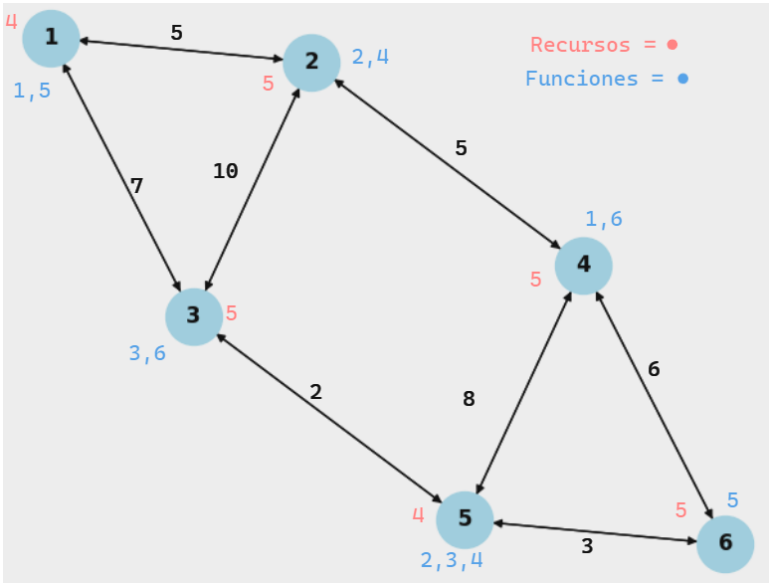

Figura 10: Grafo de 6 nodos.

En este problema ilustrado en la figura 10 se emplea un grafo de 6 nodos en el que definiremos como nodo inicial el nodo 1, y como nodo final el nodo 6. Las funciones requeridas son la 3 y 4. Como podemos ver, la función 3 se encuentra en los nodos 3 y 5 y la función 4 en los nodos 2 y 5. Estos requisitos junto con los costes de conexión se detallan en la Tabla 1.

N. Inicial	N. Final	F. Solicitadas	Coste Conexión al Nodo	Coste de Uso de Enlace
1	2	[3, 4]	1 unidad	2 unidades

Cuadro 1: Requisitos de la solución y costes definidos.

Para un único agente, se obtiene el resultado mostrado en la figura 11:

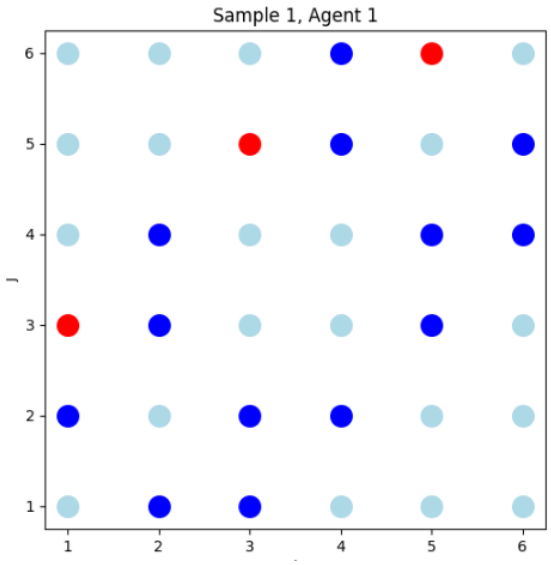

Figura 11: Solución para un agente.

Las soluciones obtenidas son devueltas en una serie de diccionarios y han sido interpretadas de forma gráfica en forma de matriz para facilitar su visualización. La forma de leer los resultados será comenzando desde la columna correspondiente al nodo inicial, en este caso, la primera columna. Desde ahí, observamos a qué nodo lleva el siguiente salto, el cual estará iluminado en rojo. Sabiendo el siguiente nodo al que se salta, nos situamos en la columna correspondiente a dicho nodo y repetimos el proceso hasta llegar al nodo final. Los nodos en azul oscuro serán aquellos con los que la conexión es posible pero no se ha llevado a cabo, y los nodos en azul claro serán aquellos saltos que serán imposibles por la propia definición de la red (nodos no conectados). Se observa en 11 como el camino descrito será:

$$1 \rightarrow 3 \rightarrow 5 \rightarrow 6$$

Que será el camino óptimo para nuestro problema.

Si modificamos los recursos disponibles del nodo 3 o reducimos la capacidad de los enlaces que llevan a este nodo, lo conseguiremos aislar y forzar un camino distinto.

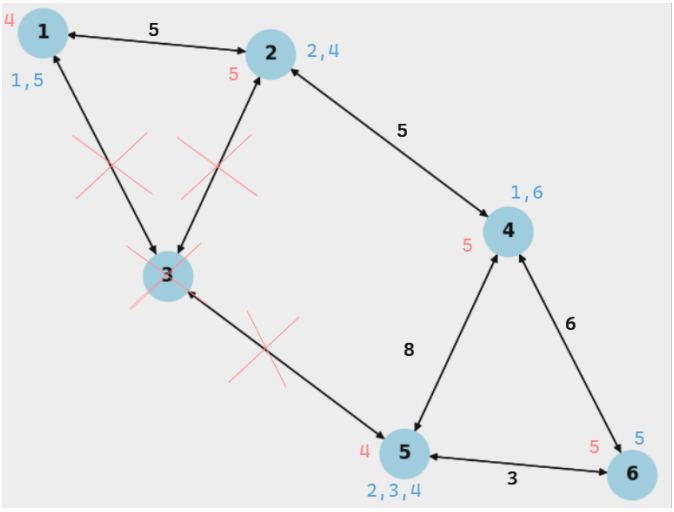

Figura 12: Nuevo grafo.

Como vemos en la figura 12, se ha eliminado por completo tanto el nodo 3 como cualquier enlace que pudiese llevar a dicho nodo, aislándolo por completo y forzando que las soluciones obtenidas únicamente tengan en cuenta los nodos 1, 2, 4, 5 y 6, y los enlaces que los unen.

