Algoritmo cuántico para resolver el problema de la mochila binaria en instancias de baja dimensionalidad

Danilo López-Sandoval, Carlos-Alberto Cobos-Lozada

dlopezs@unicauca.edu.co; [ccobos@unicauca.edu.co](mailto:ccobos@unicauca.edu.co)

Grupo de I+D en Tecnologías de la Información (GTI), Universidad del Cauca, Calle 5 # 4-70, 190003, Popayán, Colombia.

**Resumen:** Día a día los problemas que enfrenta la humanidad son más complejos, y las herramientas que nos brinda la computación clásica empiezan a ser insuficientes para resolverlos, como por ejemplo los problemas de optimización donde el espacio de búsqueda crece exponencialmente. En este contexto, la computación cuántica ha empezado a mostrar su dominio y en los últimos años empresas como IBM empiezan a ofertar servicios de cómputo cuántico para tareas de optimización y análisis de datos. Este artículo presenta y detalla los pasos seguidos para construir una solución al problema de la mochila binaria utilizando un algoritmo de Computación Cuántica Adiabática haciendo uso de las librerías de Qiskit y su implementación en Python. Para validar la eficiencia del algoritmo propuesto, se realizan experimentos con varias instancias y se comparan los resultados con unos algoritmos seleccionados para la evaluación

Las etapas de desarrollo y pruebas (experimentación) se realizaron en paralelo, en un proceso de retroalimentación constante. Implementándose una versión básica de un algoritmo que permitiera hacer la integración de los tres programas ejecutables un par de las técnicas seleccionadas, que eran inmediatamente probadas en experimentos rápidos en ambientes pequeños, cuyos resultados fueron usados para guiar el desarrollo de pequeñas variaciones sobre las técnicas. Estas variaciones incluyeron aspectos como: variación en valores de los parámetros, inclusión de factores adicionales en las ecuaciones, cambios en el orden de ejecución de los pasos, fusión de componentes de múltiples algoritmos, entre otros.

**Palabras-clave:** Computación cuántica; Problema de la mochila; Computación cuántica adiabática; Modelo de Ising; Problemas de optimización

**Abstract:**

**Keywords:** Quantum computing; Backpack problem; Adiabatic quantum computing; Ising's model; Optimization problems

# INTRODUCCION

El problema de la mochila es un conocido problema de optimización combinatoria que pertenece a la clase de problemas NP-completos (Karp, 2010). Existen diferentes variantes de este problema que se utilizan como guía para resolver una gran variedad de problemas, entre ellos, problemas de empaque y reducción de existencias, toma de decisiones financieras, titulización respaldada por activos, subastas combinatorias (Coffey, 2017), control de presupuestos, toma de decisiones y corte de material (H. Wang, Ma, Zhang, & Li, 2009).

Entre las variantes más conocidas del problema de la mochila se encuentran: la mochila binaria (Binary Knapsack Problem, BKP), el problema de la suma de subconjuntos (Pi = Wi), el problema de la mochila sin límite, el problema de la mochila acotada (Martello & Toth, 1987), el problema de la mochila d-dimensional (d-KP), el problema de las múltiples mochilas (MKP) (Gurski, Rehs, & Rethmann, 2019), el problema de la mochila de elección múltiple multidimensional y el problema de la mochila cuadrática (Assi & Haraty, 2019)

A la fecha se han desarrollado una variedad de técnicas para resolver las diferentes variantes del problema de la mochila, las cuales se pueden agrupar de la siguientes manera: (i) exactos, (ii) programación dinámica, (iii) programación entera, (iv) métodos metaheurísticos, (v) métodos lagrangianos, (vi) métodos basados en árboles de búsqueda con back tracking y (vii) enfoques de red (Salkin & Kluyver, 1975).

Esta investigación se relaciona con el problema de la mochila binaria que se define formalmente por la **Ecuación (1)** y se describe de la siguiente forma: dada una mochila con una capacidad limitada **C** ∈ Z+, y un conjunto de ***n*** elementos (artículos o ítems), cada uno con un beneficio **Pi** ∈ Ζ+ y un peso **Wi** ∈ Ζ+ donde i = 1, 2, ..., n, se debe seleccionar un subconjunto de ***m*** elementos () de modo que se genere la mayor (máxima) ganancia o beneficio posible, sujeto a una restricción principal la cual define que los pesos totales de los elementos seleccionados no excedan la capacidad **C** de la mochila (Wang et al., 2009), teniendo en cuenta que es un valor binario {0,1} que indica si el elemento *i* debe ir o no en la mochila.

|  |  |
| --- | --- |
|  | (1) |

Existen diferentes maneras de clasificar un problema de mochila binaria, entre las más destacadas se tiene una definición basada es el número de elementos n (baja o pequeña dimensionalidad si n < 20, dimensionalidad media si 20 ≤ n < 200 y alta dimensionalidad si n ≥ 1000 (Blado & Toriello, 2019). y una definición mejor aceptada de clasificación de acuerdo con la complejidad de las instancias, esto es la relación o no de los pesos de cada ítem con su valor, y se definen como instancias no correlacionadas, débilmente correlacionadas, casi fuertemente correlacionadas, fuertemente correlacionadas, Inversamente correlacionadas, e instancias con pesos y beneficios iguales (subconjuntos de instancias de suma) (Pisinger, 2005).

