UCUENCA

Universidad de Cuenca

Facultad de Ingeniería

Carrera de Ingeniería de Sistemas

Desarrollo de un algoritmo híbrido metaheurístico para el problema de distribución de planta en las MIPYMES textiles del Ecuador

Trabajo de titulación previo a la obtención del título de Ingeniero de Sistemas

Autor:

Luis Miguel Sotamba Once

Director:

Lorena Catalina Sigüenza Guzmán

ORCID: 0000-0003-1367-5288

Cuenca, Ecuador

2023-09-20

Resumen

El Problema de Distribución de Instalaciones (FLP por sus siglas) se refiere a encontrar la disposición más efectiva de las instalaciones en una planta de fábrica, considerando varios aspectos. Estas instalaciones podrían incluir departamentos, personal o maquinaria, por mencionar algunos ejemplos. El FLP es un problema NP-Hard, por lo que no existen algoritmos capaces de proporcionar una solución óptima en un tiempo polinomial razonable. Debido a esto, los investigadores han hecho uso de metaheurísticas, y han intentado combinarlos para obtener híbridos de estos. Las metaheurísticas híbridas pueden aprovechar las fortalezas de cada enfoque para generar soluciones de mayor calidad y cercanas a la óptima. Sin embargo, muchos de los estudios que desarrollan una metaheurística híbrida no usan una metodología que explique el proceso de desarrollo. Por dicha razón, este trabajo se enfoca en componer una metaheurística híbrida siguiendo una metodología que fomenta el uso del análisis FODA, el Enfoque de Elección Estratégica y el Pensamiento Convergente/Divergente. La metaheurística híbrida llamada GENTSA fue construida utilizando un Algoritmo Genético, Recocido Simulado y Búsqueda Tabú. GENTSA resuelve el FLP que emerge en las MiPYMES textiles del Ecuador, la misma que fue usada como caso de estudio. Además, como parte de su validación, se usaron funciones de prueba obtenidas del estado del arte. GENTSA fue comparado con otras metaheurísticas en donde obtiene la mejor distribución de instalaciones con el menor costo; y se ajusta razonablemente bien a dominios de problemas más allá de su alcance de diseño original.

Palabras clave: distribución de instalaciones, metaheurística híbrida, MiPYMES

Abstract

The Facility Layout Problem (FLP) refers to finding the most effective arrangement of facilities in a factory, considering various aspects. These facilities could include departments, personnel, or machinery, to name a few examples. The FLP is an NP-Hard problem, meaning that there are no algorithms capable of providing an optimal solution in a reasonable polynomial time. Due to this, researchers have turned to metaheuristics and have attempted to combine them to create hybrids. Hybrid metaheuristics can leverage the strengths of each approach to generate higher-quality solutions that are close to optimal. However, many studies that develop hybrid metaheuristics do not use a methodology that explains the development process. For this reason, this work focuses on composing a hybrid metaheuristic following a methodology that encourages the use of SWOT analysis, the Strategic Choice Approach, and Convergent/Divergent Thinking. The hybrid metaheuristic called GENTSA was constructed using a Genetic Algorithm, Simulated Annealing, and Tabu Search. GENTSA solves the FLP that arises in small and medium-sized textile businesses in Ecuador, which was used as a case study. Additionally, as part of its validation, test functions obtained from the state of the art were used. GENTSA was compared to other metaheuristics where it achieves the best facility layout with the lowest cost and reasonably adapts to problem domains beyond its original design scope.

Keywords: facility layout, hybrid metaheuristic, MiPYMES

Índice de contenido

Índice de figuras

Índice de tablas

Capítulo I – Introducción

Motivación y contexto

El Problema de Distribución de Planta (FLP por sus siglas en inglés de Facility Location Problem) es un problema de optimización que involucra determinar el arreglo más efectivo de las instalaciones en la planta de una fábrica, basado en criterios, objetivos y restricciones específicas. FLP puede ser categorizado en dos tipos según la evolución del diseño de la planta: FLP estático (SFLP por sus siglas en inglés) y FLP dinámico (DFLP por sus siglas en inglés). SFLP considera que el flujo de materiales entre las instalaciones permanece constante a lo largo del tiempo, mientras que DFLP indica que el flujo de materiales cambia con el tiempo debido a factores como los ciclos de vida cortos de los productos, el reemplazo de maquinaria y los cambios en el diseño de productos, por mencionar algunos (Hosseini-Nasab et al., 2018). El FLP tiene una gran importancia en el campo de la ingeniería industrial, ya que un diseño adecuado puede ayudar a reducir el transporte de materiales y personal, lo que conduce a una disminución de hasta el 50% en los costos operativos totales (Drira et al., 2006). El FLP ha recibido considerable atención por parte de los investigadores debido a que se considera un problema NP-Hard, lo que significa que no existen algoritmos capaces de proporcionar una solución óptima en un tiempo polinomial razonable (Pérez-Gosende et al., 2021). Resolver este problema presenta un desafío para los investigadores, ya que implica considerar diversas restricciones organizativas para lograr el diseño más efectivo. Para abordar el FLP, Flores-Siguenza et al. (2022) realizaron una revisión de literatura, teniendo en cuenta varios enfoques de solución utilizados por los investigadores. Los resultados indican que los algoritmos metaheurísticos son los enfoques comúnmente más utilizados. Los algoritmos metaheurísticos son estrategias que guían el proceso de búsqueda, explorando eficientemente el espacio de soluciones disponibles para encontrar soluciones óptimas o casi óptimas (Blum, 2003). La ventaja de las metaheurísticas radica en su adaptabilidad a cualquier problema de optimización, donde cada algoritmo define ciertas reglas para la exploración del espacio de soluciones (diversificación) y posteriormente explotar la experiencia acumulada de búsqueda (intensificación).

Sin embargo, según el teorema de "No Free Lunch", no es posible esperar que una metaheurística funcione bien para todos los problemas de optimización (Wolpert & Macready, 1997). Cada metaheurística tiene sus propias debilidades y fortalezas; y ciertas metaheurísticas pueden ser más adecuadas para problemas de optimización específicos. Debido a esto, en la última década ha surgido un enfoque que busca combinar las ventajas de múltiples metaheurísticas para formar algoritmos híbridos. Una metaheurística híbrida intenta simultáneamente minimizar las desventajas de los algoritmos que la conforman, e

incluso puede ofrecer mejoras en velocidad computacional o calidad de la solución (Ting et al., 2015). Sin embargo, la falta de uso de una metodología para la hibridación de metaheurísticas ha resultado en una hibridación aleatoria, lo cual debe evitarse (Ting et al., 2015).

Debido al desafío de seleccionar las metaheurísticas adecuadas para problemas de optimización específicos, junto con la falta del uso de una metodología de hibridación que guíe el análisis, desarrollo y comparación de las metaheurísticas para la composición de una metaheurística híbrida aplicada a un caso de estudio del mundo real, es la principal motivación para llevar a cabo el presente estudio.

Pregunta de Investigación

¿Es posible desarrollar un algoritmo híbrido metaheurístico para encontrar una solución al problema de distribución de planta en las MIPYMES textiles del Ecuador?

Respuesta Propuesta

El presente estudio está sujeto a resolver el problema de distribución de planta que surge en las MIPYMES textiles del Ecuador a través del desarrollo de un algoritmo híbrido metaheurístico siguiendo una metodología obtenida a partir de la revisión de la literatura. Como resultado de este proceso, la composición del algoritmo pueda ser reproducible y comprensible.

Contexto del proyecto

Este trabajo de titulación se enmarca en el proyecto de investigación ResilTEX – Modelo Resiliente de Distribución de Planta para MIPYMES con un Enfoque en Productividad y Seguridad Ocupacional. ResilTEX incorpora una empresa textil ecuatoriana como caso de estudio para validar el modelo FLP propuesto, la cual es usada de igual manera en este trabajo. En este contexto, a través de un algoritmo híbrido metaheurístico, se buscó encontrar la mejor disposición de facilidades en la distribución de planta.

ResilTEX está enfocado en permitir la detección de zonas críticas en la distribución de planta con respecto a la seguridad ocupacional y su efecto en la productividad dentro de las empresas textiles del Ecuador. Para mayor información sobre este proyecto, referirse a Llivisaca et al. (2022).

Justificación y objetivos

A continuación, se lista el objetivo general y objetivos específicos del trabajo de titulación; así como también la justificación de realización de este trabajo.

Objetivo general

Desarrollar un algoritmo de optimización híbrido metaheurístico para encontrar una solución óptima al problema de distribución de planta en las MIPYMES textiles del Ecuador.

Objetivos específicos

1. Identificar los algoritmos metaheurísticos y otros métodos que son aplicados al problema FLP dinámico realizando una revisión sistemática de la literatura con el fin de obtener los cinco algoritmos más utilizados frecuentemente.
2. Analizar los algoritmos elegidos haciendo uso de una multi-metodología con el fin de elegir la metaheurística base y los componentes que conforman el algoritmo híbrido metaheurístico.
3. Desarrollar el algoritmo híbrido metaheurístico utilizando el lenguaje de programación C++ y evaluarlo aplicando métodos estadísticos adecuados con la finalidad de verificar si existe superioridad en comparación con los cinco algoritmos frecuentemente aplicados a FLP.

Justificación

Debido a la crisis sanitaria del COVID 19 en el 2020, varios sectores empresariales tuvieron que detener sus actividades, lo que se esperaba que la recuperación de sus niveles de producción y ventas sean lentas. En un inicio, el reinicio de las actividades se realizó con varias restricciones de aforo y circulación que causaron que las empresas trabajen con un porcentaje de mano de obra reducido y una sub-utilización de sus capacidades de producción (*Servicio Nacional de Gestión de Riesgos y Emergencias*, 2020).

