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

****

**DANILO LÓPEZ SANDOVAL**

**Director: Ph.D. CARLOS ALBERTO COBOS LOZADA**

UNIVERSIDAD DEL CAUCA

FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

DEPARTAMENTO DE SISTEMAS

GRUPO DE I+D EN TECNOLOGÍAS DE LA INFORMACIÓN (GTI)

**LÍNEA INVESTIGACIÓN EN SISTEMAS INTELIGENTES**

POPAYÁN, MARZO DE 2022

Capítulo 1

# **INTRODUCCION**

# **PLANTEAMIENTO DEL PROBLEMA**

El problema de la mochila es un conocido problema de optimización combinatoria que pertenece a la clase de problemas NP-completos [1]. Existen diferentes variantes del problema de la mochila 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 [2], control de presupuestos, toma de decisiones y corte de material [3].

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 [4], el problema de la mochila d-dimensional (d-KP), el problema de las múltiples mochilas (MKP) [5], el problema de la mochila de elección múltiple multidimensional y el problema de la mochila cuadrática [6]. 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 [7].

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 [3], 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 [8]) 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) [9].

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 [3], [10]. Además recientemente se han realizado investigaciones con algoritmos cuánticos como el algoritmo evolucionario cuántico [11], el algoritmo genético cuántico [12] y el algoritmo VQE (Variational Quantum Eigensolver) [13], con lo cual empieza a vislumbrarse una línea de investigación en este ámbito.

Por su parte, la computación cuántica nace como una alternativa al paradigma computacional convencional basado en máquinas de Turing y de Von Neumann, la cual ha demostrado su superioridad ante la computación clásica para algunos problemas específicos [14], [15]. 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 [16].

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 [17], 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 [2][11].

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 [18].

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.

## **1.2 APORTES DEL PROYECTO**

Con el desarrollo de este proyecto se busca contribuir a la línea de investigación de Sistemas Inteligentes del Grupo de I+D en Tecnologías de la Información (GTI), buscando mostrar la forma como se puede abordar un problema de optimización binaria con el enfoque de computación cuántica adiabática, dando a conocer el funcionamiento de un algoritmo adiabático cuántico aplicado al problema de la mochila binaria, y comparando los resultados obtenidos frente a algoritmos del estado del arte en instancias de baja dimensionalidad (n < 20), todo desde la perspectiva de la computación cuántica para la solución de un problema teórico de optimización ampliamente conocido y que puede ser aplicado en la solución de diversas situaciones de la vida real.

Se espera que las herramientas ofrecidas para solucionar problemas de optimización mediante computación cuántica evolucionen y que con este proyecto se sienten las bases para que en el futuro se puedan realizar propuestas e investigaciones que manejen problemas de la mochila con alta dimensionalidad.

## **1.3 OBJETIVOS**

A continuación, se presentan los objetivos como fueron aprobados en el Anteproyecto por parte del Consejo de Facultad de la Facultad de Ingeniería Electrónica y Telecomunicaciones.

### **1.3.1 Objetivo general**

Proponer un algoritmo cuántico para resolver instancias de baja dimensionalidad del problema de la mochila binaria basado en conceptos de computación cuántica adiabática.

### **1.3.2 objetivos específicos**

* Establecer la línea base de la investigación con el modelado e implementación de un marco de prueba implementado en Python que incluya tres (3) algoritmos del estado del arte que resuelven el problema de la mochila binaria en instancias de baja dimensionalidad (n<20) con diferentes grados de complejidad y métricas de comparación (tasa de éxito para encontrar el resultado óptimo, mejor óptimo promedio encontrado y tiempo de ejecución) reconocidas por la comunidad científica.
* Definir un algoritmo cuántico y su implementación en Python sobre QiskitTM Aqua, usando el proceso de investigación iterativo propuesto por Pratt, para resolver el problema de la mochila binaria en instancias de baja dimensionalidad.
* Documentar el desempeño del del algoritmo cuántico propuesto, a partir de un estudio comparativo de los resultados de este respecto a los algoritmos del estado del arte implementados en el marco de prueba, en función de las métricas e instancias disponibles.

## **1.4 RESULTADOS OBTENIDOS**

A continuación, se resumen los principales resultados obtenidos en la elaboración del presente trabajo de grado:

* **Monografía de trabajo de grado**: Se refiere al presente documento en el cual se presenta el estado del arte en el campo de problemas de optimización binaria, un marco de trabajo implementado con el cual se evalúan cuatro algoritmos con enfoques de solución diferentes y la definición, implementación y pruebas del marco de trabajo evaluado con un algoritmo cuántico y tres algoritmos del estado del arte, por último, se exponen y analizan los resultados obtenidos al evaluar el marco de trabajo.
* **Framework de pruebas:** Se refiere al código fuente del marco de trabajo construido para realizar las pruebas e implementar los algoritmos que dan solución a problemas de optimización de mochila binaria, el código fuente del marco de trabajo se encuentra disponible en [19] y en un anexo digital (**Anexo A**) de esta monografía. La documentación del código se presenta en el **Anexo B**.
* **Ponencia:** el articulo titulado*“Computación Cuántica Adiabática aplicada a la solución del Problema de la Mochila Binaria”*, logro aplicar y ser aceptado para realizar una ponencia en la Jornada iberoamericana de ingeniería de software e ingeniería del Conocimiento, JIISIC’2020.
* **Artículo publicado:** artículo titulado *“Computación Cuántica Adiabática aplicada a la solución del Problema de la Mochila Binaria”* con las pruebas y evaluación del algoritmo cuántico consignado en el **Anexo C**, el cual se encuentra publicado en la Revista Ibérica de Sistemas e Tecnologias de Informação - Risti (ISSN: 1646-9895), Aceptado el 25/11/2020
* **Artículo final de la investigación**: artículo con la descripción y resultados del marco de trabajo propuesto en este documento, el cual ha sido titulado **“nombre”** y que se encuentra en proceso de envío a una revista JCR Q1, como Knowledge-Based Systems o Information Sciences. Ver **Anexo D**.

## **1.5 ESTRUCTURA DE LA MONOGRAFÍA**

A continuación, se describe de manera general el contenido y organización de la presente monografía:

**CAPITULO 1: INTRODUCCIÓN**: Hace referencia al presente capitulo que introduce el tema de investigación, presenta la pregunta de investigación que origino el trabajo, los aportes al problema, también los objetivos (general y específicos) definidos en el anteproyecto, un breve resumen de los resultados obtenidos, aportes y finalmente la organización de la monografía.

**CAPITULO 2: CONTEXTO TEÓRICO Y ESTADO DEL ARTE**: En este capítulo se presentan conceptos teóricos relacionados con computación cuántica. Además, se presentan propuestas relevantes del estado del arte que se han venido aplicando y mejorando con el paso del tiempo para la resolución de problemas de optimización.

**CAPITULO 3: FRAMEWORK PARA SOPORTAR LA EXPERIMENTACIÓN:** En este capítulo se explica la estructura y funcionamiento general del marco de trabajo desarrollado para la ejecución de los experimentos, así como un ejemplo de construcción y ejecución de un experimento.

**CAPITULO 4: EXPERIMENTOS Y RESULTADOS**: Se presenta el protocolo de experimentación y se presentan los resultados de los algoritmos de la línea base y del algoritmo propuesto en los distintos escenarios de prueba. También se presenta un análisis comparativo de los resultados soportado con las pruebas estadísticas no paramétricas de Friedman y Wilcoxon.

**CAPITULO 5: CONCLUSIONES Y TRABAJOS FUTUROS**: En este capítulo se presentan las conclusiones obtenidas al finalizar el trabajo de grado e ideas que el grupo de investigación espera realizar en un trabajo futuro.

**CAPITULO 6: BIBLIOGRAFIA**: Este último capítulo contiene las referencias bibliográficas de sitios web, artículos y libros consultados para la realización del proyecto.

CAPÍTULO 2

# **CONTEXTO TEÓRICO Y ESTADO DEL ARTE**

## **2.1** **CONTEXTO TEÓRICO**

### **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 [16].

En computación cuántica, un bit cuántico (Quantum bit, qubit) es la unidad de información más pequeña almacenada en una computadora cuántica de dos estados [19]. 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) [20].

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 [16]. Entre los algoritmos cuánticos más famosos se encuentran el algoritmo de Shore’s [21] utilizado para factorización numérica y el algoritmo de Grover’s [22] utilizado para búsquedas en una base de datos no ordenada. Ambos algoritmos redujeron la complejidad de la solución al problema [23]; 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 [17].

Durante la última década, la computación cuántica ha atraído un interés generalizado y ha inducido intensivas investigaciones, debido especialmente a su paralelismo innato que reduce la complejidad algorítmica. Tal capacidad de procesamiento en paralelo se puede utilizar para resolver problemas de optimización combinatoria que requieren la exploración de grandes espacios de posibles soluciones [24]. 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 [24].

### **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 [2].

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 [15]. 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 [25].

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. En general las computadoras cuánticas universales no pueden resolver problemas NP-hard de manera eficiente, pero se ha encontrado evidencia en los sistemas experimentales de Ising cuántico que sugiere una aceleración cuántica sobre la computación tradicional debido al efecto del túnel cuántico (hace referencia a pasar de un estado A a un estado B no contiguo para evitar estancarse en óptimos locales) [25].

### **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 [15]. 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 [15][25][26].

|  |  |
| --- | --- |
|  | (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 [27]). Con esta definición y debido al teorema adiabático de la mecánica cuántica el sistema cuántico permanecerá en el estado fundamental todo el tiempo.

**2.1.4 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 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[[1]](#footnote-1) que actúa sobre el i-ésimospin 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, Chudak, Macready, & Rose, 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).