Debido a la importancia y el reto que representa el problema de la mochila binaria, en los últimos años se han reportado un gran número de algoritmos que buscan su solución, estos se agrupan en: exactos, de programación dinámica, basados en back-tracking (incluidos ramificación y poda), y metaheurísticos. Entre los algoritmos metaheurísticos más destacados se encuentran los algoritmos genéticos, el recocido simulado, la optimización por enjambre de partículas (PSO) y la búsqueda tabú (Vickram, Krishna, & Srinivas, 2016; Wang et al., 2009). Además, recientemente se trata con algoritmos cuánticos como el algoritmo evolucionario cuántico (Li & Li, 2019), el algoritmo genético cuántico (Wang, Guo, Xiang, & Mao, 2012) y el algoritmo cuántico Variational Quantum Eigensolver (VQE) (Bolos, 2019).

La computación cuántica nace como una alternativa al paradigma computacional clásico basado en máquinas de Turing y de Von Neumann, la cual ha demostrado su superioridad ante la computación clásica para algunos problemas específicos (Albash & Lidar, 2018; Vogel, 2011). Con este nuevo paradigma se pueden estudiar problemas de alta complejidad que requieren gran cantidad de operaciones y manejan gran cantidad de variables, para esto, se hace uso de algunas propiedades de la física cuántica como el entrelazamiento cuántico o la superposición cuántica, con las cuales se pueden realizar más operaciones en una misma unidad de tiempo disminuyendo radicalmente los tiempos de respuesta (Vega Fernández & Ramírez Celis, 2017).

A la fecha se han definido varios algoritmos cuánticos eficientes para problemas discretos como la factorización entera, la simulación cuántica, la estimación del valor propio, la integración, la solución de ecuaciones diferenciales parciales y la solución a problemas numéricos de álgebra lineal (Hadfield, 2018), pero una búsqueda en la literatura reveló que existen muy pocos artículos publicados sobre algoritmos cuánticos que se apliquen al problema de la mochila binaria (Coffey, 2017; Li & Li,2019).

Para lograr la solución de problemas combinatorios complejos como el de la mochila binaria, se hace necesario entender y comprender los conceptos claves de la computación cuántica, los modelos hamiltonianos, la computación cuántica adiabática y el desarrollo actual de soluciones a problemas concretos de optimización. Finalmente, realizar nuevas propuestas y comparar las soluciones obtenidas con los modelos cuánticos y sus contrapartes tradicionales para establecer si se obtienen mejoras. Iniciando este trabajo, el autor de la presente propuesta ha realizado una primera revisión e implementación de un algoritmo cuántico para la solución del problema de la mochila binaria, encontrando que lo existente no permite la solución del 100% de las instancias evaluadas, ya que los modelos de Ising existentes tienden a llenar la mochila totalmente y en algunas instancias la solución óptima se encuentra dejando la mochila con espacio libre (López, & Cobos, 2020).

En este sentido, en este documento se plantea la siguiente pregunta de investigación: ¿Cuáles son las características de un hamiltoniano cuántico basado en un modelo de Ising y su implementación, que al ser ejecutada en emuladores de computación cuántica permite obtener resultados comparables o mejores que los obtenidos mediante algoritmos clásicos del problema de la mochila binaria en instancias de baja dimensionalidad (n < 20)?

Es preciso aclarar dos aspectos de esta pregunta de investigación. Primero, se planeta el uso de emuladores de computación cuántica debido a que a la fecha no se cuenta con el acceso a computadores cuánticos que se programen con lenguajes de alto nivel como Python; existe la posibilidad de contar con tiempo limitado de procesamiento en un computador cuántico de IBM, pero programado a nivel de circuitos y compuertas cuánticas, lo que desborda el alcance y el objetivo central de la investigación. Y segundo, se planeta el uso de instancias de baja dimensionalidad (n < 20), esto debido a que los emuladores tienen restricciones para poder procesar problemas con mayor número de dimensiones.

# Conceptos Cuánticos aplicados al problema de la mochila

## Computación cuántica

La computación cuántica es un nuevo paradigma de computación que surgió como resultado de la fusión de la informática y la mecánica cuántica. El origen de la computación cuántica se remonta a principios de los 80’s cuando Richard Feynman observó que algunos efectos de la mecánica cuántica no se pueden simular de manera eficiente en una computadora clásica (Vega, & Ramírez, 2017).