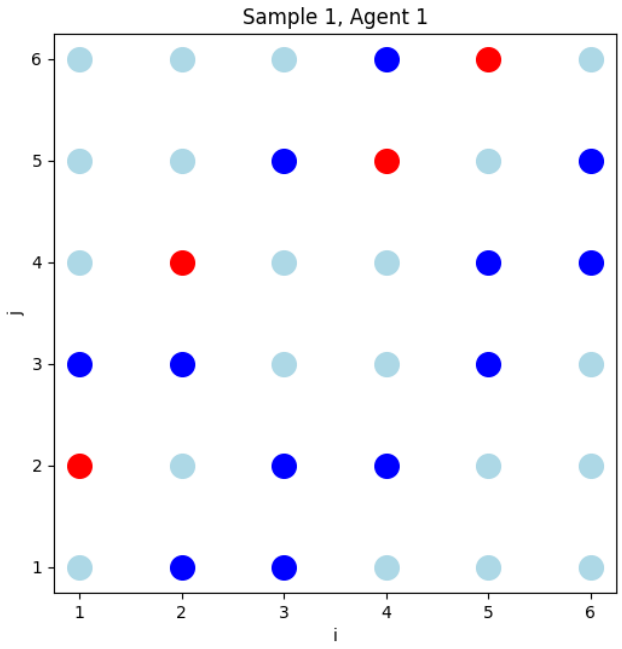

Figura 13: Nueva solución para 1 agente.

Vemos en la figura 13 como al aislar el nodo 3 se fuerza el camino:

$$1 \rightarrow 2 \rightarrow 4 \rightarrow 5 \rightarrow 6$$

Que sigue reuniendo las funciones deseadas, pero que realiza un salto extra al haber inhabilitado el nodo 3.

Vamos ahora a valorar el mismo problema para un número mayor de agentes. La introducción de nuevos agentes implica más conexiones en la red. Estos agentes actuarán de forma secuencial aunque el orden de salida será establecido por el ordenador cuántico o el simulador cuanto ejecute la petición mandada. Por ello aunque en este caso el agente 1 es el que ha obtenido la solución de menor coste, y por tanto ha sido el primero en ejecutarse, en otras ocasiones es posible que sea el agente 2 o el 3 el que obtenga dicha solución. En cuanto al cálculo de las soluciones, cada agente tiene sus variables asociadas, por lo que las variables que describen el comportamiento de cada uno de ellos afectan a la valoración de los recursos de los nodos y los enlaces para los demás.

Esto significa que si un agente ha pasado por un determinado enlace y nodo, el resto de agentes tendrán en cuenta dicho paso con lo que cada uno podrá valorar de forma independiente que camino escoger. En cuanto a las funciones solicitadas, aunque todos comparten la lista de funciones requeridas, cada uno de los agentes debe obtenerlas en su totalidad de forma independiente, por lo que la obtención de una función x para el agente a no influirá en el agente b, que deberá asegurarse de obtener la misma función x a lo largo de su propia solución.

Si volvemos a la configuración original del problema e introducimos 2 agentes adicionales, se obtiene la solución mostrada en la figura 14:

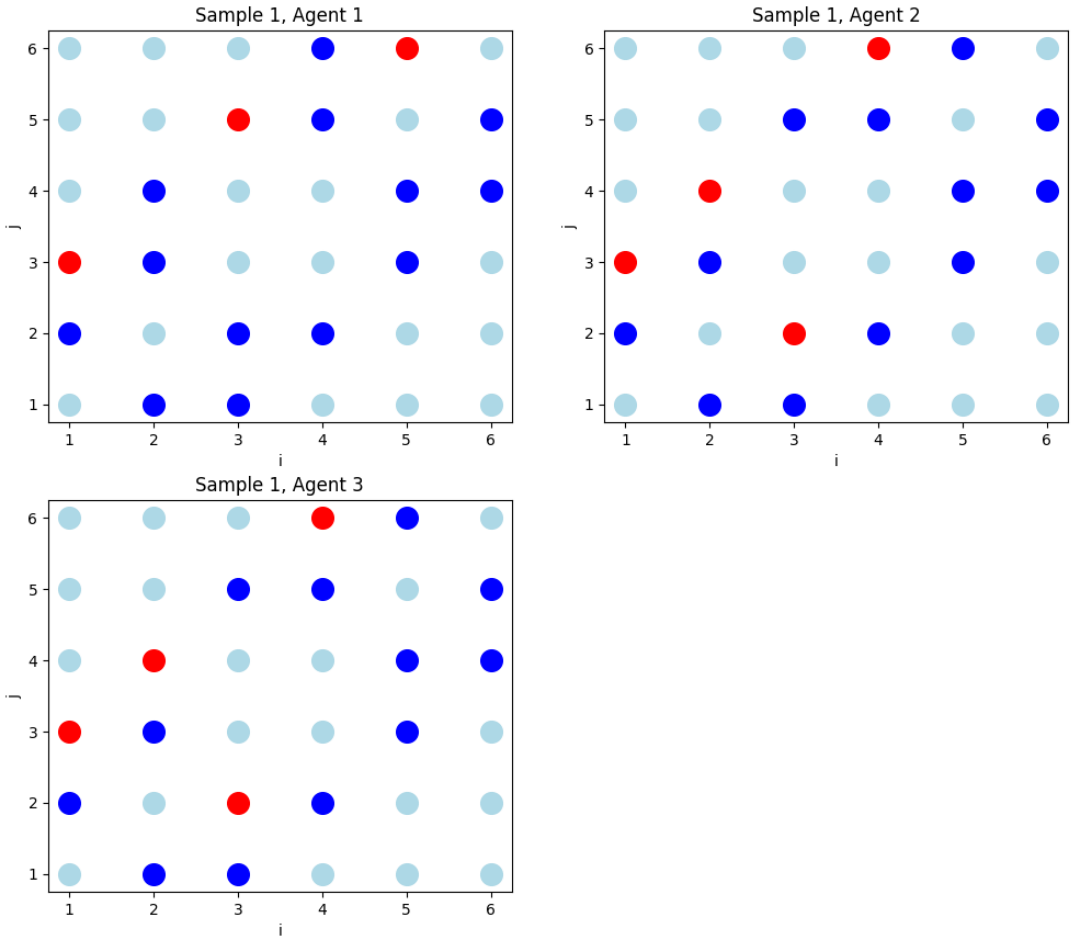

Figura 14: Solución para tres agentes.