A empresas con recursos limitados, como las micro, pequeñas y medianas empresas (MIPYMES), les resultó complicado el acatar las restricciones impuestas por el gobierno, puesto que inciden negativamente en sus costos, la demanda de productos y la disponibilidad de la mano de obra. Debido a esto, la adecuación en la distribución de planta ayuda a cumplir las restricciones impuestas mientras se garantiza la continuidad de la producción.

Las MIPYMES textiles constituyen un actor clave para la sociedad. Este sector genera empleo a cerca del 42% de la mano de obra nacional, Por ejemplo, en el primer semestre del 2019, las MIPYMES textiles de la provincia de Tungurahua registraron ventas superiores a los $116 millones. En el mismo periodo, en la provincial del Azuay, se registraron ventas por más de $35 millones. Lastimosamente, fue uno de los sectores más afectados durante la pandemia.

En efecto, la suspensión de actividades ha tenido grandes repercusiones en este sector, teniendo una caída en sus ventas del 48% y 37%, en Tungurahua y Azuay, respectivamente

(*SRI*, 2020). Para su reactivación, las MiPyMEs textiles han acatado los protocolos definidos por el gobierno, sin tener la posibilidad de adaptarlos a su contexto, ni de analizar los impactos en sus operaciones o en la calidad de sus productos.

Capítulo II – Marco Teórico

Problemas de Optimización

La optimización trata con problemas de minimización o maximización de funciones con varias variables. Dichas variables están usualmente sujetas a restricciones de igualdad o desigualdad. La optimización juega un rol central en la investigación de operaciones, ciencia de gestión e ingeniería de diseño, por mencionar unas pocas (Sivanandam & Deepa, 2007). Uno de los criterios cruciales que influencian los métodos de solución usados para resolver los problemas de optimización es la estructura del dominio sobre el cual es definido. En donde, si una solución es buscada en un conjunto finito o infinito contable, es un problema de optimización discreta o combinatoria. Y, si una solución es buscada en un dominio como el de los números reales, es un problema de optimización continua (Jongen et al., 2004).

Un problema de optimización puede ser definido formalmente de la siguiente manera (Blum, 2003):

$$P = (S, f) \qquad \textit{Eq. 1}$$

En donde

- P es el problema de optimización a ser resuelto.
- S es el espacio de búsqueda, en donde se encuentran las posibles soluciones del problema.
- f es la función objetivo a ser minimizada o maximizada.

Además, el espacio de búsqueda está definido como:

$$S = \{s = \{(x_1, v_2), \dots, (x_n, v_n)\}\} \mid v_i \in D_i \qquad \textit{Eq. 2}$$

En donde

- s representa una solución candidata.
- El par x, v se refiere a la variable que conforma la solución candidata y la dimensión a la que pertenece en el dominio D, respectivamente.

Para resolver un problema de optimización se debe encontrar una solución $s^* \in S$ de tal forma que el valor de la función objetivo para un problema de minimización sea $f(s^*) \leq f(s),\ \forall s \in S$ o $f(s^*) \geq f(s),\ \forall s \in S$ para un problema de maximización. Por lo tanto, s^* es llamada la solución óptima global de (S, f) y el conjunto S^*, que es un subconjunto de S, es llamado el conjunto de soluciones óptimas globales.

Problemas de Optimización Combinatoria

Los problemas de optimización combinatoria (COP por sus siglas en inglés) involucran un dominio de soluciones finito o infinito contable, en donde cada solución puede ser formada por la combinación de sus variables, y dichas variables son definidas por el problema. Las variables suelen estar limitadas por restricciones propias del problema, lo que hace que aumente la complejidad en la búsqueda de una solución. En el trabajo de Peres y Castelli (2021) se realiza una investigación acerca de los COP, en donde una de sus preguntas de investigación involucra la búsqueda de los conceptos fundamentales de las COP. La Figura 1 muestra una descripción formal de los COP.

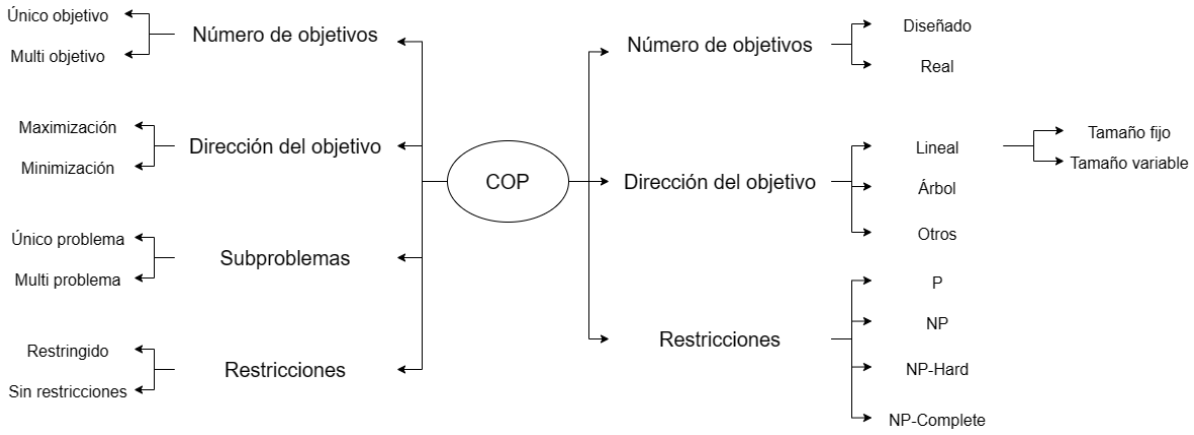


Figura 1. Clasificación de los COP según Peres y Castelli (2021)

En donde los componentes son descritos como:

- **Número de objetivos:** cantidad de mejores valores que retorna la función objetivo, los cuales pueden ser uno o varios.
- **Dirección del objetivo:** si el valor retornado por la función objetivo tiene que ser el más bajo, se trata de un problema de minimización. Si el valor retornado por la función objetivo tiene que ser el más alto, se trata de un problema de maximización.
- **Subproblemas:** el problema puede ser único o dividido en subproblemas debido a su complejidad. En donde cada subproblema puede ser un COP por sí mismo.
- **Restricciones:** un problema puede contener restricciones o no. Las restricciones son un conjunto de prerrequisitos que determina si una solución es válida o no. Los cuales pueden ser: *soft*, cuando estas restricciones pueden o no pueden ser tomadas en cuenta; o, *hard*, cuando estos requisitos tienen necesariamente que ser cumplidos.

- **Fuente:** se refiere a que los COP pueden ser creados por alguien con propósitos educativos, investigativos, etc., o identificados en el mundo real.
- **Codificación de la solución:** la solución de un COP es un arreglo de elementos, el cual puede ser representado como una secuencia lineal de elementos, un árbol u otro tipo de codificación. Además, en una secuencia lineal de elementos, el tamaño puede ser fijo o variable.
- **Complejidad computacional:** tiempo que le toma a un algoritmo calcular la solución a un COP en función al tamaño de la entrada, el cual puede ser P, NP, NP-complete, y NP Hard (se revisarán estas clases más adelante).

Un aspecto importante de los COP es que pueden ser modelados en varias áreas de investigación, considerando el problema que se enfrenta en el mundo real. En este estudio, se usa el FLP que surge en las MIPYMES del Ecuador como COP a ser resuelto. Sin embargo, para una mayor cobertura del tema con respecto a otras áreas, Schrijver (2005) realizó una interesante investigación acerca del estudio matemático de problemas de seis importantes áreas: asignación, transporte, flujo máximo, árbol más corto, ruta más corta, y el problema del viajante.

Complejidad de los COP

Al hablar de complejidad nos estaremos refiriendo a la complejidad temporal. La complejidad de los problemas de los COP es un tema amplio, complejo y extenso en la teoría de la computación. La complejidad temporal de un problema es el tiempo que le toma a un algoritmo encontrar una solución a una instancia de un problema en función de la cantidad de datos que se le suministra (Ralston et al., 2003). La complejidad puede dividirse en las clases P y NP, en donde los problemas que están en P indican que estos pueden ser resueltos por un algoritmo en un tiempo polinomial. Mientras que, los problemas que están en NP indican que cualquier solución dada a estos puede ser verificado en un tiempo polinomial. Además, cuando los problemas son tan difíciles como los problemas NP más difíciles, se los clasifica como NP-Hard. Y, cuando los problemas son NP como NP-Hard, pertenecen a la clase de problemas NP-Complete; es decir, son problemas en la clase NP y que cualquier problema NP se puede reducir a ellos en tiempo polinomial. De estas clases, surge la pregunta ¿$P = NP$?, la cual es considerada uno de los problemas más importantes en el área de las matemáticas y ciencia teórica de la computación (Sipser, 1992).

Muchos de los COP están en NP-Hard y no en NP, pero su dificultad es comparable a los problemas NP-Hard. Hasta el momento no se ha encontrado un algoritmo eficiente capaz de resolver todos los problemas NP-Hard en tiempo polinomial (Neumann & Witt, 2010). Un

ejemplo de un problema de optimización combinatoria NP-Hard es FLP, que implica seleccionar un conjunto de ubicaciones dado un conjunto de instalaciones (por ejemplo, estaciones de trabajo, máquinas, departamentos), con el fin de minimizar un costo determinado (más adelante se entrará en detalle acerca de FLP).

Para ampliar el conocimiento y obtener una información más detallada, referirse a los siguientes estudios relevantes que tratan acerca de problemas de optimización en otras áreas que están fuera del alcance de este trabajo: Curry y Dagli (2014) y Kovacs (2018).