En computación cuántica, un bit cuántico (Quantum bit, qubit) es la unidad de información más pequeña almacenada en una computadora cuántica de dos estados (Hey, 1999). Al contrario del bit clásico el cual tiene dos valores posibles, “0” o “1”, un qubit puede estar en el estado "1", en el estado "0" o en cualquier superposición de los dos estados. El estado de un qubit se puede representar mediante la notación de corchetes (Bra-Ket) presentada en la Ecuación (2).

|  |  |
| --- | --- |
|  | (2) |

Donde denota más de un vector en algún espacio vectorial y representa una superposición lineal de la partícula dados los vectores de estado cuántico individuales. representan los valores de bit clásicos 0 y 1 respectivamente. y son números complejos tales que , los cuales especifican las amplitudes de probabilidad de los estados correspondientes. Cuando se realiza la medición del estado de un qubit, se puede obtener cero con una probabilidad o uno con una probabilidad . Existe una particularidad en el ámbito cuántico, al observar el estado cuántico de un qubit, este colapsa a un solo estado, cero (0) o uno (1) (Narayanan, 1999).

Un sistema de n-qubits puede representar estados al mismo tiempo. Este crecimiento exponencial del espacio de estado con el número de qubits es lo que sugiere una aceleración exponencial de la computación en las computadoras cuánticas sobre las computadoras tradicionales.

Con este nuevo paradigma se pueden estudiar problemas de alta complejidad que tienen gran cantidad de operaciones y manejan gran cantidad de variables. Para esto, se hace uso de algunas propiedades de la física cuántica como el entrelazamiento cuántico o la superposición cuántica, con las cuales se pueden realizar más operaciones en una misma unidad de tiempo disminuyendo radicalmente los tiempos de respuesta (Vega, & Ramírez, 2017). Entre los algoritmos cuánticos más famosos se encuentran el algoritmo de Shore’s (Shor, 1994) utilizado para factorización numérica y el algoritmo de Grover’s (Grover, 1996) utilizado para búsquedas en una base de datos no ordenada. Ambos algoritmos redujeron la complejidad de la solución al problema (Laboudi & Chikhi, 2012). al igual que los algoritmos de cuánticos aplicados a la estimación del valor propio, la integración, la solución de ecuaciones diferenciales parciales y la solución a problemas numéricos de álgebra lineal (Hadfield, 2018).

Debido a la complejidad de diseñar y probar algoritmos cuánticos complejos en una maquina real, algunos investigadores han optado por emular algunas propiedades de la computación cuántica en algoritmos tradicionales (Layeb, 2011).

## Computación cuántica adiabática (AQC)

La computación cuántica adiabática (Adiabatic Quantum Computation, AQC) es un enfoque equivalente al modelo de circuito de computación cuántica, adecuado para problemas del tipo de optimización combinatoria, incluyendo particiones, coberturas, particionado de árboles y gráficos, y satisfacción booleana (Coffey, 2017).

Debido a sus inicios, la AQC puede considerarse como una clase particular de Recocido cuántico (Quantum Annealing, QA), la cual utiliza los principios de la mecánica cuántica para resolver problemas de optimización (Albash, & Lidar, 2018). Desde la propuesta inicial de QA, ha habido mucho interés en la búsqueda de problemas prácticos donde pueda ser ventajoso con respecto a los algoritmos clásicos, particularmente el recocido simulado (Simulating annealing, SA). Muchos de estos enfoques transforman un problema computacional en un problema donde se debe encontrar el estado fundamental de un modelo Ising Spin Glass (ISG) cuántico, el cual, en el peor de los casos es un problema NP-completo (Cao et al., 2016).

## Hamiltoniano en computación cuántica adiabática

En un modelo de circuito de computación cuántica, un cálculo puede evolucionar en todo el espacio de Hilbert (es una generalización del espacio euclidiano, es un espacio de producto interior que es completo con respecto a la norma vectorial definida por el producto interior) y está codificado en una serie de puertas de lógica cuántica unitarias (Albash, & Lidar, 2018). Por otro lado, en un modelo AQC el cálculo se realiza mediante un Hamiltoniano inicial (), cuyo estado fundamental codifica la solución a un problema de interés, y otro hamiltoniano (), cuyo estado fundamental es trivial. Entonces, si se prepara un sistema cuántico para estar en el estado fundamental , y luego se cambia adiabáticamente el hamiltoniano por un tiempo T de acuerdo con la Ecuación (3) y T es lo suficientemente grande, al final, el estado cuántico en T devolverá una solución al problema de interés (Albash, & Lidar, 2018; Cao et al., 2016; Lucas, 2014).

|  |  |
| --- | --- |
|  | (3) |

Donde s ∈ [0, 1], es un parámetro de tiempo considerado como una función dependiente del tiempo s(t) = para un tiempo total de evolución T (en general se puede considerar cualquier función que satisfaga s(0) = 0 y s(1) = 1), y y no conmutan (Dos operadores no conmutan cuando se cumple que [A, B] = AB - BA 0 (Benenti, Casati, & Strini, 2007)). Con esta definición y debido al teorema adiabático de la mecánica cuántica el sistema cuántico permanecerá en el estado fundamental todo el tiempo.

## Modelo Ising

Uno de los modelos más utilizados en física se llama el modelo Ising. Propuesto entre 1920 y 1930 por Ernst Ising y Wilhelm Lenz como una forma de entender el funcionamiento de los materiales magnéticos. El modelo Ising (Una clase conveniente, restringida y ciertamente no universal de Hamiltoniano) tiene la versatilidad de codificar eficientemente muchos problemas NP y ha motivado la realización física de QA.