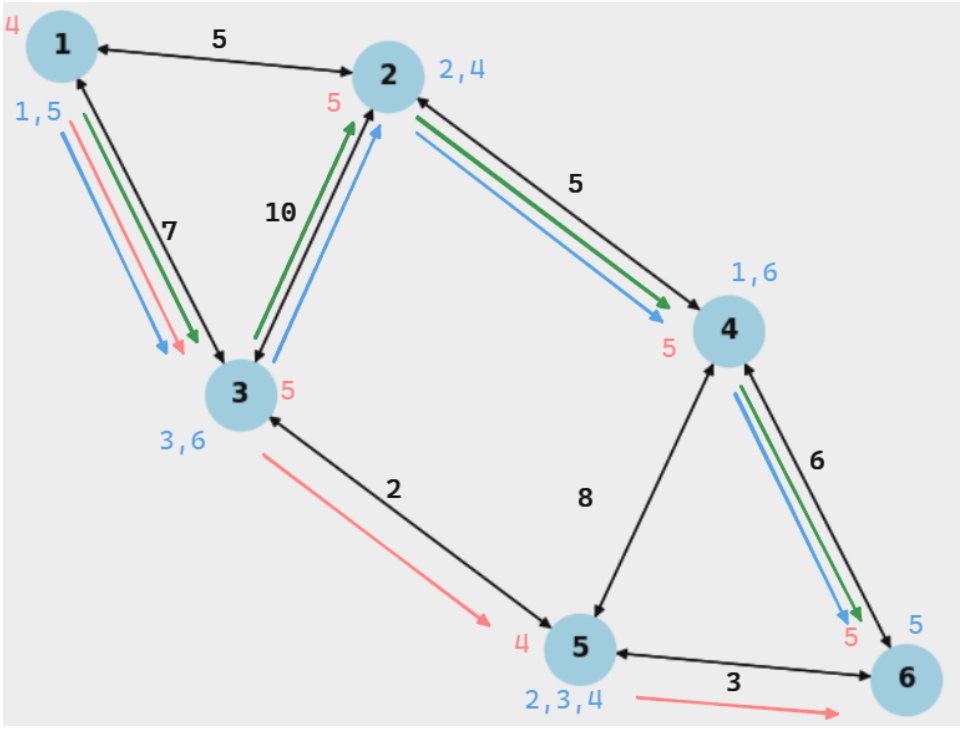

Figura 15: Caminos descritos por los tres agentes.

Como se observa en los caminos descritos en la figura 15, una de las soluciones es equivalente a la vista anteriormente. Sin embargo, al introducir más agentes es necesario cambiar la ruta ya que el enlace entre el nodo 3 y 5, así como el enlace entre el nodo 5 y 6 habrán dejado de estar disponibles (tienen 2 y 3 de capacidad respectivamente, con unas conexiones que requieren de 2 unidades de ancho de banda, por lo que solo se puede pasar una vez por ellos). Por esto, ofrece un camino alternativo que si bien aumenta el número de saltos a realizar, no rompe ninguna restricción:

$$1 \rightarrow 3 \rightarrow 2 \rightarrow 4 \rightarrow 6$$

Este camino se produce en 2 de los 3 agentes ya que los enlaces y recursos de cada nodo lo permiten.

Los enlaces no son lo único que se puede agotar pues los nodos pierden recursos cada vez que un agente pase por ellos. Si el paso de los agentes por un determinado nodo lo inhabilita, los siguientes agentes buscarán una ruta alternativa que siga cumpliendo las restricciones definidas y tratando de minimizar la función de coste. Por ejemplo, si se modifica el archivo correspondiente a los recursos de los nodos, modificando el valor correspondiente al nodo 3 para que solo pueda servir a un agente:

Figura 16: Recursos modificados.

Y solicitamos las funciones 3 y 6 para 2 agentes, es decir, modificamos ligeramente la tabla inicial de requisitos a la siguiente:

N. Inicial	**N. Final**	**F. Solicitadas**	**Coste Conexión al Nodo**	**Coste de Uso de Enlace**
1	2	[3, 6]	1 unidad	2 unidades

Cuadro 2: Nuevos requisitos de la solución y costes definidos.

Para esta configuración se obtiene la solución de la figura 17.

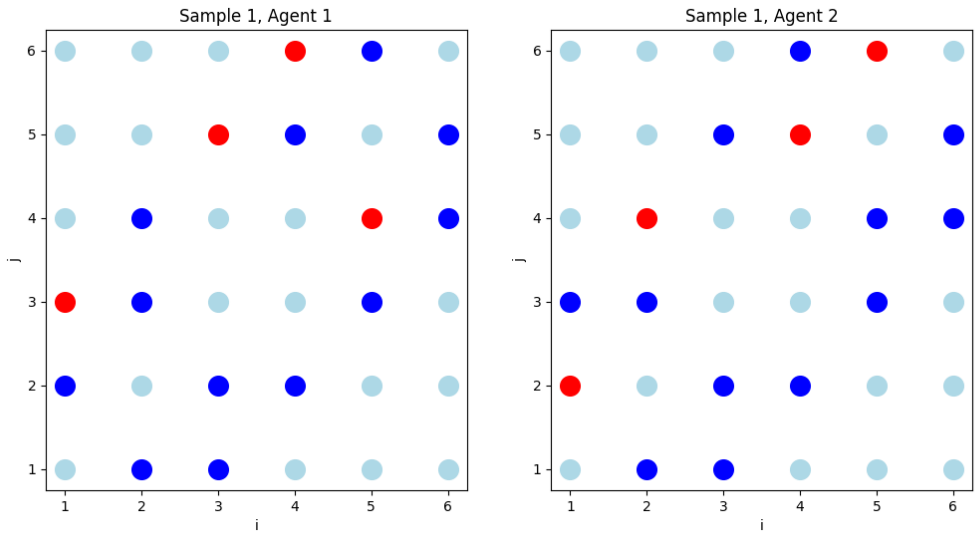

Figura 17: Solución para 2 agentes.

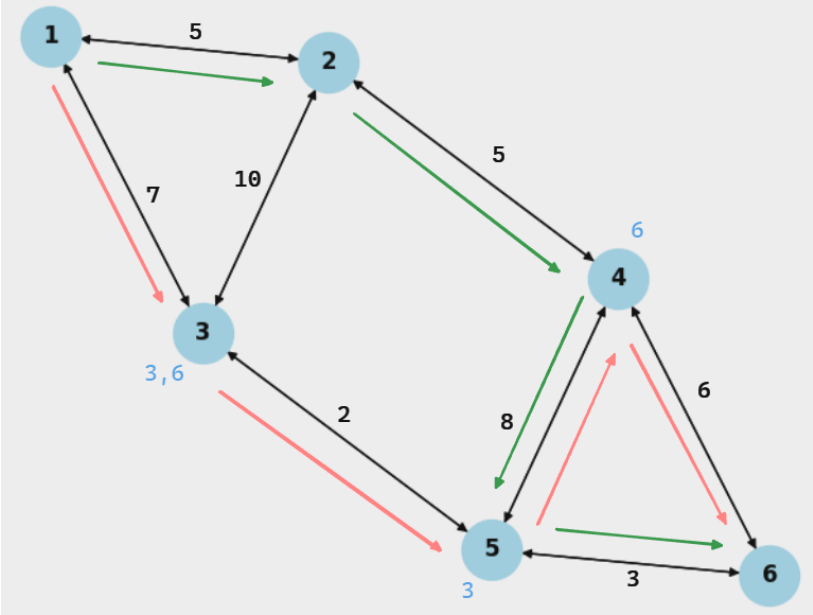

Figura 18: Caminos descritos por los 2 agentes.