Con base en lo anterior, en [26] 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. El documento no presenta una implementación ni resultados de ejecución de los algoritmos, además menciona, pero no desarrolla la reducción del vector solución de una cantidad N de qubits a N.

## 

## **ESTADO DEL ARTE**

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 algoritmos exactos, algoritmos de programación dinámica, algoritmos basados en back-tracking (incluidos ramificación y poda), algoritmos metaheurísticos y recientemente algoritmos basados o que simulan lógica cuántica.

Entre los algoritmos metaheurísticos más destacados se encuentran los algoritmos genéticos, el recocido simulado, la optimización por enjambre de partículas (PSO), la búsqueda tabú [3], [10], el algoritmo evolutivo cuántico [11], el algoritmo genético cuántico [12] y el algoritmo VQE (Variational Quantum Eigensolver) [13].

### **Trabajos previos en el ámbito de algoritmos clásicos**

Varios algoritmos propuestos en la literatura para resolver el problema de la mochila binaria tienen baja precisión y caen fácilmente en soluciones óptimas locales. Para superar estos problemas, en 2016 [28] se propone una versión binaria del algoritmo del mono (MA). Para validar la eficiencia del algoritmo propuesto, se realizan experimentos con varias instancias y se comparan los resultados con cinco algoritmos metaheurísticos reportados en la literatura (BPSO, MBPSO, NGHS, DGHS y S-bAFSA). Los experimentos muestran que el algoritmo CGMA propuesto tiene grandes ventajas en la resolución de problemas de la mochila binaria fijos y aleatorios y problemas de pequeña y gran escala. Dado que las pruebas de CGMA se realizaron contrastando los resultados de algoritmos de diferentes tipos y con una amplia variedad de instancias, CGMA se consideró en su fecha de publicación, una alternativa eficaz para resolver problemas binarios de la mochila.

En 2017 [29] se propone una metaheurística hibrida para solucionar el problema de la mochila fuertemente correlacionada (SCKP). Se propone un algoritmo Hibrido de Optimización de Colonias de Hormigas ACO el cual combina el Sistema de Hormigas MAX-MIN y el Sistema de Colonias de Hormigas con el algoritmo 2-optimal, los cuales se ejecutan secuencialmente (la salida del primer algoritmo es la entrada del segundo). El algoritmo MMACS propuesto tiene como objetivo resolver óptimamente problemas SCKP, en caso de que no se encuentre una solución óptima, se utiliza el algoritmo 2-optimal; si la heurística 2-optimal no logra encontrar una solución óptima, al menos mejorará la calidad de la solución al reducir la brecha entre la solución encontrada y la óptima. Con este nuevo algoritmo se pretende mejorar las soluciones encontradas por el algoritmo ACO mediante el algoritmo 2-optimal para obtener mejores soluciones manteniendo un tiempo de ejecución reducido. El algoritmo propuesto se probó en un conjunto de instancias de alta y media dimensionalidad y se contrastó con el Algoritmo evolucionario inspirado en Cuántica (QEA). El algoritmo QEA no consigue encontrar soluciones apropiadas a los problemas. Por otro lado, el algoritmo propuesto es ineficiente en instancias de baja dimensionalidad, pero en la medida que la dimensionalidad crece encuentra soluciones óptimas. Este algoritmo propuesto tiene el problema de funcionar solo en instancias SCKP y no en otros tipos de problemas de la mochila.

Ese mismo año (2017) [30] se realiza el análisis de los algoritmos de la Búsqueda tabú (Tabú Search, TS), la Búsqueda Dispersa (Scatter Search, SS) y un algoritmo de búsqueda local (Local Search, LS). El objetivo del estudio fue determinar la eficiencia y precisión de cada uno en la solución del problema de la mochila binaria. Las pruebas se realizaron con el software HeuristicLab framework, donde los algoritmos se comparan en función de la solución con mejor calidad, la diferencia de las soluciones entre la solución con mejor calidad y los tiempos de ejecución; medidos y comparados en un total de treinta observaciones tomando instancias de dimensionalidad media. Como resultado se tiene que el algoritmo SS registra la menor complejidad de tiempo (menor tiempo de ejecución) y el algoritmo TS consigue la menor desviación de la solución con la mejor calidad de la mochila. Como trabajos futuros, los autores plantean el uso de algoritmos metaheurísticos, como firefly, colonia de hormigas o GRASP, y comparar los resultados con más métricas y diferentes tamaños de muestra.

También en 2017 [31] se propone un algoritmo binario de araña social (BSSA) para resolver el problema de la mochila binaria. La principal contribución de este estudio es que se propone el primer algoritmo binario de araña social que se combina con dos técnicas de manejo de restricciones para el problema de la mochila binaria. El algoritmo propuesto integró la exploración del algoritmo de araña social (SSA) y la explotación con un operador de reparación, además proponen dos técnicas de restricción basadas en el factor de penalización y una estrategia codiciosa para mejorar la eficiencia del algoritmo propuesto. Los resultados de la simulación en cinco instancias recientes según la literatura y con conjuntos de datos fuertemente correlacionados demuestran que el algoritmo propuesto tiene un rendimiento superior en comparación con un algoritmo genético y un algoritmo basado en PSO, sin embargo, las pruebas no son lo suficientemente extensas para concluir la eficacia del algoritmo propuesto.