Problema de Distribución de Planta (FLP)

FLP es un COP que pertenece a la clase NP-Hard (Pérez-Gosende et al., 2021), en donde la solución del problema es el arreglo de sus instalaciones (estaciones de trabajo, máquinas, departamentos) dentro de un espacio dado (planta) sujeto a algunas restricciones, las cuales pueden ser - y no están limitadas a – que las instalaciones respeten una distancia dada, algunas facilidades estén a continuación de otras, las instalaciones no se sobrepongan las unas con las otras, por nombrar algunas. En FLP, una instalación es una entidad que facilita el rendimiento de cualquier trabajo, el cual puede ser una máquina, herramienta, un centro de trabajo, una célula de manufacturación, un departamento, etc. (Drira et al., 2006). FLP es un problema muy importante en la industria, puesto que una distribución de planta efectiva incrementa el rendimiento, la productividad en general y la eficiencia; al mismo tiempo que, reduce hasta el 20% - 50% de los costos operativos totales de la manufacturación de un producto (Hosseini-Nasab et al., 2018). Para muchos FLP, el objetivo común a ser minimizado son los costos del manejo de materiales (MHC por sus siglas en inglés), ya que la reducción de este disminuye los niveles de trabajo en proceso (WIP por sus siglas en inglés), los tiempos de respuesta y la congestión en general (Singh & Sharma, 2006).

FLP es un problema de optimización que abarca diferentes campos de investigación, por lo que un FLP puede considerar ciertas características como una distribución de planta estático, o dinámico, o incluso puede ser que el proceso de creación de un producto involucre cientos de facilidades y que sean dos plantas en lugar de una, etc. Esta variedad de características que pueden existir ha motivado a los investigadores a intentar clasificar FLP. La revisiones de literatura acerca de FLP realizadas por Hosseini-Nasab et al. (2018) y extendida por Pérez-Gosende et al. (2021) colocan a FLP como un problema que está compuesto principalmente por el tipo de problema, enfoque de planificación, características de las facilidades, configuración del sistema de manejo de materiales y los métodos para generar y evaluar alternativas de distribución. Cada característica se describe a continuación.

Tipo de problema: Esto se refiere a si el proceso de toma de decisiones en el FLP es para plantas completamente nuevas (distribución desde cero) o para plantas existentes que requieren ajustes en la distribución (re-distribución).

Enfoque de planificación: Dependiendo de la variabilidad del flujo de materiales durante el horizonte de planificación, el problema puede considerarse estático o dinámico. En un enfoque estático, el flujo de materiales entre los departamentos permanece constante a lo largo del horizonte de planificación (SFLP). Esto puede ejemplificarse en situaciones donde la modificación de la distribución de planta no es necesaria por un largo periodo de tiempo, debido a que el mercado no es muy volátil. Sin embargo, en un enfoque dinámico, el horizonte de planificación se divide en períodos de tiempo discretos, y se diseña una distribución diferente para cada período (DFLP). Un ejemplo de ello sucede en las empresas que tienen diferentes cambios en el mercado, lo que obliga a las empresas a adaptarse al cambio junto con su distribución de planta para cubrir con las necesidades de demanda.

Características de las facilidades: Esto incluye factores como el tamaño y la forma de la planta, el número de departamentos o estaciones de trabajo, el flujo de materiales y la proximidad entre departamentos.

Configuración del sistema de manejo de materiales: Esto se refiere a cómo se transportan los materiales dentro de la planta, incluyendo el uso de transportadores, montacargas, vehículos de guiado automatizado (AGVs) u otro equipo de manipulación.

Métodos para generar y evaluar alternativas de distribución: Esto incluye las técnicas y herramientas utilizadas para generar diferentes alternativas de distribución, como modelos de optimización matemática, conocimiento experto, simulación o software especializado. También incluye los criterios y métricas utilizados para evaluar y comparar el rendimiento de las diferentes distribuciones.

Las revisiones mencionadas anteriormente toman en cuenta varios estudios primarios y obtienen las características descritas listadas arriba. Un trabajo que toma en cuenta aspectos específicos es mostrado en la revisión realizada por Flores-Siguenza et al. (2022), en donde investigan los estudios que han resuelto FLP tomando en cuenta restricciones de resiliencia y seguridad (dicho estudio fue realizado en el proyecto de investigación ResilTEX). Los autores han encontrado que los indicadores comúnmente usados son compromiso más alto de administración, aprendizaje organizacional, flexibilidad organizacional, concientización, cultura justa y preparación para emergencias. Por ejemplo, Macuzić et al. (2016) usan un proceso jerárquico analítico para determinar el peso relativo de las variables de resiliencia.

Aleksić et al. (2013) proponen un modelo basado en la teoría de conjuntos difusos para evaluar el potencial de la resiliencia organizacional. Otros trabajos como el de Ahmadi et al. (2017) tratan las características de las facilidades, específicamente del tamaño y la forma de la planta (múltiples plantas), y revisan características como los enfoques de la formulación y las metodologías de resolución.

Formulación Matemática de FLP

La modelación de FLP comprende la abstracción del problema FLP en términos que puedan ser tratados por un actor que será quien guíe el proceso de búsqueda de la mejor distribución de las facilidades contenidas en la planta de una fábrica. Se menciona un "actor" ya que este puede ser una persona, un software o un modelo matemático (Pérez-Gosende et al., 2021). Una persona puede ser capaz de, a través de su conocimiento, generar diferentes distribuciones usando un enfoque de "prueba y error". Por su parte, un software generará diferentes alternativas basadas en reglas o restricciones previamente establecidas. Sin embargo, existen distintos modelos matemáticos que pueden ser usados para modelar FLP. La revisión de literatura realizada por Kusiak y Heragu (1987) menciona los siguientes modelos: asignación cuadrática, cobertura de conjuntos cuadráticos, programación lineal entera, programación lineal mixta y teoría de grafos. Además, estudios como el de Drira et al. (2006) añaden a redes neuronales como una manera más de modelar FLP. Sin embargo, la revisión más reciente realizada por Pérez-Gosende et al. (2021) ya no toma en cuenta los modelos matemáticos, pues ya han sido cubiertos en las revisiones mencionadas, sino hace hincapié en características más generales, como el tipo de dato, refiriéndose a si las variables son asignadas a valores que son conocidos o difusos, a una demanda cierta o incierta, o a el tipo de distancia considerada entre facilidades.

El presente trabajo se enfoca en el problema de asignación cuadrática, puesto que es el modelo que se usa en el proyecto de investigación ResilTEX. Sin embargo, se anima al lector a leer las revisiones de la literatura mencionadas anteriormente para tener una idea más a detalle de los otros modelos matemáticos.

Problema de Asignación Cuadrática

El problema de asignación cuadrática (QAP, por sus siglas en inglés) es un problema de optimización en el que se busca determinar la mejor asignación de una serie de objetos a un conjunto de ubicaciones, minimizando una función objetivo que tiene términos cuadráticos (Burkard, 1984). Koopmans y Beckmann (1957) fueron los primeros en modelar el problema de ubicar plantas con flujo de material entre facilidades, en donde se tienen dos conjuntos de elementos: uno de origen y otro de destino. Cada elemento del conjunto de origen debe

asignarse a un elemento del conjunto de destino, con el objetivo de encontrar la asignación que minimice el costo total. El costo de asignar un par de elementos se define mediante una matriz de costos, que indica el costo de asignar un elemento de origen a un elemento de destino específico. Además, QAP es resuelto con técnicas de optimización para buscar la asignación óptima que minimice el costo total. Este enfoque implica la formulación de un modelo matemático que representa el problema y la aplicación de algoritmos de optimización para encontrar la solución óptima o aproximada. En donde, algunos de los algoritmos utilizados en el método de la asignación cuadrática incluyen algoritmos genéticos o algoritmos de búsqueda local (Loiola et al., 2007). QAP tiene aplicaciones en diversos campos como la logística, la planificación de la producción, el diseño de circuitos, la ubicación de instalaciones y la asignación de recursos. Su objetivo principal es encontrar asignaciones eficientes que minimicen los costos y optimicen el rendimiento en diferentes escenarios y situaciones (Kusiak & Heragu, 1987).

El modelo matemático de QAP es formulado de la siguiente manera (Koopmans & Beckmann, 1957):

$$Min \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}x_{ij} + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} f_{ik}c_{ij}x_{ij}x_{kl} \qquad \textit{Eq. 3}$$

Sujeto a:

$$\sum_{j=1}^{n} x_{ij} = 1\,, i = 1,2,\dots,n \qquad \textit{Eq. 4}$$

$$\sum_{i=1}^{n} x_{ij} = 1\,, i = 1,2,\dots,n \qquad \textit{Eq. 5}$$

$$x_{ij} \in \{0,1\}, i, = 1,2,\dots,n \qquad \textit{Eq. 6}$$

En donde n hace referencia al número total de plantas/ubicaciones; a_{ij} representa el ingreso neto de la planta de operación i en la ubicación j; f_{ik} es el flujo de materiales desde la planta i a la planta k; c_{ij} representa el costo de transportar unidades de materiales desde la ubicación j a la ubicación l; y, x_{ij} es igual a 1 si la planta i está en la ubicación j, 0 caso contrario.

$$b_{ijkl} = \begin{cases} f_{ik}c_{jl} + a_{ij} & if \quad i = j, j = l \\ f_{ik}c_{jl} & if \quad i \neq k \text{ ó } j \neq 1 \end{cases} \qquad \textit{Eq. 7}$$

Y redefine la ecuación de Koopmans y Beckmann a

$$Min \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} b_{ijkl}x_{ij}x_{kl} \qquad \textit{Eq. 8}$$

La Eq. *3* sujeto a las restricciones de Koopmans y Beckmann sirven para modelar el FLP, además, redefiniendo las siguientes variables:

- a_{ij} costo fijo de ubicar la facilidad i en la ubicación j.
- f_{ik} flujo del material entre la facilidad i y la facilidad k.
- c_{jl} costo por unidad de flujo de material entre la ubicación j y la ubicación l.

Este modelo matemático ha sido ampliamente usado y adaptado a diferentes FLP según las restricciones propias del problema (Silva et al., 2021), siendo el modelo matemático FLP propuesto en ResilTEX una adaptación de este.