La solución mostrada en la figura 18 ofrece 2 caminos:

$$1 \rightarrow 3 \rightarrow 5 \rightarrow 4 \rightarrow 6$$

Con lo que se inhabilitaría el nodo 3, así como el enlace 3-5. Y:

$$1 \rightarrow 2 \rightarrow 4 \rightarrow 5 \rightarrow 6$$

En un principio se puede pensar como primera opción en el camino:

$$1 \rightarrow 3 \rightarrow 5 \rightarrow 6$$

Sin embargo esta opción inhabilitaría el enlace 5-6, forzando en el siguiente agente el camino:

$$1 \rightarrow 2 \rightarrow 4 \rightarrow 5 \rightarrow 4 \rightarrow 6$$

Que aunque recoge todas las funciones, pasaría 2 veces por el nodo 4, lo cual se encuentra penalizado por la cuarta restricción.

8.2. Grafo de 19 nodos

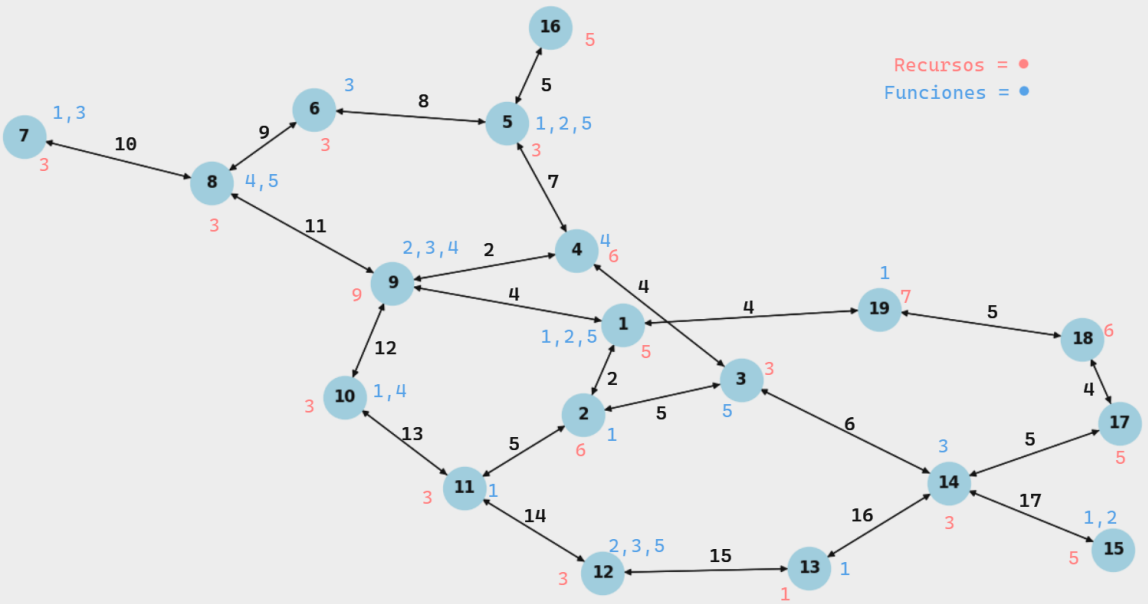

Figura 19: Grafo de 19 nodos.

Se utilizará el grafo mostrado en la figura 19 para ilustrar el comportamiento de los agentes en una instancia del problema más compleja. Se definen como nodos iniciales y finales el 2 y el 11 respectivamente. Las funciones a recolectar serán la 2, 3 y 4. El coste de conexión será a cada nodo será de 1 unidad y el de uso de cada enlace de 2 unidades. En la tabla 3 se reúnen todos estos requisitos y costes:

N. Inicial	N. Final	F. Solicitadas	Coste Conexión al Nodo	Coste de Uso de Enlace
2	11	[2, 3, 4]	1 unidad	2 unidades

Cuadro 3: Requisitos de la solución y costes definidos.

Para un único agente, el camino realizado será el más corto hasta el nodo final que le permita recolectar las funciones en cuestión. La solución obtenida se muestra en la figura 20:

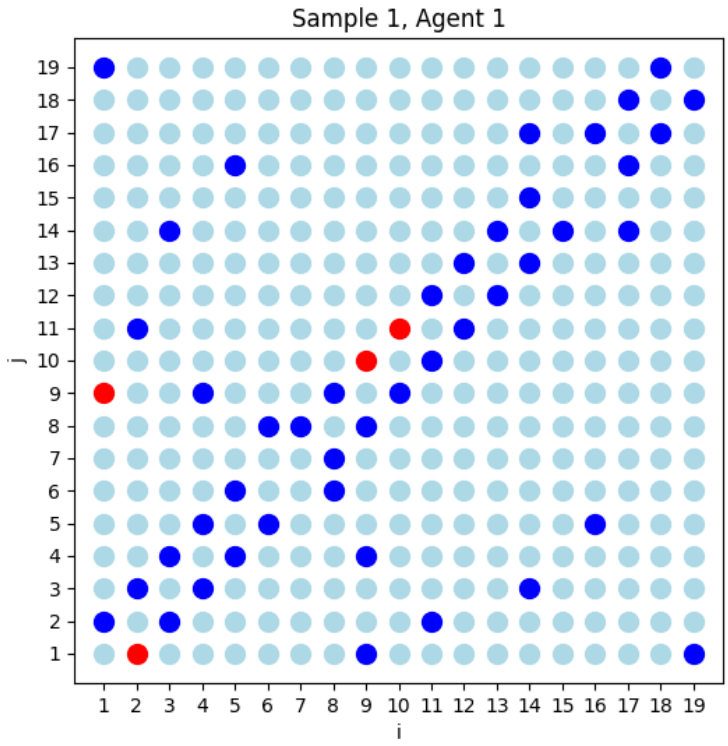

Figura 20: Camino óptimo para 1 agente.

Donde vemos como el camino óptimo tiene un coste de 4 saltos.