El enfoque modela un material magnético como una colección de moléculas, cada una de las cuales tiene un espín que puede alinearse o anti-alinearse con un campo magnético aplicado , y que interactúan entre sí con base en un campo de interacción (Brush, 1967).

En la Ecuación (4) se representa el modelo clásico de Ising, el cual se puede escribir como una función cuadrática de un conjunto de n giros, donde si ∈ {-1, +1} representa el spin de la i-ésima partícula.

|  |  |
| --- | --- |
|  | (4) |

La Ecuación (5) representa un Hamiltoniano () como la versión cuántica del modelo Ising, donde se reemplaza por en la Ecuación (4).

|  |  |
| --- | --- |
|  | (5) |

es una matriz de Pauli que actúa sobre el i-ésimo spin en un espacio de Hilbert de N qubits donde 𝕀 es una matriz identidad de 2 x 2 y , ∈ ℝ son coeficientes (Lucas, 2014; Cao et al., 2016). es la fuerza del campo aplicado sobre el i-ésimo spin y actúa como el campo de interacción entre los spines vecinos i, j (Bian et al.,  2010). Un estado fundamental es una superposición de todos los estados posibles en la base propia de (Lucas, 2014).donde es la puerta NOT sobre el i-ésimo qubit (Coffey, 2017).

## Como Abordar un Problema de Optimización Mediante el Mapeo de una Función de Costo a un Hamiltoniano Cuántico

Para dar solución al problema de la mochila binaria mediante el modelo de computación cuántica, en (Lucas, 2014) se realiza la formulación de Ising de los 21 problemas NP-completos de Karp. Entre los problemas resueltos se tiene una primera aproximación de la formulación de un Hamiltoniano Ising para el problema de la mochila binaria con una ecuación que consta de n + C qubits la cual permite representar y orientar la búsqueda de una solución. Dado que la cantidad de qubits en los sistemas cuánticos actuales tiende a ser un factor de importancia, se hace necesario que en los algoritmos se logre reducir al máximo los qubits requeridos, por ello en (Coffey, 2017).se realiza la reducción del vector solución de una cantidad N de qubits a N.

Los dos modelos de solución se detallan a continuación ya que podrían servir como base para lograr resolver problemas de optimización similares.

## Solución con (n + C) qubits

Teniendo en cuenta la definición del problema de la mochila propuesta en la **Ecuación (1)** y trabajando con la versión binaria de la misma, se tienen *n* variables donde *1 ≤ i ≤ n*, las cuales denotan valores binarios que se fijan con uno (1) si un determinado elemento debe ir en la mochila o cero (0) en caso contrario, y *j* variables donde *1 ≤ j ≤ C* que denotan una variable binaria la cual es 1 si el peso de la mochila es j, o cero (0) en caso contrario; Así, la cantidad de qubits requerida para modelar la solución resulta de la suma del número de artículos y la capacidad máxima de la mochila: (*n* + C) → [].

La solución al problema consiste en construir un modelo de Ising H = + , con como el término que controla el cumplimiento o no de la restricción del problema, cómo se define en la **Ecuación (6).**

|  |  |
| --- | --- |
|  | (6) |

Con se obliga a que el peso solo pueda tomar un valor y que el peso de los objetos en la mochila sea igual al valor que se desea. Por ejemplo, si una solución da un valor igual a 10 para la variable peso, solo , el resto de para los otros valores de k. Si la capacidad de la mochila (C) fuese de 20, esta solución es factible y en ese caso sería igual a , eliminando así a . Pero si C tuviera u valor igual a 8, esta solución no sería factible y en este caso sería igual a haciendo que el valor de sea grande (positivo, alejando la solución del mínimo), y entre más se supere el peso máximo de la mochila, mayor será el valor de . Finalmente se tiene en la **Ecuación (7)**.

|  |  |
| --- | --- |
|  | (7) |

El término sirve para maximizar el beneficio (a mayor beneficio más se acerca al optimo o mínimo del Hamiltoniano) y el término asegura que se cumpla la restricción de peso total. Debido a que se requiere que no sea posible encontrar una solución donde no se cumpla a expensas de que se vuelva más negativo, se requiere fijar la condición: 0 < B \* Max () < A (Lucas, 2014).

## Solución con (n + ⌊log2 C⌋ + 1) qubits

Realizando una modificación a la solución planteada anteriormente, la cual requiere (*n* + C) qubits, se desea reducir drásticamente la cantidad de giros adicionales que deben agregarse. Así, codificando la magnitud del peso máximo (C) como un valor binario, se define una variable entera M tal que , con M = ⌊⌋; De esta manera solo se necesitan (M+1) variables binarias: ... ], en lugar de C variables binarias: [ ... ], para codificar una variable que puede tomar C valores (C >> M).