El algoritmo de Optimización de enjambre de partículas binarias (BPSO) original y sus variantes no pueden proporcionar resultados totalmente satisfactorios debido al uso de funciones de desplazamiento inapropiadas, estas funciones no le proporcionan al algoritmo un buen equilibrio entre exploración y explotación en el espacio de búsqueda, lo que limita su desempeño. Para superar este problema, en 2017 [32] se propone agregar una función de desplazamiento variable en el tiempo denominada TVT-BPSO. Los resultados experimentales presentados demuestran que TVT-BPSO supera a las variantes de BPSO y a tres variantes de BPSO propuestas para solucionar el problema de mochila binaria utilizando instancias de baja dimensión, alta dimensión y un problema de truss de 200 individuos. Los autores sugieren que TVT-BPSO puede escalar mejor a problemas combinatorios de alta dimensión que las variantes existentes y aunque no se evalúe la solución propuesta con diferentes implementaciones de otros algoritmos metaheurísticos, se sugiere que la solución propuesta puede lograr resultados competitivos o mejores.

En 2018 [33] se presenta un acelerador de hardware basado en FPGA para reducir el tiempo de procesamiento requerido para resolver problemas de la mochila binaria en instancias de baja, media y alta dimensionalidad utilizando el algoritmo de la Búsqueda Armónica Binaria (BHS). Los resultados experimentales revelaron una aceleración significativa en comparación con dos implementaciones de software paralelas del mismo algoritmo, así como una implementación de hardware que utiliza el Algoritmo de optimización por enjambre de partículas binarias (BPSO). Todos los resultados obtenidos por las distintas implementaciones son óptimos lo que indica que la calidad de las soluciones no se vio afectada, sin embargo, se observa una diferencia considerable en el tiempo de ejecución de BHS frente a BPSO.

También en 2018 [34] se presenta un nuevo algoritmo de optimización caótico basado en la mariposa monarca (CMBO). En el algoritmo de la mariposa monarca (MBO) se introduce la teoría del caos con el objetivo de acelerar la optimización y mejorar las capacidades de búsqueda global/local. Se utilizan doce mapas caóticos unidimensionales para ajustar los parámetros de CMBO y una mutación gaussiana con la cual se perturba una pequeña parte de las soluciones con peor aptitud. El desempeño de CMBO se verificó y analizó con tres grupos de instancias de problemas de la mochila binaria (no correlacionados, débilmente correlacionados y fuertemente correlacionados). Los resultados muestran que la introducción de un mapa caótico apropiado y la perturbación gaussiana pueden mejorar significativamente la calidad de la solución junto con el rendimiento general del algoritmo propuesto. El CMBO propuesto puede superar al MBO estándar y los algoritmos ABC, CS, DE, GA, FA, SFLA, HS y MBO, pero no se tiene en cuenta la dimensionalidad que se puede manejar en los diferentes problemas de mochila.

Algunos algoritmos metaheurísticos pueden fallar al quedar atrapados en un óptimo local, por ello en 2019 [35] se propone un algoritmo de optimización de ballenas basado en oposición (OWOA) que permita encontrar la solución a problemas de la mochila binaria. La oposición se realiza calculando el vector opuesto de una posible solución, por ejemplo, si un candidato es (001000101), el vector opuesto será (110111010). Se realizaron 3 experimentos para validar el algoritmo propuesto, en el primer experimento se comparó WOA con OWAO ejecutado la comparación 100 veces y encontrando el valor medio de los resultados, y en el segundo y tercer experimento se validó el rendimiento de OWOA comparando el resultado con los algoritmos HS-Jaya y CGMA utilizando 10 casos de problemas de mochila diferentes. Los resultados obtenidos muestran que en el valor medio de los resultados se nota una mejora notable en el rendimiento en comparación con OWOA. Como trabajos futuros se recomienda resolver diferentes problemas de optimización, como pruebas de software y problemas del vendedor viajero, y estudiar una nueva variante de OBL para mejorar WOA.