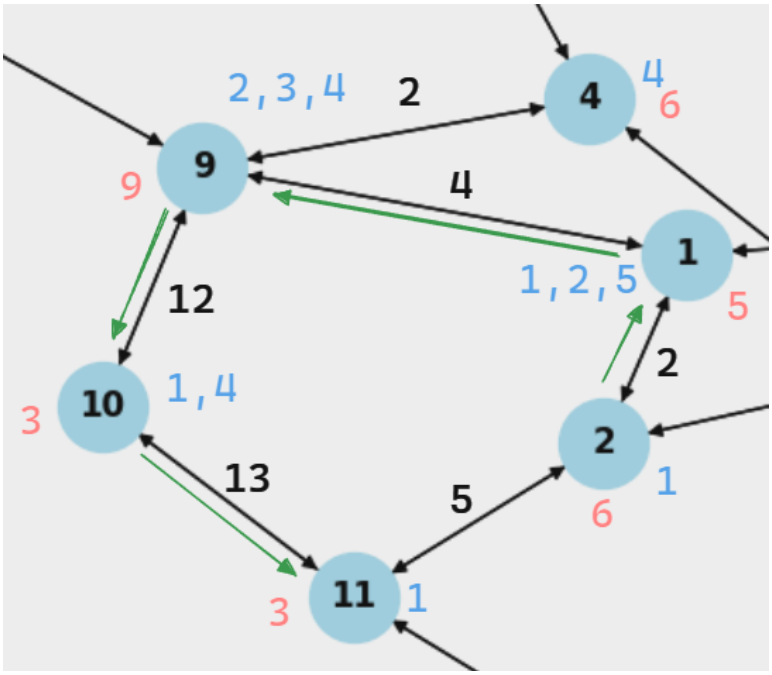

Figura 21: Representación gráfica del camino óptimo para 1 agente.

El camino descrito por el agente y mostrado gráficamente en la figura 21 es el siguiente:

$$2 \rightarrow 1 \rightarrow 9 \rightarrow 10 \rightarrow 11$$

Es el óptimo para un único agente, pues se recogen todas las funciones deseadas y no se rompe ninguna restricción. Sin embargo, es importante destacar como el enlace que une los nodos 2 y 1 tiene 2 unidades de capacidad. Habiendo definido el ancho de banda consumido por los agentes en los enlaces como 2 unidades también, este enlace quedaría inhabilitado tras el paso del agente. Si se aumenta el número de agentes y se solicitan las mismas funciones y nodos iniciales y finales, el camino escogido por el segundo agente deberá forzosamente ser distinto, como se muestra en la figura 22.

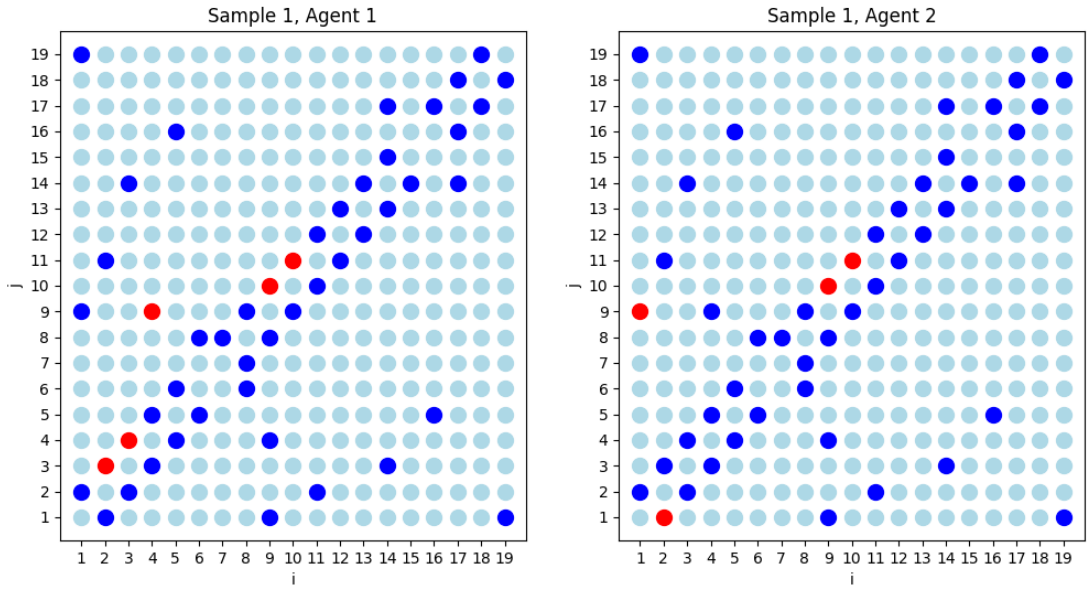

Figura 22: Camino óptimo para 2 agentes.

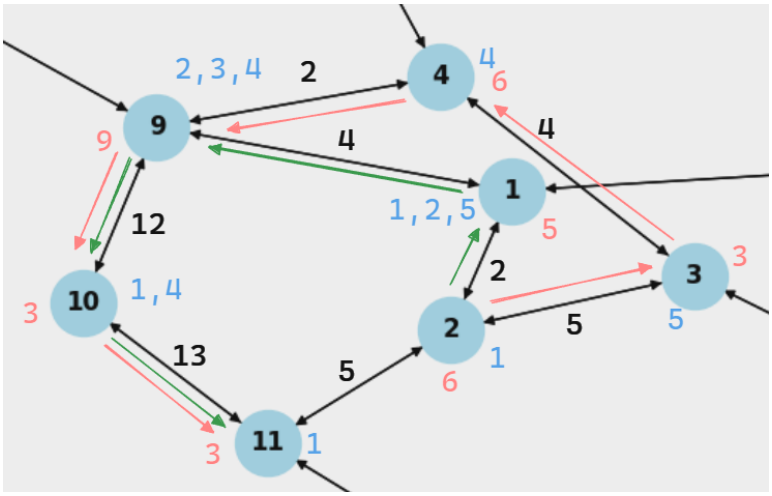

Figura 23: Representación gráfica del camino óptimo para 2 agentes.

Como se observa en la figura 23, a la solución anteriormente vista se le suma una nueva descrita por el segundo agente:

$$2 \rightarrow 3 \rightarrow 4 \rightarrow 9 \rightarrow 10 \rightarrow 11$$

Resultado de buscar una alternativa tras la inhabilitación del enlace entre el nodo 2 y 1. Este camino a su vez inhabilita el enlace entre el nodo 4 y el 9.