Aplicando la nueva configuración, para resolver el problema de la mochila, se requieren (*n*+⌊⌋+1) qubits (Lucas, 2014). En la **Ecuación (8)** se presenta construido de acuerdo a la nueva representación de (Coffey, 2017).

|  |  |
| --- | --- |
|  | (8) |

Si se tiene una solución con un peso de 10 y una capacidad máxima *C* = 20, entonces la cantidad de qubits requeridos para representar el peso de la solución es M=⌊⌋+1=5, por lo tanto, se tiene el vector = 10, que corresponde a la representación de 10 en binario de izquierda a derecha; Así, se cumple la condición 16 32. Esta solución es factible, por lo que , quedando eliminado . Por otro lado, si *C* = 9, se tendría el vector con , que representa un valor alto de contrapeso ya que se supera la capacidad máxima permitida. Este valor crecerá entre más se rebase la capacidad de la mochila.

Este segundo hamiltoniano es el que se toma como referente en el presente trabajo como solución del problema de la mochila binaria. En resumen, el Hamiltoniano H = + esta dado por la **Ecuación (9)**, que utiliza una representación en base cero (o) para los vectores.

|  |  |
| --- | --- |
|  | (9) |

Los diferentes valores implicados se representan de la siguiente manera (Bolos, 2019):

La solución se representa como un vector = [ ... ] ∈ {0, 1} en donde cada elemento se representa en un índice del vector. Si el valor es 1, entonces el elemento se almacena en la mochila y si es cero (o) el elemento queda fuera.

* El peso de la solución se representa en binario de izquierda a derecha como un vector = [ ... ] ∈ {0, 1}.
* El valor del peso de los elementos se representa como un vector W= [].

El beneficio de los elementos se representa como un vector P = [].

* La variable C representa la capacidad de la mochila.
* La variable A representa un número lo suficientemente grande para dominar la multiplicación del valor B por la suma de los pesos de todos los elementos.

Para construir una función de costo C() se debe tener en cuenta que se desea maximizar la **Ecuación (1)** mientras se cumple con su respectiva restricción (Bolos, 2019). De la definición de H, se tienen M+1 nuevas variables binarias que se representarán en el vector y = [… ], donde los elementos pueden tomar valores diferentes de cero simultáneamente (Coffey, 2017). La solución en la función de costo se representará en un vector K compuesto por los vectores y , con = [… … ]. En la **Ecuación (10)** se presenta la función de costo C(K) definida por (Bolos, 2019) y su polinomio expandido.

Se puede observar que para llevar el primer término de la **Ecuación (10)** a cero, se debe cumplir que y se tenga un S = . El valor de L debe dominar la suma de los valores de P, esto es, . Si por el contrario , el primer término nunca será cero que multiplicado por L evitará seleccionar soluciones no factibles. El valor mínimo de la función de costo será uno donde la restricción es respetada y la suma de los valores se maximice (Bolos, 2019).

|  |  |
| --- | --- |
|  | (10) |
|  |

## Mapeo de la función de costo a su hamiltoniano cuántico

Al realizar el mapeo de las variables por (𝕀 - )/2, por (𝕀 - )/2 y eliminar el primer término que es constante, se obtiene la Ecuación (11), donde 𝕀 es una matriz identidad de orden *n+m* y es la matriz resultante de tensorizar *n+m* matrices donde en cada posición diferente de *i* los factores del tensor son matrices identidad y la posición *i* es una matriz de Pauli Z.

La matriz de Pauli se compone de dos vectores, un vector X y un vector Z, para este caso se modifica el vector Z únicamente. Para hallar los coeficientes de la matriz se debe resolver cada sumatoria resultante en la **Ecuación (11)**, encontrar el respectivo índice para cada valor de la sumatoria e ir adicionándolo a una lista. Por ejemplo, para el termino:

, se muestra su algoritmo en la Figura 1.

|  |  |
| --- | --- |
|  | (11) |

Hay que tener en cuenta que no es necesario calcular los tensores por cuenta propia, ya que Qiskit tiene incorporado un objeto que toma los índices de la matriz de Pauli Z en el tensor y realiza los cálculos restantes. Como resultado se obtiene una lista de objetos de Pauli y un valor shift que se encarga se acumular la diferencia de las magnitudes de la operación de cada sumatoria. Esos valores se envían al algoritmo Variational Quantum Eigensolver (VQE) con el objetivo de minimizar la función y retornar una solución X’ donde los primeros n términos son la solución al problema de la mochila.

# Marco de Trabajo para Realizar la Evaluación y Comparación de los Algoritmos Seleccionados



## Descripción del Marco de Trabajo

En la actualidad el manejo que se da en el ámbito del desarrollo de soluciones cuánticas en problemas de optimización se enfoca en modelos de circuitos cuánticos que trabajan con programación a bajo nivel, esto quiere decir con compuertas cuánticas que alteran el estado de un qubit el cual funciona sobre una máquina que trabaja bajo las leyes de la mecánica cuántica, esta tecnología permite realizar cálculos en paralelo a través del principio de superposición de estados.