Los autores de otro estudio publicado en 2019 [36] se fijan el objetivo observar los efectos de los lobos dominantes en la eficiencia de la metaheurística de Optimización del Lobo Gris (GWO). Para probar el desempeño de los enfoques desarrollados y observar los efectos de los lobos dominantes modificados, se emplean tres conjuntos diferentes de evaluaciones comparativas que incluyen problemas continuos, combinatorios, sin restricciones y con restricciones. Entre los problemas escogidos para probar el enfoque propuesto se utilizó el problema de la mochila binaria. Todas las modificaciones desarrolladas se comparan con el GWO estándar y el algoritmo de optimización por enjambre de partículas (PSO). El estudio experimental y los resultados verificados estadísticamente demuestran que los lobos dominantes en GWO tienen efectos cruciales en la eficiencia del GWO estándar. Además, las pruebas estadísticas demuestran que las modificaciones de GWO desarrolladas superan significativamente a algunos de los algoritmos informados en la literatura relacionada, como por ejemplo PSO. Para dar solución al problema de la mochila los artículos son ubicados en orden decreciente de acuerdo con la densidad de cada artículo (beneficio/peso), luego los artículos clasificados se asignan a la mochila hasta que se excede la capacidad de la mochila; este método de llenado podría dejar soluciones optimas por fuera del estudio.

### **Trabajos previos en el ámbito de computación cuántica**

En 2017 [2] se propone un algoritmo para solucionar el problema de la mochila con ganancias y pesos enteros, mediante computación cuántica adiabática. En este documento se realizan dos implementaciones, una utiliza un Ising Hamiltoniano que requiere n + c qubits para representar la solución y otra más eficiente la cual utiliza n + [ C] + 1 qubits. En este documento se puede observar que reducir la cantidad de qubits aplicando una trasformación logarítmica tiene un impacto positivo en el rendimiento del algoritmo, ya que reduce considerablemente la cantidad de qubits necesarios para procesar una solución. En el trabajo presentado no se realizan pruebas comparativas detalladas con diferentes tipos de algoritmos del estado del arte, ya sean algoritmos clásicos, otras propuestas cuánticas puras o híbridos.

En 2019 [11] se propone un Algoritmo Evolutivo Genético Cuántico Mejorado (AEC-M) para solucionar el problema de la mochila binaria, el cual esta basado en la tecnología de catástrofe del ángulo de rotación dinámico donde se diseña un operador de puerta giratoria cuántica que ajusta de forma adaptativa los valores del ángulo de rotación de los qubits de acuerdo con el valor de aptitud y las generaciones de la evolución. Lo anterior se hace con el objetivo de que los spines de cada qubit del cromosoma apunten en la dirección de la solución. Se evalúa la solución propuesta con un algoritmo evolutivo cuántico (AEC) y un algoritmo evolutivo clásico AE. Los resultados experimentales permiten observar que (AEC-M) tiene un mejor rendimiento que AEC y que AE. Si se varia la cantidad de generaciones a evolucionar con respecto del valor de aptitud obtenido, se puede observar que AEC con cantidades de generaciones a evolucionar pequeñas (1 - 100) arroja valores de aptitud estables, visualizándose una curva de convergencia muy suave hasta estabilizarse en un valle que varía muy poco; a diferencia de AEC-M el cual arroja mejores resultados de aptitud con un numero grande en la cantidad de generaciones a evolucionar, pero el crecimiento de la curva es muy brusco, con lo cual se observa que resultan óptimos locales muy a menudo.

CAPÍTULO 3

# **MARCO DE TRABAJO PARA REALIZAR LA EVALUACION Y COMPARACION DE LOS ALGORITMOS SELECCIONADOS**

## **DESCRIPCION DEL MARCO DE TRABAJO**

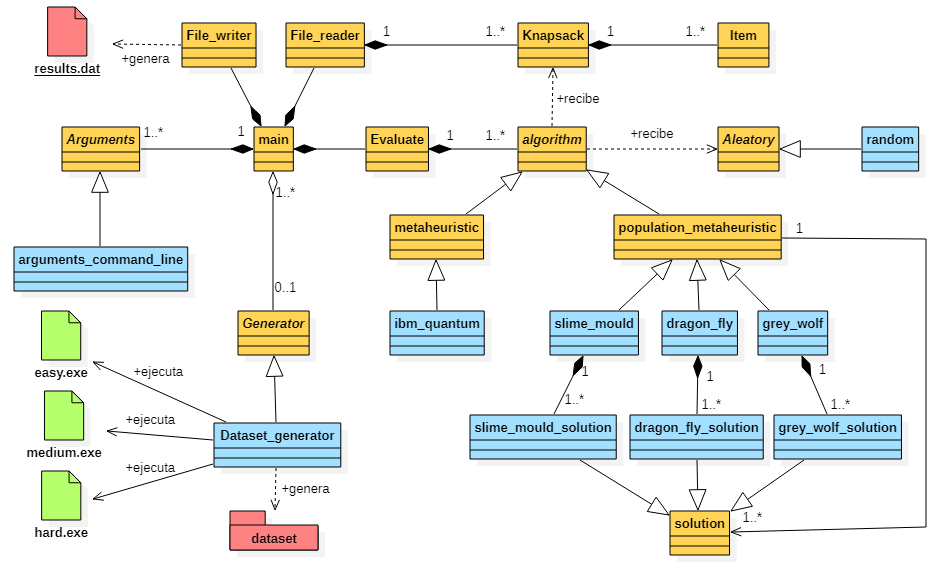
En la actualidad el manejo real que se tiene para el desarrollo de soluciones cuánticas a problemas de optimización se enfoca en modelos de circuitos cuánticos, los cuales trabajan con programación a bajo nivel, esto quiere decir con compuertas cuánticas que alteran el estado de un qubit el cual funciona sobre una máquina que trabaja bajo las leyes de la mecánica cuántica, lo cual 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 utilizan 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 aun no esta 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 debido a que es una de las librerías que tiene una base de desarrollo 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 la **figura de abajo** se muestra una vista de alto nivel del algoritmo implementado. Las clases de color azul corresponden al núcleo del marco de trabajo, y las de color verde son clases derivadas asociadas a implementaciones de algoritmos específico y los artefactos en color verde corresponden a los algoritmos generadores de los conjuntos de datos.