Si se añade un agente más al problema anterior, este último debe buscar un camino alternativo a los seguidos por los otros dos agentes ya que algunos de los enlaces en ambas soluciones han sido eliminados. La solución obtenida es la siguiente:

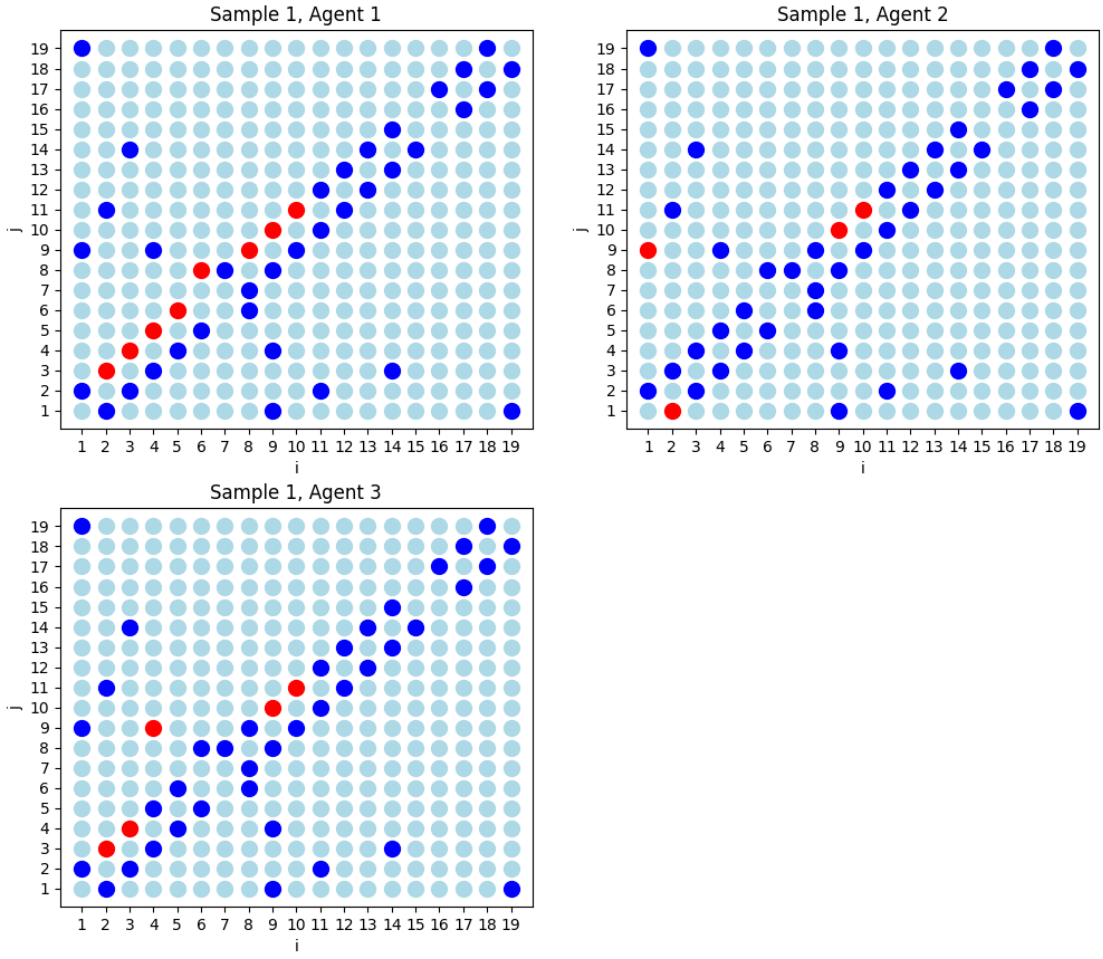

Figura 24: Camino óptimo para 3 agentes.

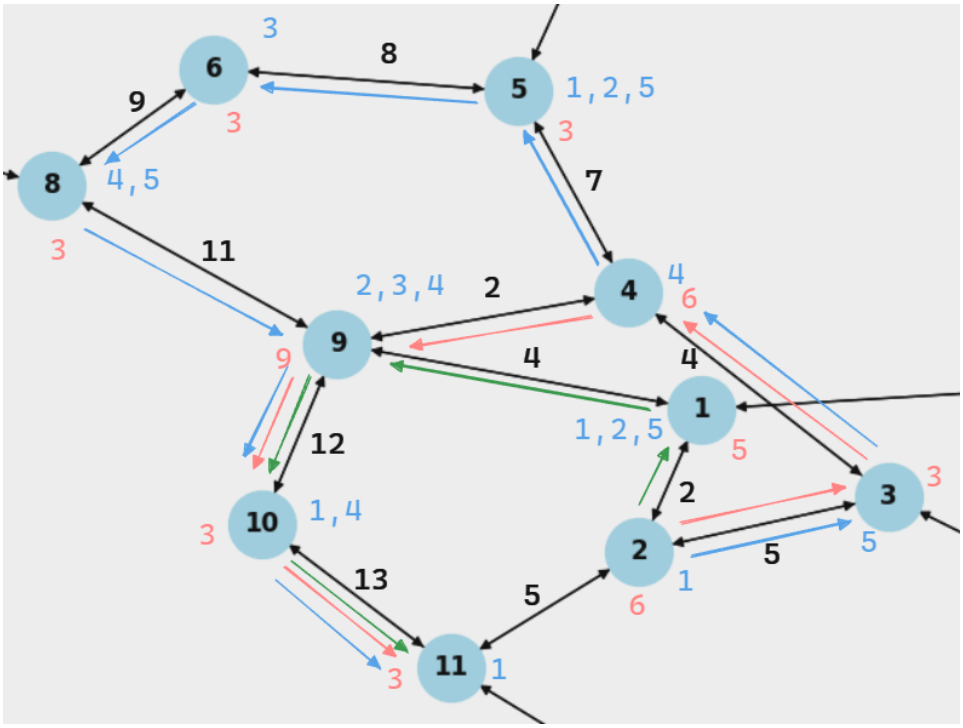

Figura 25: Representación gráfica del camino óptimo para 3 agentes.

El resultado de añadir un agente más es una nueva solución aún más larga para recolectar las funciones requeridas y llegar al nodo final sin romper las restricciones, como se muestra en la figura 25. Esta solución busca una alternativa al no disponer de los enlaces suprimidos por el paso de los otros dos agentes, describiendo el camino:

$$2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow 11$$

Es importante destacar que los resultados ofrecidos por el solver híbrido-de D-Wave pueden fluctuar para un mismo polinomio y constantes. Estas constantes en la práctica son a menudo calculadas con técnicas de machine learning o post-procesados, algo a lo que no se ha tenido acceso para este trabajo, por lo que se avisa que para el grafo de 19 nodos, y el caso de 3 agentes, los resultados pueden ser imprecisos.