Debido a que esta tecnología aún está en una fase temprana de desarrollo, aun no se tienen lenguajes de programación específicos y se presentan algunos problemas de escalabilidad ya que con el incremento del espacio de búsqueda de un algoritmo aumenta la cantidad de qubits requeridos para realizar operaciones de cálculos complejos. Debido a esto y mientras se realizan avances significativos en materia de Hardware y Software, se plantea el uso de emuladores de computación cuántica que incorporan la lógica de la mecánica cuántica para desarrollar heurísticas que permitan dar soluciones complejas y eficientes sobre una infraestructura de computación clásica.

Con el auge de las investigaciones en el campo de computación cuántica y su superioridad frente a la computación clásica de Turing para un puñado de problemas específicos, varias empresas como Google, Microsoft e IBM se han involucrado en esta área de investigación.

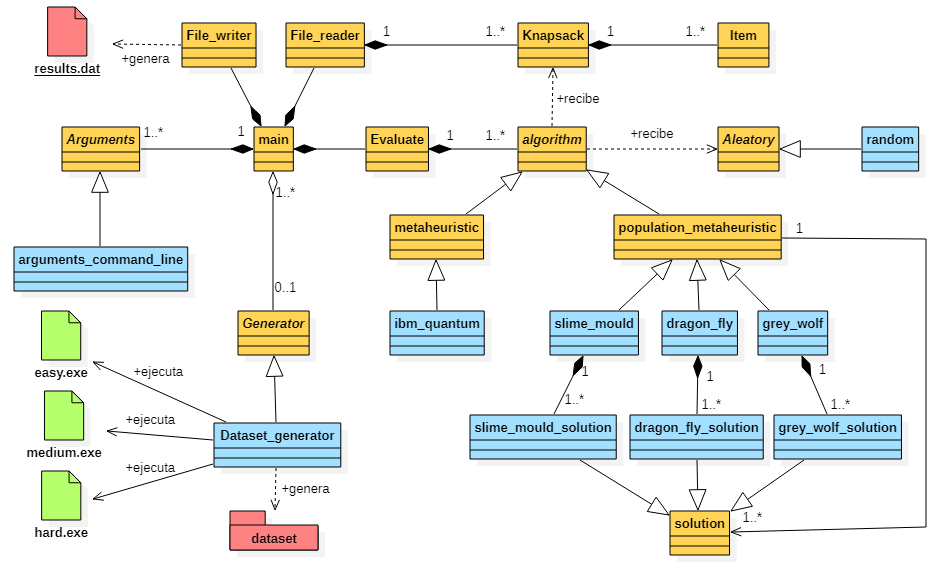
Teniendo en cuenta que el modelo de circuitos de compuertas cuánticas definido para computación cuántica aún no está en la capacidad de dar solución a algoritmos complejos y que en el presente trabajo se desea utilizar algoritmos de computación cuántica que tuvieran la capacidad de resolver problemas de optimización, se decide escoger una librería desarrollada por el equipo de IBM para el estudio de soluciones cuánticas denominada Qiskit. Esta librería contiene un conjunto de herramientas y algoritmos desarrollados en base a la experimentación que se ha venido estudiando en una de las primeras computadoras cuánticas reales de IBM, por lo que tiene una base teórica robusta, cuenta con una documentación extensa y detallada, y además cuenta con todo el apoyo de la infraestructura que IBM ha venido creando en los últimos años.

En el proceso de revisión en la literatura se puede vislumbrar un amplio abanico de algoritmos que se han venido utilizando para dar solución al problema de la mochila, donde una de las preocupaciones es tratar de encontrar un equilibrio entre exploración y explotación del espacio de búsqueda y lograr reducir el tiempo de respuesta utilizado como métrica de comparación de eficiencia y eficacia. Entre los algoritmos del estado del arte propuestos y la escogencia de los algoritmos que son objeto de evaluación de los mismos, se logra apreciar un uso recurrente de algoritmos que funcionan bajo metaheurísticas poblacionales y se plantea la necesidad de realizar estudios comparativos con instancias de prueba que tengan diversidad en cuanto a la correlación de elementos, dimensionalidad y tamaños de muestra. Esto nos da un camino de estudio y algunos criterios para la escogencia de los algoritmos que serán objeto de comparación en el presente trabajo.

En la **Figura 1** se muestra un diagrama de clases de alto nivel del algoritmo implementado. Las clases de color naranja corresponden al núcleo del marco de trabajo, y las clases de color azul son clases derivadas asociadas a implementaciones de algoritmos específico o implementaciones de clases abstractas, los artefactos en color verde corresponden a los archivos ejecutables que contienen los algoritmos generadores de los conjuntos de datos, por último, los artefactos de color rojo hace referencia a documentos de respuesta con los resultados obtenidos de la evaluación de los algoritmos y a los archivos resultantes de la generación del conjunto de datos.

El marco de trabajo de divide en cinco módulos:

* **Modulo principal (main):** modulo principal que controla el funcionamiento del algoritmo, definiendo la lógica requerida donde se permite crear y realizar pruebas sobre múltiples arquitecturas, algoritmos y conjuntos de datos.
* **Modulo generador:** se encarga de generar el conjunto de datos de acuerdo a los parámetros ingresados por línea de comandos.
* **Módulo de archivos:** se centra en realizar el cargue del conjunto de datos con los cuales se realiza la comparación de rendimiento de los algoritmos y se encarga de escribir los resultados en un documento de texto.
* **Módulo de algoritmos:** define la estructura base para lograr integrar nuevos esquemas de algoritmos.