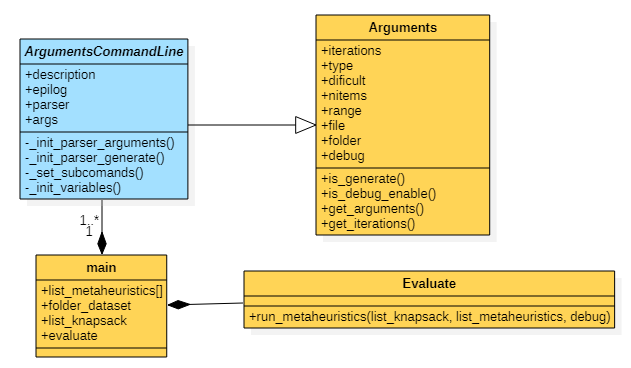
El marco de trabajo de divide en cinco módulos:

* **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.

En esta sección se explica brevemente cada uno de los módulos mencionados. Adicionalmente, una documentación completa se encuentra disponible en el **Anexo B**.

### **3.1.2 Modulo principal (main)**

En la figura de abajo se muestran las clases que hacen parte del módulo principal. 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.



#### **3.1.2.1 Submódulo de argumentos**

Se contempla implementar una clase abstracta para que le programa se pueda escalar y agregar las clases concretas que se consideren pertinentes dependiendo del modelo de argumentos que se desee utilizar. La clase abstracta **Arguments** contiene cuatro métodos básicos para la gestión de los argumentos ingresados en la ejecución del programa, estos son:

* **is\_generate().** indica si se debe generar un nuevo conjunto de datos
* **get\_iterations().** Retorna la cantidad de iteraciones que definió el usuario, con este valor se define cuantas veces se debe ejecutar cada algoritmo
* **get-arguments():** retorna una instancia de tipo **ArgumentParser** la cual contiene los argumentos ingresados por el usuario en la línea de comandos al momento de la ejecución del programa.
* **is\_debug\_enable():** retorna True si el usuario desea que el programa se ejecute en modo de depuración, retorna False en caso contrario.

Debido a que el programa se ejecuta por línea de comandos, se tiene la clase concreta **Arguments\_command\_line** la cual se encarga de implementar los métodos heredados de la clase padre y además de eso realiza la configuración del menú de ayuda para el usuario, el cual admite los siguientes argumentos:

Argumentos opcionales:

* **[-h, --help].**

muestra el mensaje de ayuda

* **[-i ITERATIONS, --iterations ITERATIONS]**

número de iteración para correr cada archivo de mochila disponible

* **[-d, --debug]**

le indica al programa que habilite la salida de texto a modo de información para poder realizar una depuración

* **[-fl FILE, --file FILE]**

indica el nombre o ubicación de un archivo con la información de una mochila a evaluar.

* **[-fd FOLDER, --folder FOLDER]**

indica el nombre o ubicación de una carpeta que contenga archivos con la información de las mochilas que se desean evaluar.

Subcomandos

* **{generate}**

Subcomando encargado de realizar la gestión de las opciones para generación de generación del conjunto de datos. Se profundiza en el apartado correspondiente al módulo generador

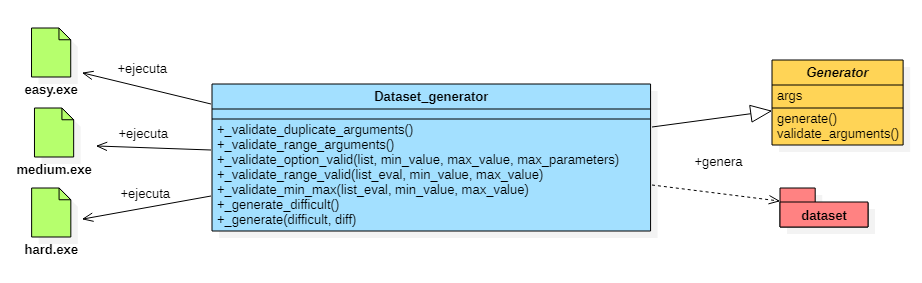
#### **3.1.2.1 Submódulo principal (main)**

Las clases **main** y **evaluate** se encargan de orquestar todo el funcionamiento del marco de trabajo, entre sus funciones principales se encuentran:

* Gestionar los algoritmos que serán objeto de pruebas.
* Gestionar la generación u obtención del conjunto de datos necesario para realizar la evaluación.
* Coordinar la ejecución de todas las instancias de los algoritmos seleccionados de acuerdo a los parámetros ingresados por el usuario.