8.3. Ajuste del solver y tiempos

Para los casos vistos anteriormente se ha empleado el solver híbrido proporcionado por D-Wave. Además, en el caso del grafo de 6 nodos, se ha usado también el simulador para contrastar resultados. Los tiempos de ejecución se encuentran definidos desde el comienzo. En el caso del solver híbrido se ajustan manualmente los segundos durante los cuales se procesará nuestra petición. De este tiempo, un pequeño porcentaje será tiempo de uso de un computador cuántico puro, y el resto será un refinamiento del resultado mediante computación clásica. Este reparto no puede ser modificado. En cuanto al simulador, al crear el objeto solver se pueden definir tanto el número de iteraciones como ejecuciones, así como la precisión utilizada en los cálculos. Si no se especifican, el objeto se creará con unos valores por defecto. Para este trabajo, se ha establecido el número de iteraciones y ejecuciones en 200000 y 30 respectivamente. Como resultado a estos parámetros el tiempo de ejecución, ya sea en el simulador o en el solver híbrido será siempre el mismo.

Si se dispone de una cuenta en D-Wave Leap -lo que proporcionará el token de acceso a los solvers-, se proporcionará no solo un registro de las ejecuciones solicitadas, sino también un breve resumen de cada una de ellas. Entre esa información se encuentra el tiempo de ejecución y su distribución entre los distintos computadores. Para el caso del grafo de 6 nodos y 3 agentes, estos han sido los tiempos empleados:

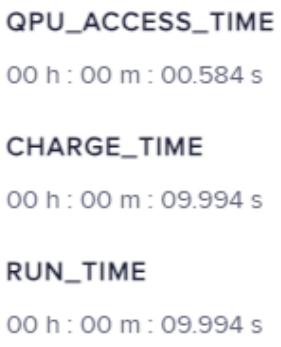

Figura 26: Tiempos de ejecución para el grafo de 6 nodos y 3 agentes.

Como se observa, de los 10 segundos establecidos para la ejecución de este problema, solo 0.584 han sido en la ”Quantum Processing Unit”, dis-

tribuyendo el resto entre la carga y el postprocesado del resultado.

En el caso del grafo de 19 nodos y 3 agentes:

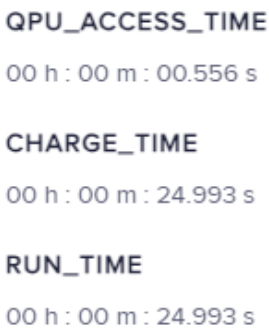

Figura 27: Tiempos de ejecución para el grafo de 19 nodos y 3 agentes.

Similar a lo que ocurre en el caso anterior, de los 25 segundos de ejecución total fijados, solo una pequeña proporción de estos corresponde a la ejecución en el computador cuántico, concretamente, 0.556 segundos.

Si comparamos las diferencias de tiempos entre las soluciones para el grado de 6 nodos y para el grafo de 19, vemos como la diferencia real de tiempo se refiere principalmente al postprocesado realizado por el ordenador clásico donde se ejecuta la petición. Los tiempos referidos al procesamiento en el ordenador cuántico no presentan diferencia significativa entre las ejecuciones, lo que indica que el tiempo de ejecución no escala exponencialmente junto con el tamaño del problema, lo cual corrobora una de las ventajas principales de la computación cuántica frente a la clásica que es la eficiencia de la misma cuando se aplica a problemas complejos. Si comparamos los tiempos de ejecución en un ordenador normal, con algunos de los algoritmos específicos para este problema veremos como para casos sencillos el ordenador clásico puede llegar a ser ligeramente más rápido, pero según aumenta el tamaño del problema el tiempo invertido aumenta mucho más que en un ordenador cuántico. Como ejemplo de estas ejecuciones lo visto en [7], donde para el mismo grafo de 6 nodos obtienen unos tiempos medios de 0.07 segundos, para un grafo de 19 nodos 0,2 segundos y finalmente para un grafo de 52 nodos llega hasta casi los 5 segundos. Si bien es cierto que en el caso del grafo de 19 nodos este es más complejo que el va-

lorado en este trabajo, y que no se ha trabajado con un grafo equivalente al de 52 nodos como que se ha visto en el artículo, estos resultados resaltan la principal debilidad de la computación clásica en comparación a la cuántica: la forma exponencial que tiene de escalar la eficiencia en la obtención de soluciones cuando aumenta el tamaño o complejidad del problema.

Además de los tiempos, es importante destacar también que en el mismo artículo citado anteriormente, no solo se puede observar como los tiempos de obtención de las soluciones crecen de manera exponencial, sino que además dichas soluciones son ligeramente inconsistentes, existiendo una ligera desviación que no asegura obtener siempre la solución óptima global. Por contrario cuando se aplica la computación cuántica, y siempre que todos los parámetros que puedan afectar al resultado hayan sido especificados correctamente, la solución obtenida siempre deberá ser la misma y además será la mejor solución posible.

Los tiempos de ejecución vistos anteriormente son fijados para las instancias del problema en cuestión, y son lo suficientemente amplios como para que se obtenga la solución óptima. Para un caso con menos bits a tener en cuenta en la solución, ya sea por un menor número de agentes, funciones, recursos o una combinación de todo lo anterior, se puede llegar a soluciones óptimas en tiempos mucho menores de lo establecidos, para lo cual se debería hacer un cálculo sobre el tiempo estimado antes de solicitar las ejecuciones o crear los solvers.

9. Conclusión

En este trabajo se ha investigado la aplicación de la computación cuántica para la optimización de redes, demostrando su potencial para superar las limitaciones de los métodos clásicos. Concretamente, aquellos problemas cuya resolución tiene una complejidad que crece exponencialmente junto con el tamaño del problema, como son los problemas de optimización -que ha sido el caso de este trabajo-, o las simulaciones de partículas son los que suelen presentar los mayores inconvenientes a la hora de ser resueltos mediante métodos tradicionales. Los métodos clásicos no solo son muy costosos en cuanto a tiempo de ejecución y capacidad de cómputo, sino que tienden a obtener soluciones que si bien pueden ser aceptables, no son las mejores soluciones posibles. Estos son los llamados mínimos locales, soluciones que son las mejores dentro de un espacio de soluciones reducido que no representa la totalidad de las mismas. Mediante la aplicación de la computación cuántica se consigue escapar de estos mínimos locales y llegar a las soluciones verdaderamente óptimas. Además si se realiza un correcto ajuste tanto de las expresiones matemáticas como de los solvers empleados (y por supuesto del hardware sobre el que se ejecuta), las soluciones obtenidas no solo serán mejores en cuanto a costo, sino que también serán más rápidas. Utilizando la formulación QUBO y el solver proporcionado por D-Wave, se ha logrado resolver el problema de enrutamiento y asignación de recursos de manera eficiente, y se ha visto como se podría escalar a instancias del problema más complejas y realistas.