Métodos de Solución para FLP

Los métodos de solución aplicados a FLP son enfoques o estrategias utilizadas para encontrar soluciones óptimas o aproximadas al problema. Estos métodos buscan determinar la disposición más efectiva de las facilidades en una planta, considerando diferentes objetivos y restricciones dado la formulación matemática del problema. Dentro de los métodos aplicados para encontrar una solución a FLP están los métodos exactos, heurísticas, metaheurísticas, y metaheurísticas híbridas. Los métodos exactos garantizan encontrar la solución óptima, en donde los tres tipos más usados recientemente son branch and bound, programación dinámica y el sub-gradiente modificado. Sin embargo, estos métodos no son adecuados cuando el tamaño del problema es grande (Zhu et al., 2018). Por otro lado, las heurísticas permiten construir una solución óptima o mejorar una ya existente (Drira et al., 2006). Por ejemplo, CRAFT es un algoritmo de mejora muy popular que usa intercambio entre pares (Singh & Sharma, 2006). No obstante, las heurísticas son dependientes del problema (Abdel-Basset et al., 2018). Por su parte, una metaheurística es una heurística de alto nivel que puede proporcionar una solución aceptable haciendo suposiciones acerca del PO que intenta resolver (Žerovnik, 2015). Los algoritmos metaheurísticos son el enfoque de solución ampliamente aplicados a FLP (Flores-Siguenza et al., 2022), siendo los algoritmos genéticos,

recocido simulado, búsqueda tabú, optimización de colonias de hormigas y optimización de enjambres de partículas los más populares (Pérez-Gosende et al., 2021; Zhu et al., 2018). Además, la hibridación de metaheurísticas trata de combinar las ventajas de cada metaheurística u otro algoritmo, con el fin de realizar mejoras en términos de velocidad y velocidad computacional; a su vez que intenta compensar las debilidades de un algoritmo con otro de manera paralela (Ting et al., 2015). Debido a esto, a continuación, se detallan las metaheurísticas comúnmente utilizadas en FLP.

Metaheurísticas

Las metaheurísticas son estrategias de alto nivel para explorar espacios de búsqueda utilizando diferentes métodos. El espacio de búsqueda puede verse como un conjunto de todas las posibles soluciones de un problema específico. Una metaheurística tiene dos propiedades que definen la forma en que se lleva a cabo la búsqueda, llamadas intensificación y diversificación. La diversificación se refiere a la exploración del espacio de búsqueda, mientras que la intensificación se refiere a la explotación de la experiencia acumulada en la búsqueda (Blum, 2003). La intensificación y la diversificación son propiedades importantes que debe tener una metaheurística, ya que un equilibrio entre ambas permite que la metaheurística identifique rápidamente regiones en el espacio de búsqueda con soluciones de alta calidad, sin perder tiempo en regiones ya exploradas, omitidas o regiones que no proporcionan soluciones de alta calidad. Las metaheurísticas que trabajan en soluciones únicas se llaman Metaheurísticas basadas en Trayectoria, las cuales abarcan metaheurísticas basadas en búsqueda local como TS (Búsqueda Tabú), ILS (Búsqueda Local Iterativa) y VNS (Búsqueda de Entorno Variable). Por otro lado, las Metaheurísticas basadas en Población se centran en la evolución de un conjunto de soluciones en un espacio de búsqueda. Este estudio tiene en cuenta dos metaheurísticas basadas en trayectoria: SA (Recocido Simulado) y TS, y tres metaheurísticas basadas en población: GA (Algoritmos Genéticos), PSO (Optimización por enjambres de partículas) y ACO (Algoritmo de la colonia de hormigas).

Recocido Simulado (SA)

Es una de las metaheurísticas más antiguas y seguramente una de las primeras en mostrar una estrategia explícita para escapar del óptimo local. SA fue presentada como un algoritmo de búsqueda que vincula la optimización y la mecánica estadística (Kirkpatrick et al., 1983). SA está motivada por el proceso de recocido físico de los metales, que consiste en calentar un metal a una alta temperatura para que los átomos en el metal estén en un estado aleatorio, y luego enfriarlo lentamente según un programa específico. El patrón de los átomos define el estado del metal en el proceso de recocido, y esos estados se denominan configuraciones.

SA escapa del mínimo local ya que acepta soluciones peores en comparación con la mejor solución encontrada basada en una función de probabilidad que depende de la temperatura, la cual disminuye a medida que avanza la búsqueda, por lo que la probabilidad de aceptar soluciones peores también disminuye. El proceso de búsqueda combina dos fases: aleatorización y mejora iterativa. Siddique y Adeli (2016) describen el algoritmo SA basado en los siguientes pasos:

1. Elegir aleatoriamente una solución inicial x_i dentro del espacio de solución e inicializar la temperatura T.
2. Calcular el valor de la función objetivo con respecto a la solución x_i.
3. Seleccionar otra solución x_j dentro del vecindario de la solución actual.
4. Calcular el valor de la función objetivo con respecto a la solución x_j.
5. Calcular la diferencia, Δf, del valor de la función objetivo entre la solución actual x_i y la solución elegida x_j.
6. Aceptar la solución x_j si $\Delta f \leq 0$; o, si $\Delta f > 0$ aceptar la solución x_j basado en la función de probabilidad $p(x_j)$.
7. Disminuir la temperatura de acuerdo con el método de enfriamiento $T = g(T, k)$ donde k es el número de iteración actual.
8. Terminar la ejecución en caso de que el criterio de parada se haya cumplido.

Por otro lado, la elección del método de enfriamiento adecuado es crucial para el rendimiento del algoritmo, ya que este se encarga de disminuir la temperatura del sistema hasta llevarlo a la mínima energía. Cuanto más baja sea la temperatura, el algoritmo realiza una búsqueda más fina en el vecindario de la solución mínima ya determinada y encuentra una solución mínima mejor (Du & Swamy, 2016b). Además, Geman y Geman (1984) mostraron que es necesario que el método de enfriamiento utilizado por el algoritmo disminuya la temperatura de manera logarítmica para asegurar la convergencia al mínimo global con probabilidad 1, donde la temperatura inicial debe ser alta. En la práctica, el método de enfriamiento logarítmico es demasiado lento debido a su naturaleza logarítmica. Sin embargo, existen heurísticas de enfriamiento rápido que mejoran la velocidad de convergencia, en donde el método de enfriamiento y la temperatura inicial deben adaptarse a la instancia del problema en particular, ya que el costo de escapar de un mínimo local depende de la estructura del espacio de búsqueda (Blum, 2003).

Por lo tanto, es crucial elegir un método de enfriamiento adecuado para obtener una solución cercana al óptimo global. Aarts et al. (2005) indican que, en la práctica, los ingredientes básicos para aplicar SA a un problema de optimización son: una representación concisa del

problema, un vecindario y un método de enfriamiento. En donde, las dos primeras opciones no tienen reglas generales definidas y están más vinculadas a la experiencia y habilidades del diseñador.

SA ha sido aplicado ampliamente a FLP y ha sido modificado para obtener mejores resultados. Palubeckis (2017) implementó el algoritmo SA con la técnica de múltiples inicios (Multi-start SA). Esto significa que, al inicio del algoritmo se generan aleatoriamente múltiples soluciones iniciales en lugar de una sola, como en el SA estándar. Turgay (2018) utilizó SA para encontrar soluciones subóptimas al problema minimizando múltiples objetivos. Hunagund et al. (2018) utilizaron SA con tres operaciones de movimiento en el vecindario de soluciones: inserción, intercambio y reversión. El SA propuesto logró soluciones de igual o mejor costo en comparación con las mejores soluciones reportadas en la literatura para los problemas de prueba realizados. Zolfi et al. (2023) incorporaron dos buffers de memoria que sirven para almacenar las soluciones generadas; y mecanismos de reinicio que son usados para reiniciar el parámetro de la temperatura en el algoritmo SA estándar. Dado que el proceso de búsqueda dinámica en el SA estándar es una cadena de Markov, donde la solución sucesora se elige considerando únicamente la solución actual, la implementación de la memoria permite utilizar la información de búsqueda para mejorar la calidad del resultado final. Matai (2015) modificó el SA estándar para tener en cuenta objetivos cualitativos y cuantitativos. Y. J. Xiao et al. (2016) utilizaron SA para mejorar la secuencia de ubicación de los departamentos. SA utiliza como solución inicial el resultado de un modelo de programación lineal que identifica las ubicaciones y dimensiones exactas, respetando los tamaños especificados de las instalaciones y cumpliendo con las restricciones de relación de aspecto y forma máxima. Ghadikolaei y Shahanaghi (2013) utilizaron el SA estándar con una estrategia de enfriamiento exponencial y generaron soluciones de vecindario mediante el intercambio entre departamentos. S. Wang et al. (2014) mejoraron el SA a través de cinco operadores que generan el vecindario de la solución actual. Estos operadores son seleccionados y ejecutados de forma aleatoria. Se pueden encontrar más referencias a otras aplicaciones en Siddique y Adeli (2016), Baghel et al. (2012), y Suman y Kumar (2006).