### **3.1.2 Modulo generador**

En la figura de abajo se muestra el módulo generador encargado de generar el conjunto de datos pertinentes de acuerdo a la configuración ingresada por el usuario, en ella se puede observar que posee 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.



Para habilitar esta funcionalidad es necesario indicarle al programa mediante línea de comandos que debe realizar la generación. Este procedimiento se realiza enviando el subcomando {generate} en la ejecución del programa. Este subcomando admite los siguientes argumentos:

* **[h, --help]]**

muestra el mensaje de ayuda

* **[-t TYPE [TYPE ...], --type TYPE [TYPE ...]]**

indica el tipo de correlacion entre los elementos de la mochila. Opciones: ([1=uncorrelated, 2=weakly correlated, 3=strongly correlated, 4=subset sum])

* **[-d DIFFICULT [DIFFICULT ...], --difficult DIFFICULT [DIFFICULT ...]]**

indica la dificultad del conjunto de datos a generar. Opciones: ([1=Easy, 2=Medium, 3=Hard])

* **[-n NITEMS [NITEMS ...], --nitems NITEMS [NITEMS ...]]**

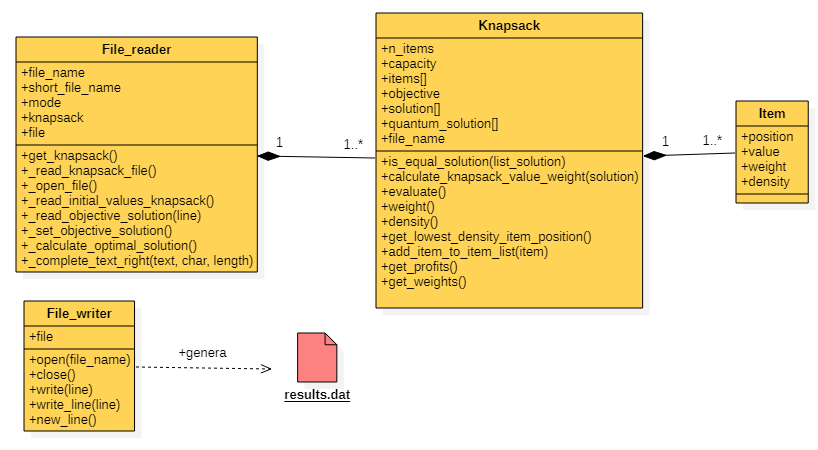
indica la cantidad de items almacenados en cada mochila; si se envían dos argumentos se tomará como un rango, si se envían más de dos se tomará como una lista. [ { n1 - n2} <> {n1, n2,..., n} ]

* **[-r RANGE [RANGE ...], --range RANGE [RANGE ...]]**

indica el rango de creacion del conjunto de datos; si se envían dos argumentos se tomará como un rango, si se envían más de dos se tomará como una lista. [ { n1 - n2} <> {n1, n2,..., n} ]

### **3.1.2 Modulo de archivos**

En la figura de abajo se muestra el módulo de archivos encargado de gestionar el acceso a los archivos necesarios en la ejecución del algoritmo. Este modulo 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.