En cuanto a la consecución de los objetivos:

- Adquisición de conocimientos teóricos: Siendo estos la base indispensable para poder trabajar con las herramientas vistas, y el primer paso para poder modelar el problema, expresarlo e implementarlo, estos conocimientos han sido adquiridos de forma plena permitiendo así el desarrollo del trabajo con éxito.
- Implementación y ejecución: Tras la adquisición de una base teórica que permita modelar el problema, se tenía como objetivo la implementación de todo lo expresado de forma matemática en un código que sea ejecutable, que permita valorar la calidad del modelado y realizar los ajustes necesarios sobre la formulación y el simulador o solver empleado. Después de los diversos experimentos exitosos mostrados y las soluciones obtenidas se da por conseguido este objetivo.

- Contribución al campo de computación cuántica aplicada: Este trabajo de investigación ha requerido de un largo proceso de aprendizaje y revisión del material ya publicado hasta la fecha, usando esto como base, se ha aplicado el conocimiento adquirido a un problema cuyas soluciones y procedimientos desarrollados pueden tener aplicación en la práctica, por ejemplo, en el mundo empresarial para la optimización de rutas o carteras de inversión. Se considera por tanto que el trabajo aquí expuesto supone una positiva contribución académica al campo de la computación cuántica aplicada.
- Objetivo general: El objetivo principal de este trabajo era aprender a resolver un problema de optimización de ruta mediante metaheurísticas cuánticas. Habiendo adquirido los conocimientos teóricos necesarios, resuelto varias instancias del problema y aprendido a trabajar y configurar tanto el simulador como el solver de D-Wave, se considera un éxito completo en la consecución de dicho objetivo.

Los resultados muestran que esta alternativa no solo puede encontrar soluciones óptimas en poco tiempo, sino que también ofrece una nueva perspectiva para enfrentar problemas difícilmente resolubles mediante técnicas tradicionales. El avance que supone no solo es relevante en el ámbito académico sino que demuestra también las ventajas que puede conllevar la integración de esta tecnología para resolver los problemas de optimización que se pueden encontrar en el mundo de la empresa.

A nivel académico, este trabajo contribuye al creciente campo de la computación cuántica aplicada a problemas prácticos, proporcionando una base para futuras investigaciones. Se ha conseguido exponer una posible resolución de una versión más compleja de los problemas resueltos hasta el momento en publicaciones relacionadas, sirviendo como inspiración para futuras investigaciones y contribuciones.

Es fundamental seguir investigando y contribuyendo a este innovador campo con el objetivo de poder resolver problemas aún más complejos y desarrollar hardware cuántico accesible. Hay que destacar también las ventajas que ofrece el enfoque híbrido empleado en este trabajo, que promete facilitar el acceso a esta tecnología combinando técnicas cuánticas y clásicas para obtener soluciones.

Por esto, la computación cuántica no solo representa uno de los límites del conocimiento humano, sino que además está destinada a convertirse en un pilar esencial en diversos ámbitos y disciplinas, donde destaca el potencial revolucionario que tiene en cuanto a la forma que tenemos de abordar los problemas más complejos.

9.1. Trabajo Futuro

Habiendo demostrado la eficacia y eficiencia de esta tecnología en la resolución de problemas de optimización, aún quedan cosas por mejorar que supondrían un importante incremento en la calidad de los resultados obtenidos:

- Cálculo de constantes: Las constantes asociadas a cada restricción definida así como la función de coste pueden ser calculadas de una manera mucho más precisa mediante técnicas avanzadas como machine learning. Esto resultaría en un input de mayor calidad para los solvers empleados, y una tasa de obtención de soluciones óptimas más alta.
- Postprocesados: Después de obtener una solución, es recomendable realizar un procesado de la misma en busca de posibles errores o mejoras con las que modificar el input y/o los parámetros del solver establecidos.
- División del problema: Si se quiere aprovechar al máximo las unidades de procesamiento cuánticas en lugar de usar las híbridas, es posible dividir el problema en varios de tamaño más reducido, de forma que se obtengan soluciones independientes proporcionadas por estas unidades y que después sean reconstruidas para alcanzar la solución general. Esta es la única manera de no agotar el tiempo de uso máximo por usuario de los procesadores cuánticos, a no ser que se disponga de una cuenta con permisos extendidos, que suelen estar reservadas para empresas.

Referencias

[1] Rajak, A., Suzuki, S., Dutta, A. and Chakrabarti, B. K. (2023). Quantum annealing: An overview. Philosophical Transactions of the Royal Society A, 381(2241), 20210417.

[2] Yarkoni, S., Raponi, E., Bäck, T. and Schmitt, S. (2022). Quantum annealing for industry applications: Introduction and review. Reports on Progress in Physics, 85(10), 104001.

[3] Glover, F., Kochenberger, G., Hennig, R. and Du, Y. (2022). Quantum bridge analytics I: a tutorial on formulating and using QUBO models. Annals of Operations Research, 314(1), 141-183.

[4] Krauss, T. and McCollum, J. (2020). Solving the network shortest path problem on a quantum annealer. IEEE Transactions on Quantum Engineering, 1, 1-12.

[5] Kim, J. and Kim, S. K. (2019). Genetic algorithms for solving shortest path problem in maze-type network with precedence constraints. Wireless Personal Communications, 105, 427-442.

[6] Alkazzi, J. M. and Okumura, K. (2024). A Comprehensive Review on Leveraging Machine Learning for Multi-Agent Path Finding. IEEE Access.

[7] García, A. M. and Moreno, S. (2021). Optimización Adaptativa basada en Colonias de Hormigas para la Composición de Cadenas de Funciones Virtuales en una Red 5G Dinámica. In JITEL 2021 Libro de Actas: XV Jornadas de Ingeniería Telemática, A Coruña 2021 (pp. 229-236). Universidad de La Coruña.

[8] Moreno, S., Mora, A. M., Padilla, P., Carmona-Murillo, J. and Castillo, P. A. (2019, October). Applying ant colony optimization for service function chaining in a 5g network. In 2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS) (pp. 567-574). IEEE.

[9] Eramo, V., Miucci, E., Ammar, M. and Lavacca, F. G. (2017). An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures. IEEE/ACM Transactions on Networking, 25(4), 2008-2025.

[10] Allybokus, Z., Perrot, N., Leguay, J., Maggi, L. and Gourdin, E. (2018). Virtual function placement for service chaining with partial orders and anti-affinity rules. Networks, 71(2), 97-106.

[11] D-Wave System Documentation. `https://docs.dwavesys.com/docs/latest/index.html`.

[12] The shortest path problem using QUBO. `https://perceval.quandela.net/docs/notebooks/QUBO.html`.

[13] 'Practices with Adiabatic Quantum Computing', notas de clase, Universidad Politécnica de Madrid, 2023.