1. Diagrama de clases de alto nivel

A continuación, se explica brevemente cada uno de los módulos mencionados.

### Modulo principal (main)

Este módulo se puede subdividir en dos partes, por un lado, tenemos las clases encargadas de la gestión argumentos ingresados por línea de comandos, y por el otro lado tenemos la clase principal y de evaluación encargadas de gestionar el funcionamiento del marco de trabajo.

### Modulo generador

Este módulo se encarga de generar el conjunto de datos pertinentes de acuerdo a la configuración ingresada por el usuario, en ella se puede observar que se relacionan tres artefactos, easy.exe, médium.exe y hard.exe, estos artefactos son programas generadores de instancias de mochila con los cuales permiten crear los conjuntos de datos necesarios.

#### Conjunto de datos

para la realización de las pruebas es necesario contar con un conjunto de datos extenso y diverso, por ello se decidió tomar tres generadores de instancias de mochila ampliamente reconocidos por la comunidad científica y se encuentran disponibles en [43]. Estos generadores están desarrollados en el lenguaje de programación C y con ellos se tiene la posibilidad de generar instancias con distintas configuraciones para realizar pruebas de algoritmos en condiciones más realistas. Los generadores se detallan a continuación:

* **Easy.exe**

Generador para construir instancias de prueba para el problema de la mochila binaria, este generador se describe en (Pisinger, 2002).

* **Medium.exe**

Generador avanzado para construir instancias de prueba para el problema de la mochila binaria, se consideran 14 tipos de instancias diferentes. este generador se describe en (Pisinger, 2002).

* **Hard.exe**

Generador para construir instancias de prueba complejas para el problema de la mochila binaria. este generador se describe en (Pisinger, 2005).

Para la construcción del conjunto de datos se definieron cuatro variables a tener en cuenta:

* **Tipo de correlación:**

Hace referencia al tipo de correlación que deben tener los elementos de la mochila.

* **Dificultad:**

Hace referencia a la dificultad que debe tener el conjunto de datos seleccionados, si se requiere una dificultad normal, el conjunto de datos se genera con el artefacto ***easy.exe***,si se requiere una dificultad mediael conjunto de datos se genera con el artefacto ***medium.exe*** y si se requiere una dificultad alta el conjunto de datos se genera con el artefacto ***hard.exe****.*

* **Cantidad de elementos:**

Hace referencia a la cantidad de elementos que van a estar disponibles para almacenar en cada instancia de mochila.

* **Rango de los coeficientes:**

Hace referencia al valor máximo permitido para los valores de peso de cada elemento disponible para almacenar mochila.

### Módulo de archivos

Este módulo se encarga de gestionar el acceso a los archivos necesarios en la ejecución del algoritmo. Este módulo consta de dos clases principales, una encargada de la lectura de archivos y otra encargada de la escritura de resultados en un archivo de salida.

### Módulo de algoritmos

Este módulo se encarga de brindar una arquitectura base de alto nivel para la implementación de nuevos algoritmos, la cual garantiza que se puedan adicionar nuevas metaheurísticas sin que se afecte el funcionamiento del marco de trabajo.

# Resultados

# Conclusiones

# REFERENCIAS

Albash, T., & Lidar, D. A. (2018). Adiabatic quantum computation. Reviews of Modern Physics, 90(1), 015002. https://doi.org/10.1103/RevModPhys.90.015002

Assi, M., & Haraty, R. A. (2019). A Survey of the Knapsack Problem. ACIT 2018 - 19th International Arab Conference on Information Technology, 1–6. <https://doi.org/10.1109/ACIT.2018.8672677>

Benenti, G., Casati, G., & Strini, G. (2007). Principles of quantum computation and information: Volume II: Basic tools and special topics. Principles of Quantum Computation and Information - Volume II: Basic Tools and Special Topics (Vol. I). 57 Shelton Street, Covent Garden, London WC2H 9HE: World Scientific Publishing Co. Pte. Ltd. <https://doi.org/10.1142/5838>

Bian, Z., Chudak, F., Macready, W., & Rose, G. (2010). The Ising model: teaching an old problem new tricks. D-Wave Systems, 1–32. Retrieved from <https://www.dwavesys.com/sites/default/files/weightedmaxsat_v2.pdf>

Blado, D., & Toriello, A. (2019). Relaxation Analysis for the Dynamic Knapsack Problem with Stochastic Item Sizes. SIAM Journal on Optimization, 29(1), 1–30. <https://doi.org/10.1137/16M1101209>

Bolos, S. (2019). GitHub - sorin-bolos/QiskitCampAsia2019. Retrieved March 26, 2020, from https://github.com/sorin-bolos/QiskitCampAsia2019