### **3.1.2.1 File\_reader**

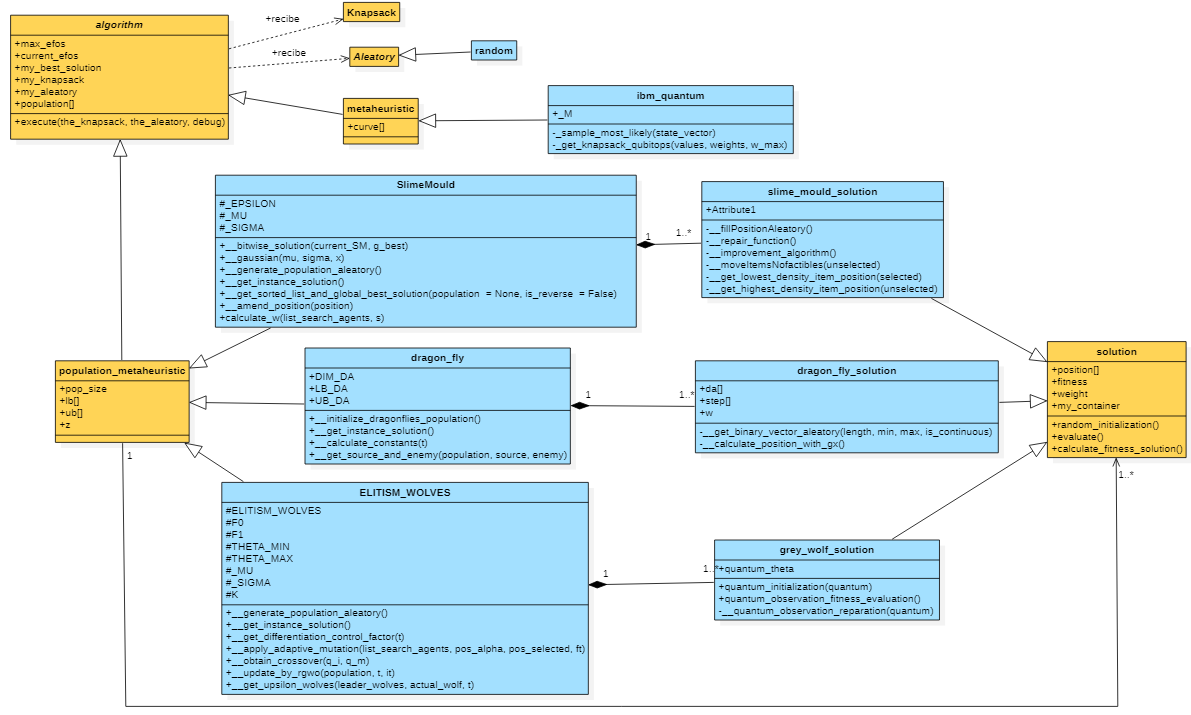
Esta clase se encarga de realizar la lectura de un archivo que contiene la información de una mochila. Debido a que se trabaja con instancias de baja dimensionalidad, si el archivo no contiene una solución objetivo, esta será calculada por medio de un algoritmo de fuerza bruta que evalúa todas las posibles combinaciones y entre ellas selecciona la mejor. Al momento de realizar la lectura se va construyendo una instancia de la clase **Knapsack** la cual almacena una lista que contiene instancias de la clase **Item** y en conjunto generan una abstracción de la mochila. finalmente se puede obtener la instancia de la mochila con la invocación del método get\_knapsack().

### **3.1.2.1 File\_writer**

Esta clase se encarga de realizar la escritura de los resultados de la evaluación de los algoritmos seleccionados en un archivo de texto.

### **3.1.4 MODULO DE ALGORITMOS**

En la figura de abajo se muestra el 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.



#### **3.1.4.1 Algorithm**

Esta clase abstracta se encarga de definir el método de ejecución que deben implementar las clases concretas, además, define las variables globales que servirán como parámetros de ejecución de cada algoritmo, entre ellas se define una variable que almacena la información de la mochila, una variable que define el mecanismo de aleatoriedad que se desea utilizar y una variable donde se almacena la mejor solución encontrada.

Además de considerar una metaheurística general, la escogencia de los algoritmos que van a ser objeto de estudio requiere que se tenga una metaheurística basada en población, con ella se permite agregar algoritmos que simulan metaheurísticas poblacionales o de enjambre.

#### **3.1.4.1 Solution**

Esta clase abstracta define los métodos necesarios para realizar las operaciones que requiera cada algoritmo concreto, tiene una relación de dependencia con la clase **Algorithm** ya que mediante esta clase se logra realizar operaciones de cruce, mutación, evaluación, cálculo de fitness entre otras operaciones a las instancias de solución evaluadas en cada iteración. Esta clase define dos constructores para su inicialización, un constructor se encarga de inicializar una instancia vacía y con el otro se puede realizar una copia de una solución previamente definida.

#### **3.1.4.1 ibm\_quantum**

En esta clase de encuentra el desarrollo de toda la lógica de simulación cuántica que se encarga de calcular los operadores de Pauli que sirven de insumo para la clase ExactEigensolver la cual es la instancia de Qiskit encargada de encontrar la solución al problema de mochila especifico.

Las clases restantes marcadas en color azul corresponden a las implementaciones de los algoritmos del estado del arte seleccionados para realizar el estudio comparativo.

CAPÍTULO 4

# **PROBLEMA DE LA MOCHILA BINARIA ABORDADO MEDIANTE COMPUTACION CUANTICA**

## **DESCRIPCIÓN**

* + 1. **Modulo 1**
    2. **Modulo 2**
    3. **Modulo 3**

CAPÍTULO 5

# **EXPERIMENTOS Y RESULTADOS**

CAPÍTULO 6

# **CONSCLUSIONES Y TRABAJOS FUTUROS**

* 1. **CONSLUSIONES**
  2. **TRABAJOS FUTUROS**

CAPÍTULO 7

# **BIBLIOGRAFIA**

1. Matriz de 2x2 cuyo primo tiene vectores propios con valores propios 0,1 (Lucas, 2014) [↑](#footnote-ref-1)