Búsqueda Tabú (TS)

La Búsqueda Tabú (TS, por sus siglas en inglés) es una metaheurística introducida por Fred Glover en 1986, que rápidamente se convirtió en el enfoque más utilizado para problemas de optimización combinatoria (Blum, 2003). El algoritmo de TS emplea una búsqueda local llamada mejora óptima, donde la solución actual se reemplaza por la mejor solución vecina en cada iteración. Además, utiliza una memoria a corto plazo conocida como lista tabú para evitar volver a visitar soluciones previamente exploradas y escapar de los óptimos locales. La

lista tabú sirve como un repositorio de soluciones visitadas recientemente y restringe el vecindario de la solución actual para excluir aquellas en la lista tabú. En cada iteración, el algoritmo identifica la mejor solución, designándola como la solución actual. Esta solución se agrega a la lista tabú, mientras que una de las soluciones anteriores, generalmente siguiendo un orden de Primero en Entrar, Primero en Salir (FIFO, por sus siglas en inglés), se elimina. El algoritmo finaliza cuando se cumple un criterio de parada o cuando todas las soluciones en el vecindario de la solución actual están incluidas en la lista tabú. El tamaño de la lista tabú controla la exploración del espacio de soluciones. Una lista más grande promueve que el proceso de búsqueda se adentre en regiones más amplias al prohibir la revisión de un número considerable de soluciones. Para mitigar la ineficiencia de almacenar soluciones completas en la memoria a corto plazo, el algoritmo puede utilizar atributos de las soluciones, como movimientos o diferencias entre soluciones. Este enfoque mejora la eficiencia, pero introduce pérdida de información, ya que un atributo puede asignarse a múltiples soluciones, excluyendo potencialmente soluciones no visitadas de alta calidad. Para abordar este problema, se utilizan criterios de aspiración que permiten la revocación o cancelación de atributos. El criterio de aspiración más sencillo y comúnmente utilizado es permitir un movimiento solo si resulta en una solución con un valor objetivo superior en comparación con la mejor solución conocida actual (Gendreau & Potvin, 2014). En conclusión, el núcleo del algoritmo TS radica en su componente de memoria a corto plazo, que facilita una exploración más completa para identificar el movimiento óptimo mientras se adhieren a restricciones específicas (Glover, 1990). La descripción del algoritmo TS es la siguiente:

1. Generar una solución inicial s.
2. Inicializar las listas tabúes $TL_1, TL_2, .., TL_r$.
3. Inicializar la variable de iteración $k = 0$.
4. Obtener el conjunto permitido a partir de la solución actual. Cada solución no debe violar las condiciones tabúes o debe satisfacer al menos un criterio de aspiración $AllowedSet\,(s,k) \leftarrow s' \in N(s)$, en donde $N(s)$ representa el vecindario de la solución actual.
5. Elegir la mejor solución del conjunto permitido como la nueva solución actual $s \leftarrow ChooseBestOf\,(AllowedSet(s,k))$.
6. Actualizar las listas tabúes y condiciones de aspiración.
7. Actualizar la variable de iteración $k \leftarrow k + 1$.
8. Si las condiciones de parada se cumplen, finalizar el proceso.

TS ha sido ampliamente utilizado para encontrar soluciones a problemas de FLP o sus variantes, ya sea utilizando el algoritmo TS simple o modificando algunos de sus

componentes. Lenin et al. (2014) utilizaron TS para generar una secuencia final de productos. El vecindario de la solución actual se generó intercambiando las posiciones de los productos en la secuencia inicial de máquinas. Además, si la solución es buena, se reemplaza con la solución actual y se añade a la lista tabú. Yu et al. (2014) utilizaron TS para encontrar la secuencia de instalaciones y utilizaron una regla heurística para determinar el espacio adicional entre cada instalación. Zuo et al. (2019) utilizaron TS para minimizar múltiples objetivos en un problema de FLP surgido en un hospital en China. En este caso, el algoritmo TS generó una solución inicial y la función objetivo se seleccionó aleatoriamente en cada iteración a partir de un conjunto de soluciones objetivo. Además, se utilizó una búsqueda local para construir el conjunto de soluciones vecinas, calcular el valor de cada solución y seleccionar la mejor de ellas. Kothari y Ghosh (2013) presentaron dos implementaciones de la búsqueda tabú. La primera utilizó una búsqueda exhaustiva del vecindario de 2-opt, mientras que la segunda utilizó una búsqueda exhaustiva del vecindario de inserción. La primera consistió en todas las permutaciones que se pueden generar intercambiando las posiciones de dos instalaciones en la permutación, y la segunda consistió en todas las permutaciones que se pueden generar eliminando una instalación de la permutación e insertándola en una posición diferente en la misma permutación. Ambas implementaciones de la búsqueda tabú comenzaron creando un número fijo de permutaciones iniciales, donde cada permutación mantenía su lista tabú. Otros trabajos sobre el uso de TS en FLP se pueden encontrar en Abdinnour-Helm y Hadley (2000), Chiang y Kouvelis (1996), (Singh, 2009), y (Samarghandi y Eshghi (2010). Además, una lista completa sobre el uso de TS en problemas de optimización se puede encontrar en Glover (1990).

Algoritmo Genético (GA)

El Algoritmo Genético (GA) es un algoritmo evolutivo que aprovecha las habilidades de la naturaleza para evolucionar y adaptarse a su entorno. Específicamente, GA simula el proceso de evolución biológica de los cromosomas utilizando operadores como la selección (SE), el cruce (CR) y la mutación (MT). Los cromosomas representan soluciones al problema, y se utiliza una función de aptitud u objetivo para evaluar y seleccionar a los padres. SE es el proceso de seleccionar a los padres que formarán la nueva población, comúnmente realizado a través de métodos como la reproducción proporcional, la selección por torneo, la selección basada en rangos o la selección de estado estable (Beheshti & Shamsuddin, 2013). CR implica mezclar partes de dos cromosomas para intercambiar material genético de los padres. Una técnica utilizada es el cruce en *n* puntos, donde dos cromosomas se dividen en *n* posiciones y sus partes se ensamblan aleatoriamente para generar un nuevo cromosoma. MT altera un cromosoma a través de cambios aleatorios, controlados por una tasa de

mutación que determina el grado de alteración que experimentará el cromosoma. Después de aplicar estos operadores a los cromosomas, es necesario evaluar la capacidad de la nueva población para resolver el problema de optimización a través de la asignación del fenotipo. En algunos casos, el cromosoma en sí mismo puede representar la solución, lo que hace innecesaria la evaluación (Kramer, 2017). El algoritmo GA termina cuando se cumplen ciertos criterios de parada, como el número de generaciones, el tiempo transcurrido o el costo de la función de aptitud/objetivo. La descripción del GA es la siguiente:

1. Inicializar la población.
2. Aplicar cruce.
3. Aplicar mutación.
4. Aplicar mapeo de fenotipos (opcional).
5. Calcular el valor de la función objetivo.
6. Si el número de cromosomas no satisface el tamaño de población, regresar al paso 2.
7. Aplicar la selección de la población de padres.
8. Si las condiciones de parada se cumplen, entonces finalizar el proceso; caso contrario, regresar al paso 2.

Según el trabajo realizado por Hosseini-Nasab et al. (2018), los algoritmos genéticos son comúnmente los más utilizados para resolver problemas de FLP. Por ejemplo, Peng et al. (2018) utilizaron un algoritmo GA adaptativo mejorado con una estrategia de inicialización de la población para reducir el espacio de búsqueda y mejorar la eficiencia. Esta estrategia de uso se divide en front-end y back-end, los cuales ayudan a generar una matriz de posiciones de facilidades para luego combinarlas y formar un cromosoma. Lin y Yingjie (2019) utilizan GA para determinar el diseño óptimo de diferentes sistemas de flujo en una fábrica. Palomo-Romero et al. (2017) utilizan GA paralelos, donde la población se divide aleatoriamente en subpoblaciones y cada una implementa un GA secuencial independiente. García-Hernández et al. (2015) proponen un GA interactivo multiobjetivo que incorpora la experiencia del diseñador en el proceso de búsqueda de soluciones. El diseñador selecciona las mejores soluciones según su criterio y el GA evalúa las soluciones considerando la selección del usuario y el flujo de materiales. El algoritmo termina cuando el usuario está satisfecho con alguna solución en particular. Safarzadeh y Koosha (2017) utilizan el GA clásico con selección tipo "wheel selection" y los operadores de cruce y mutación para parejas de cromosomas y cromosomas individuales, respectivamente. Kalita y Datta (2018) proponen un GA basado en permutaciones con operadores especialmente diseñados para explorar soluciones factibles únicamente. Las soluciones factibles se generan mediante una heurística de inicialización de individuos que consta de tres pasos: ubicación de facilidades de posición fija,

ubicación de facilidades ordenadas y ubicación de facilidades sin restricciones. Rifai et al. (2020) proponen un GA que elige una población inicial aleatoria y la ordena en función de sus valores objetivos. Se seleccionan los cromosomas de mayor valor y se eliminan los de menor valor, formando una población temporal llamada "matching pool" (piscina de apareamiento). En esta población se aplica el operador de cruce de 1 punto para generar la nueva descendencia, y también se aplica el operador de mutación.

El GA se utiliza en muchos otros problemas de optimización donde no es factible realizar una búsqueda completa para encontrar el óptimo global. Los trabajos de Lee (2018), Katoch et al. (2021), Ghaheri et al. (2015), y Slowik y Kwasnicka (2020) son ejemplos de ello.