Brush, S. G. (1967). History of the Lenz-Ising model. Reviews of Modern Physics, 39(4), 883–893. <https://doi.org/10.1103/RevModPhys.39.883>

Cao, Y., Jiang, S., Perouli, D., & Kais, S. (2016). Solving Set Cover with Pairs Problem using Quantum Annealing. Scientific Reports, 6(1), 1–15. <https://doi.org/10.1038/srep33957>

Coffey, M. W. (2017). Adiabatic quantum computing solution of the knapsack problem, 1–22. Retrieved from <http://arxiv.org/abs/1701.05584>

Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. Proceedings of 28th Annual ACM Symposium on the Theory of Computing, 41(3), 212–221. https://doi.org/10.1007/BF01913177

Gurski, F., Rehs, C., & Rethmann, J. (2019). Knapsack problems: A parameterized point of view. Theoretical Computer Science, 775, 93–108. <https://doi.org/10.1016/j.tcs.2018.12.019>

Hadfield, S. A. (2018). Quantum Algorithms for Scientific Computing and Approximate Optimization, 1–264. <https://doi.org/10.7916/D8X650C9>

Hey, T. (1999). Quantum computing: an introduction. Computing Control Engineering, 10(3), 105–112. <https://doi.org/10.1049/cce:19990303>

Karp, R. (2010). Reducibility Among Combinatorial Problems. In 50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art (pp. 219–241). <https://doi.org/10.1007/978-3-540-68279-0>

Laboudi, Z., & Chikhi, S. (2012). Comparison of genetic algorithm and quantum genetic algorithm. International Arab Journal of Information Technology, 9(3), 243–249.

Layeb, A. (2011). A novel quantum inspired cuckoo search for knapsack problems. International Journal of Bio-Inspired Computation, 3(5), 297–305. <https://doi.org/10.1504/IJBIC.2011.042260>

Li, J., & Li, W. (2019). A new quantum evolutionary algorithm in 0-1 knapsack problem. Communications in Computer and Information Science, 986, 142–151. <https://doi.org/10.1007/978-981-13-6473-0_13>

López, D., & Cobos C. A., (2020)“Adiabatic Quantum Computing applied to the solution of the Binary Knapsack Problem,” Rev. Ibérica Sist. e Tecnol. Informação, vol. https//DOI: 10.17013/risti.n.pi-pf

Lucas, A. (2014). Ising formulations of many NP problems. Frontiers in Physics, 2, 1–14. https://doi.org/10.3389/fphy.2014.00005

Martello, S., & Toth, P. (1987). Algorithms for Knapsack Problems. North-Holland Mathematics Studies, 132(C), 213–257. <https://doi.org/10.1016/S0304-0208(08)73237-7>

Narayanan, A. (1999). Quantum computing for beginners. Proceedings of the 1999 Congress on Evolutionary Computation, CEC 1999, 3, 2231–2238. <https://doi.org/10.1109/CEC.1999.785552>

Pisinger, D. (2002) “Core problems in knapsack algorithms,” Operations Research, vol. 47, no. 4, pp. 570–575, doi:[10.1287/opre.47.4.570](http://dx.doi.org/10.1287/opre.47.4.570)

Pisinger, D. (2005). Where are the hard knapsack problems? Computers and Operations Research, 32(9), 2271–2284. <https://doi.org/10.1016/j.cor.2004.03.002>

Salkin, H. M., & de Kluyver, C. A. (1975). The knapsack problem: a survey\*, 22(1), 127–144. <https://doi.org/https://doi.org/10.1002/nav.3800220110>

Shor, P. W. (1994). Algorithms for Quantum Computation: Discrete Logarithms and Factoring. Proceedings 35th Annual Symposium on Foundations of Computer Science, 59(3), 124–134. https://doi.org/10.1016/0024-3205(96)00287-1

Vega Fernández, C. A., & Ramírez Celis, J. S. (2017). Computación Cuántica: Implementación De Algoritmos De Shor Y Grover En El Computador Cuántico De Ibm. Escuela colombiana de Ingenieria Julio Garavito. Retrieved from <https://bit.ly/2PJW5HR>

Vickram, P., Krishna, A. S., & Srinivas, V. S. (2016). A Survey on Design Paradigms to solve 0/1 Knapsack Problem. International Journal of Scientific & Engineering Research, 7(11), 112–117. Retrieved from <https://bit.ly/2DMYkYu>

Vogel, M. (2011). Quantum Computation and Quantum Information, by M.A. Nielsen and I.L. Chuang. Contemporary Physics (Vol. 52). https://doi.org/10.1080/00107514.2011.587535

Wang, H., Ma, L., Zhang, H., & Li, G. (2009). Quantum-inspired ant algorithm for knapsack problems. Journal of Systems Engineering and Electronics, 20(5), 1012–1016. Retrieved from https://bit.ly/3kzdSjf

Wang, R., Guo, N., Xiang, F., & Mao, J. (2012). An improved quantum genetic algorithm with mutation and its application to 0-1 knapsack problem. Intemational Conference on Measurement, Information and Control (MIC), (M Ic), 484–488. <https://doi.org/10.1109/MIC.2012.6273347>