Optimización de Colonia de Hormigas (ACO)

El algoritmo de Optimización de Colonias de Hormigas (ACO, por sus siglas en inglés) es un procedimiento de búsqueda estocástica inspirado en el comportamiento social de las hormigas. Las hormigas tienen la notable habilidad de encontrar los caminos más cortos entre su nido y las fuentes de alimento. Durante su trayecto, las hormigas depositan feromonas en los caminos que recorren para marcar las rutas favorables que deben ser seguidas por otros miembros de la colonia. Este comportamiento crea una mayor concentración de feromonas en los caminos más cortos, ya que más hormigas llegan al nido a través de estas rutas en comparación con caminos alternativos. El ACO aprovecha esta característica de las colonias de hormigas para resolver problemas de optimización. Su componente clave es un modelo probabilístico parametrizado llamado modelo de feromonas. Se utiliza para generar soluciones al problema ensamblando de manera probabilística un conjunto finito de componentes de solución (Dorigo & Blum, 2005). Por lo tanto, los componentes de selección se refuerzan en función de la calidad de las soluciones, asumiendo que las buenas soluciones consisten implícitamente en buenos componentes de solución. El algoritmo general del ACO consta de tres partes (Blum, 2003): 1) *Construcción de soluciones basada en hormigas* (ABSC, por sus siglas en inglés), que explica cómo las hormigas construyen una solución moviéndose según una política de decisión local estocástica que tiene en cuenta los valores de feromonas y los valores heurísticos de los componentes de solución, 2) *Actualización de feromonas*, que consiste en actualizar las feromonas en la misma ruta inversa basándose en la calidad de la solución después de agregar un componente a la solución parcial actual (comúnmente conocido como actualización de feromonas paso a paso en línea), y 3) *Acciones del Daemon* (opcionales), que se pueden utilizar para implementar acciones centralizadas que las hormigas individuales no pueden realizar, como aplicar búsqueda local a la solución construida por las hormigas. El ACO se puede describir de la siguiente manera:

1. Inicializar todos los valores de las feromonas, inicialmente $p = 0$ y definir la solución actual como vacía $s_{sb} \leftarrow NULL$.
2. Inicializar la variable de iteración $k = 0$.
3. Construir una solución basada en los valores de las feromonas y asignarla a s.
4. Si s es una solución válida, entonces
    a. Aplicar una búsqueda local a la solución s (opcional).
    b. Si $f(s) < f(s_{sb})$ ó $s_{sb} = NULL,\ then\ s_{sb} \leftarrow s$.
    c. Definir $k = k + 1$.
5. Si k alcanza un valor definido, entonces el proceso continuará, sino retornar al paso 3.
6. Aplicar la actualización de feromonas $g(p, k, s_{sb})$.
7. Si los criterios de parada se cumplen, finalizar el proceso, caso contrario retornar al paso 2.

ACO, en comparación con GA o SA, ha sido poco utilizado para FLP. Existen pocos trabajos respaldados por la comunidad científica que hacen uso de ACO. Por ejemplo, Zouein y Kattan (2022) mejoraron ACO utilizando un enfoque de construcción de tres pasos: 1) cada hormiga genera una solución factible, 2) se lleva a cabo un procedimiento de búsqueda local para mejorar la secuencia inicial del diseño de la solución producida por la hormiga, utilizando una heurística voraz de intercambio por pares para generar soluciones con los costos más bajos posibles, y 3) se realiza una actualización de feromonas en el último paso, depositando feromonas en los enlaces que corresponden a la solución con el menor costo global. Liu y Liu (2019) presentaron un algoritmo ACO multiobjetivo en el cual propone una optimización de Pareto basada en la comunicación de feromona local y una búsqueda global basada en la tecnología de nicho para guiar a las hormigas en su elección de ruta. Además, durante la ejecución del algoritmo se utiliza una estrategia de actualización para actualizar el diseño, y se aplica una búsqueda local basada en el método de gradiente adaptativo y un proceso de deformación de departamentos para legalizar la solución. Komarudin y Wong (2010) utilizaron ACO introduciendo nueve tipos de búsqueda local como procedimientos de mejora, los cuales se dividen en dos categorías: búsqueda de vecindario de la forma o estructura del árbol rebanado y búsqueda de vecindario de la representación de la solución de las hormigas. En este algoritmo, todos los procedimientos tienen la misma probabilidad de ser elegidos, ya que el espacio de solución de una instancia del problema puede diferir de otras instancias del problema.

Sin embargo, desde la creación de ACO, este ha sido ampliamente utilizado en otras aplicaciones. Se pueden encontrar referencias a diversas aplicaciones como en el problema

de la mochila, el problema de horarios, detección de imágenes con ACO, por nombrar algunas (Fidanova, 2021), (Tewani, 2017), (Dorigo et al., 2006), (Dorigo, 2004), y (Mirjalili, 2019).

Optimización de Enjambre de Partículas (PSO)

El algoritmo de Optimización de Enjambre de Partículas (PSO, por sus siglas en inglés) está inspirado en el comportamiento colectivo inteligente observado en animales como aves, insectos, manadas y peces. Estos animales trabajan cooperativamente para encontrar comida, adaptando sus patrones de búsqueda en base a sus experiencias de aprendizaje individuales y colectivas. El algoritmo PSO pertenece a la categoría de procesos de búsqueda basados en enjambres, donde cada individuo se denomina partícula. En el contexto de problemas de optimización, las partículas representan soluciones potenciales dentro de un espacio de búsqueda dimensional D. Cada partícula puede mantener información sobre la posición y velocidad óptimas de todo el enjambre, así como de la suya propia, con el objetivo de descubrir la mejor solución global (D. Wang et al., 2018). Al inicio, el algoritmo asigna posiciones aleatorias a un grupo de partículas y luego procede a buscar el valor óptimo a través de actualizaciones iterativas. En cada iteración, cada partícula se actualiza en base a dos valores clave: pbest (la mejor solución encontrada por la partícula hasta el momento) y gbest (la mejor solución encontrada por cualquier partícula en toda la población). Kennedy y Eberhart (1995) definen reglas específicas de actualización para la velocidad y posición mostradas en la Eq. *9* y Eq. *10* respectivamente.

$$v_i(t+1) = v_i(t) + cr_1[x_i^*(t) - x_i(t)] + cr_2[x^g(t) - x_i(t)] \qquad \textit{Eq. 9}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \qquad \textit{Eq. 10}$$

donde c representa una constante de aceleración, y r_1, r_2 representan números aleatorios uniformes entre 0 y 1. En la regla de actualización de la velocidad, el primer componente representa la velocidad anterior, el segundo componente representa el aprendizaje cognitivo y el tercer componente representa el aprendizaje social (Du & Swamy, 2016a). Sin embargo, depender únicamente de gbest para el aprendizaje puede hacer que el algoritmo quede atrapado en un óptimo local. Además, si la velocidad se acerca mucho a cero, las partículas pueden detenerse una vez que alcanzan gbest, un fenómeno conocido como estancamiento. Para mitigar estos problemas, Shi y Eberhart (1998) introdujeron una nueva fórmula de actualización de la velocidad con un decaimiento lineal ponderado que se muestra en la Eq. 11

$$v_i(t+1) = \alpha v_i(t) + cr_1[x_i^*(t) - x_i(t)] + cr_2[x^g(t) - x_i(t)] \quad \textit{Eq. 11}$$

Esta modificación tiene como objetivo encontrar un equilibrio entre las capacidades de búsqueda local y global.

Después de actualizar las posiciones y velocidades de cada partícula en el enjambre, todo el proceso descrito se repite hasta que se cumplan ciertos criterios de detención, como alcanzar el tiempo máximo de ejecución, completar el número especificado de iteraciones, alcanzar un valor deseado de la función objetivo, entre otros. La descripción del PSO es la siguiente:

1. Definir $t = 1$ e inicializar cada particula en la población seleccionando aleatoriamente valores para su posición x_i y velocidad v_i.
2. Calcular el valor de la función objetivo para cada partícula i.
3. Si el valor fitness para cada partícula i es más grande que su mejor valor fitness encontrado hasta el momento, entonces modificar $x_i^*(t)$.
4. Determinar la ubicación de la partícula con el fitness más alto y modificar $x^g(t)$ en caso de ser necesario.
5. Para cada partícula i, calcular su velocidad de acuerdo con las ecuaciones de velocidad mostradas anteriormente.
6. Actualizar la ubicación de cada partícula i de acuerdo con la ecuación de posición mostrado anteriormente.
7. Definir $t = t + 1$.
8. Parar si los criterios de parada se cumplen, caso contrario volver al paso 2.

Algunos trabajos con respecto al FLP han hecho uso de PSO para encontrar una solución óptima o cercana a la óptima. C. Guan et al. (2019) propusieron un framework discreto para PSO con el fin de discretizarlo y abordar el problema NP-Hard. El algoritmo propuesto utiliza una estrategia de ubicación y un enfoque de codificación para cada partícula individual. Además, se ejecuta una búsqueda local por cada partícula, la cual utiliza operaciones de intercambio entre dos facilidades en el vecindario de la solución actual. Hu y Yang (2019) utilizaron PSO para encontrar una solución al FLP, inicializando y asignando la posición y velocidad a las partículas del grupo. También buscaban el individuo y el enjambre extremos en el enjambre generado basado en el valor objetivo de cada partícula. Además, después de actualizar las posiciones y velocidades de cada partícula, el algoritmo verificaba que las soluciones cumplieran con las restricciones del problema. En caso de no cumplir, la solución no era elegida. Derakhshan Asl y Wong (2017) hicieron uso de un PSO modificado para resolver FLP estáticos y dinámicos. El algoritmo comienza con la creación de las partículas y

su evaluación, actualizando los valores de velocidad y posición de cada partícula. Se aplican intercambios de departamentos hasta alcanzar un número de iteraciones especificado y se actualizan los valores de posición individual y global a los mejores encontrados. Este procedimiento se realiza en caso de que el problema sea estático o dinámico. Sin embargo, en el problema dinámico se agrega un método de intercambio de períodos y una búsqueda local al procedimiento PSO. Liu et al. (2018) utilizan PSO para optimizar múltiples objetivos, dividiendo el espacio objetivo y aplicando operaciones de mutación heurística y una búsqueda local subsecuente basada en el método del gradiente para las restricciones de no superposición. X. Xiao et al. (2019) hicieron uso de PSO y lo modificaron a través de procesos de mutación y búsqueda local debido a la débil habilidad de búsqueda local que presenta PSO. El proceso de mutación se obtiene de la metaheurística GA, que selecciona aleatoriamente dos partículas e intercambia su posición. El proceso de búsqueda local se realiza en la mejor solución global (gbest) en cada iteración con el objetivo de mejorar la precisión, consistiendo en mover y rotar las instalaciones. Zhou et al. (2020) utilizan el PSO clásico añadiendo pesos de inercia variable con el fin de ajustar la habilidad de búsqueda local y global en el algoritmo para evitar quedar estancado en un óptimo local.

PSO ha sido ampliamente utilizado en otras aplicaciones. Algunas referencias a trabajos acerca de segmentaciones de imágenes con PSO, entrenamientos de redes neuronales artificiales, control de sistemas difusos, entre otras, que podrían interesar al lector son: Zhang et al. (2015), Kulkarni et al. (2015), El-Shorbagy y Hassanien (2018), y Tian y Shi (2018).

Hibridación de Metaheurísticas

La hibridación de metaheurísticas trata con la composición de algoritmos metaheurísticos que no siguen todos los conceptos de una única metaheurística tradicional, sino que combinan varias ideas algorítmicas (Raidl, 2006). Por ejemplo, las metaheurísticas basadas en población y las de trayectoria pueden combinarse para obtener una metaheurística híbrida. Blum et al. (2010) mencionan que la hibridación de este tipo ocurre ya que la fortaleza de las metaheurísticas basadas en población es la capacidad de exploración; mientras que, la fortaleza de las metaheurísticas basados en trayectoria es su capacidad de explotación de regiones de una manera rápida.

La hibridación de metaheurísticas puede darse de tres maneras (Blum, 2003):

Intercambio de componentes entre metaheurísticas: consiste en incluir componentes de una metaheurística en otra. Es considerado uno de los métodos de hibridación más populares, puesto que trata con la combinación de las metaheurísticas de población y trayectoria.

Búsqueda cooperativa: consiste en el uso de varias metaheurísticas que intercambian información acerca de estados, sub-problemas, soluciones, u otros espacios de búsqueda. Por lo general, la búsqueda cooperativa consiste en la ejecución paralela de cada algoritmo.

Integración de metaheurísticas con métodos sistemáticos: consiste en usar las capacidades de las metaheurísticas con métodos aproximados en tres maneras posibles: 1) utilizar la metaheurística y un método exacto de manera secuencial, 2) usar un método exacto para explorar eficientemente el vecindario de una solución, 3) introducir conceptos o estrategias de un algoritmo en otro.

La motivación detrás de tales hibridaciones de diferentes conceptos algorítmicos suele ser obtener algoritmos de mejor rendimiento que exploten y unan las ventajas de las estrategias puras individuales, es decir, se cree que tales híbridos se benefician de la sinergia (Raidl, 2006).

Metaheurísticas híbridas que han demostrado obtener resultados exitosos en FLP son mencionadas en la revisión de la literatura.

Funciones del estado del arte

La introducción de una metaheurística o una metaheurística híbrida es a menudo acompañada de un conjunto de funciones del estado del arte que son usadas para probar la eficiencia del algoritmo (Garden & Engelbrecht, 2014). Las funciones del estado del arte (benchmark functions en inglés) son funciones matemáticas que han sido creadas con el propósito de probar el rendimiento de los algoritmos de optimización, debido a que pueden contener varios mínimos locales y ser un reto de resolver para los algoritmos. No existe una lista estándar de funciones del estado del arte a ser utilizadas, por lo que diferentes estudios utilizan diversas funciones. Las funciones del estado del arte se diferencian por sus propiedades inherentes como: dimensionalidad, modalidad, cuencas, valles, y separabilidad. Hussain et al. (2017) definen estas propiedades de la siguiente manera:

- La dimensionalidad hace referencia al área del espacio de búsqueda, en donde, mientras más grande es la dimensión, más grande será el número de soluciones subóptimas.
- La modalidad define el número de picos en el espacio de búsqueda del problema, estos picos representan los mínimos globales y mínimos locales. En donde se dividen en:
    - Unimodales: tiene un valle y una solución mínima global, ver Figura 3.

  - Multimodales: mantienen muchos mínimos locales y un mínimo global, ver Figura 4.
- Una cuenca es un descenso empinado rodeado de picos altos. Las funciones multimodales contienen muchas cuencas de mínimos locales, ver Figura 4.
- Un valle es un área rodeada por picos y angosto en ancho y largo. Las funciones unimodales y multimodales modelan estos valles de manera diferente y en diferente frecuencia, ver Figura 3.
- La separabilidad define si las variables de optimización pueden ser separadas o no. Si pueden ser separadas, cada variable puede ser optimizada independientemente, si no lo son, significa que todas las variables en una función están fuertemente relacionadas y no puede ser optimizadas independientemente. Un ejemplo de función separable es la función de Rastrigin, ver Figura 4.

Existen muchos trabajos que describen las funciones comúnmente usadas para evaluar el rendimiento de una metaheurística Akojwar y Kshirsagar (2016), Erdemir & Alpaslan Altun (2022), Garden & Engelbrecht (2014). Estos trabajos describen la función matemática, los rangos comúnmente usados y el valor del óptimo global. A continuación, se presentan las funciones: Ackley, Sphere, Rastrigin y Griewank; que han sido elegidas con propósitos académicos. Además, estas funciones presentan diferentes niveles de complejidad y son comúnmente usadas para evaluar metaheurísticas (Akojwar y Kshirsagar, 2016).

Función Ackley

Esta función tiene numerosos mínimos locales y los algoritmos puede caer en estos óptimos locales y fallar. Formalmente, esta función puede ser descrita como se muestra en la Eq. *12*.

$$f(x) = -aexp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right) - exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(cx_i)\right) + a + \exp(1) \qquad \textit{Eq. 12}$$

En donde los valores recomendados para a, b y c son iguales a 20, 0.2 y 2π respectivamente. Esta función es evaluada comúnmente en los rangos $[-32,32]$ en donde la mejor solución 0 es encontrada en $x = [0,0,0,..0]$.

El término exponencial presente en la función enmascara su superficie, lo que da lugar a numerosos mínimos locales. Originalmente, esta función fue diseñada para dos variables; posteriormente, se generalizó para N variables o dimensiones. La Figura 2 muestra la función graficada en tres dimensiones.

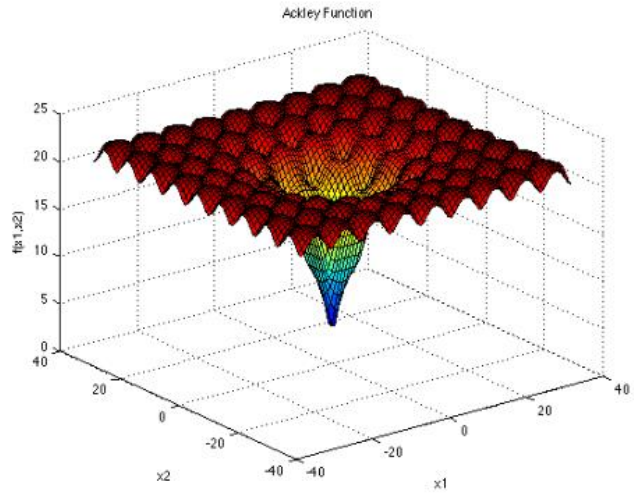


Figura 2. Función Ackley en tres dimensiones

Función Sphere

Es una función continua y unimodal, la cual es considerada muy fácil de resolver. Esta función es evaluada en los rangos $[-5.12,5.12]$ y su solución mínima es 0 encontrada en $x = [0,0,0,..,0]$. La Figura 3 representa la función Sphere en tres dimensiones. Además, la función es expresada formalmente como se muestra en la Eq. *13*.

$$f(x) = \sum_{i=0}^{d} x_i^2 \qquad \textit{Eq. 13}$$

En donde la variable d representa el número de dimensiones que cubrirá la variable x.

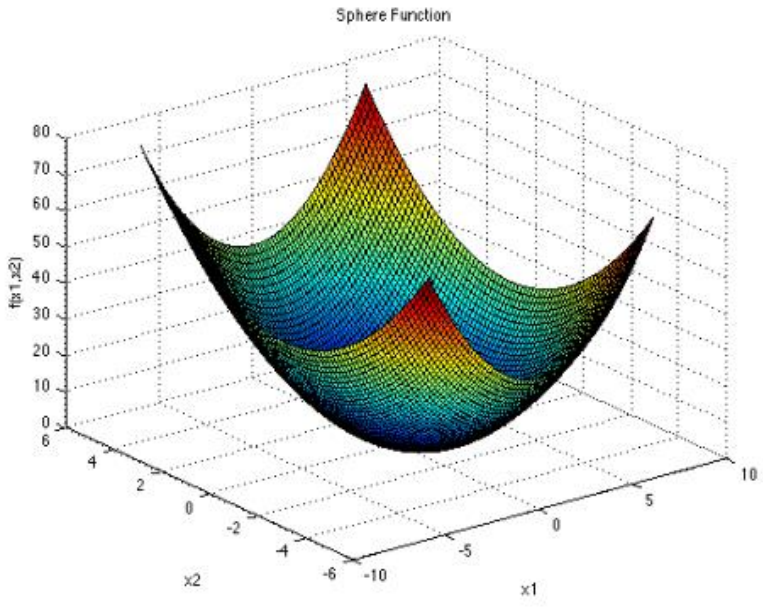


Figura 3. Función Sphere en tres dimensiones

Función Rastrigin

La función de Rastrigin representa una función multimodal y se creó para probar los algoritmos de optimización. Esta función tiene un espacio de búsqueda amplio y un gran número de mínimos locales. El rango de prueba comúnmente es de $[-5.12,5.12]$, en donde su mínimo global es 0 con las variables $x = [0,0,0,..,0]$. La formulación matemática de la función de Rastrigin es expresada en la Eq. *14*.

$$f(x) = 10d + \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i)] \qquad \textit{Eq. 14}$$

En donde d representa el número de dimensiones de la variable x. La Figura 4 muestra la función graficada en un espacio de tres dimensiones.

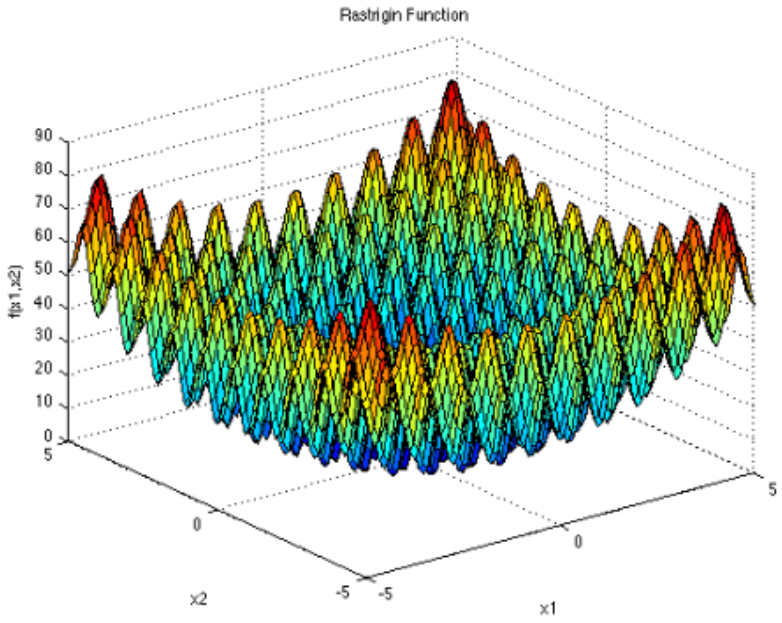


Figura 4. Función de Restrigin en tres dimensiones

Función Griewank

Esta función es usada con propósitos de optimización. El principal objetivo de la función es hacer que las técnicas que optimizan cada variable de manera independiente fallen. Esta función tiene su óptimo global en $x = [0,0,0,..,0]$ y el valor obtenido es 0. Formalmente, la función es expresada según la Ecuación 19.

$$f(x) = \sum_{i=1}^{d} \frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos(\frac{x_i}{\sqrt{i}}) + 1 \qquad \textit{Eq. 15}$$

En donde d representa el número de dimensiones de la variable x. La Figura 5 muestra la función graficada en un espacio de tres dimensiones, la cual muestra que esta puede ser engañosa dado el rango en el que es aplicado.

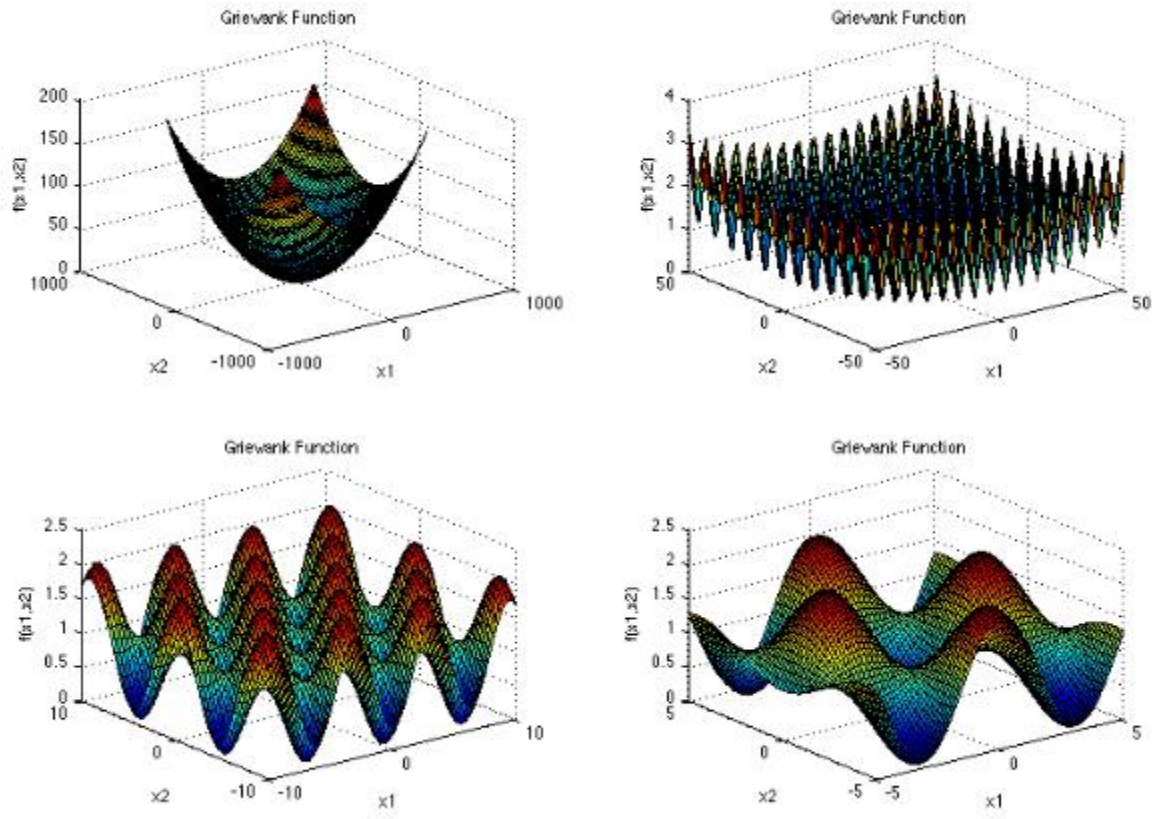


Figura 5. Función de Griewank en tres dimensiones en diferentes rangos

Metodología para la hibridación

Existen diferentes trabajos en los que se ha adoptado el uso de metaheurísticas híbridas para resolver problemas de optimización; sin embargo, estos trabajos predefinen las metaheurísticas a usar sin explicar el porqué de su selección (Sotamba et al., 2023).

En particular, seleccionar una metaheurística para una aplicación específica no es objetivo ni "racional", puede ser un asunto personal o grupal que está sujeto a motivaciones, familiaridades o el entorno de trabajo. Además, el desarrollo, implementación y comparación de metaheurísticas son actividades que involucran técnicas de resolución de problemas, ya que de manera natural se suele realizar algún tipo de proceso para llegar a un determinado objetivo.

Por este motivo, se han desarrollado métodos para tratar con problemas no estructurados, las cuales han sido categorizadas como investigación de operaciones ligeras (Soft Operations Research en inglés, SOR). Sepehrirad et al. (2015) realizó un trabajo en donde investiga el origen de diferentes métodos como: sistemas ligeros, mapeo cognitivo / SODA, enfoque de

elección estratégica, FODA, entre otros. En pocas palabras, cada método define un conjunto de pasos para llegar a un objetivo.

En la literatura, existen pocos trabajos que tratan con el análisis y comparación de metaheurísticas para clasificarlos y elegir las mejores. Por ejemplo, Ferreira (2013) propone un framework para realizar una comparación exhaustiva y coherente de metaheurísticas desde una perspectiva conceptual y experimental, aplicando múltiples metodologías. El framework tiene como base el uso de métodos SOA, el cual fomenta el uso del análisis FODA (fortalezas, oportunidades, debilidades, amenazas) para seleccionar un método adecuado, el Enfoque de Elección Estratégica para tratar la comparación y evaluación de metaheurísticas, y el Pensamiento Convergente/Divergente para guiar, ajustar y asistir en la implementación de metaheurísticas.

Por otro lado, en el trabajo propuesto por De Armas et al. (2022) se genera un framework para descomponer y analizar metaheurísticas, al que llaman Marco de Plantillas de Conjuntos (PTF por sus siglas en inglés). Este framework está conformado por componentes que ayudan a clasificar las metaheurísticas. Además, el framework utiliza un índice de similitud para discernir si una metaheurística es un caso especial, trabaja de la misma manera o es diferente a otra metaheurística. Sin embargo, este marco de referencia no utiliza métodos SOA y ayuda exclusivamente en la clasificación de metaheurísticas ya desarrolladas, con el objetivo de determinar si son novedosas o no.

Por su parte, (Osaba et al., 2018) propone una metodología de diseño, desarrollo, experimentación y despliegue final de los algoritmos metaheurísticos a la solución de problemas del mundo real. La metodología se enfoca en la formulación y el modelamiento del problema, la codificación de la solución y los operadores de búsqueda, la comparación de rendimiento y la replicabilidad, y el despliegue del algoritmo en aplicaciones del mundo real. Si bien esta metodología establece claros puntos a seguir en cada fase, no establece lineamientos claros a seguir cuando se trata de la composición de una metaheurística híbrida.

Por lo tanto, para construir una metaheurística híbrida es necesario llevar a cabo una serie de pasos que incluye la selección de un conjunto de algoritmos, el análisis de su funcionamiento, la combinación y la evaluación de su desempeño. Esto implica el uso de métodos de elección, como los métodos SOA, para tomar decisiones informadas.

En este sentido, en el presente trabajo se emplea la metodología propuesta por Ferreira para desarrollar un nuevo algoritmo híbrido metaheurístico. La adopción de esta metodología nos

permitirá comprender mejor la elección de las metaheurísticas, ofreciendo una visión más clara sobre el rendimiento y limitaciones de la metaheurística híbrida.

Revisión de la literatura

Esta sección tiene como propósito presentar las metaheurísticas híbridas desarrolladas para abordar los problemas de FLP. Nos centraremos en las dos categorías de FLP mencionadas previamente: DFLP y SFLP. En la Tabla 1 se proporcionan detalles sobre las investigaciones revisadas, con el fin de obtener una visión más completa del panorama abordado por los algoritmos híbridos desarrollados en cada estudio. Las características consideradas son las siguientes: 1) Autores: los responsables de cada trabajo, 2) Algoritmos utilizados: las combinaciones de algoritmos que conforman el híbrido propuesto, 3) Área de aplicación: el ámbito al cual se destina la solución, ya sea en el ámbito académico o en la industria, 4) Metodología de hibridación: el nombre o descripción de la metodología empleada para combinar los algoritmos, 5) Tipo de FLP: la categoría específica de FLP a la que pertenece el problema que se intenta resolver, 6) Caso de estudio: cómo se validó el algoritmo, ya sea mediante instancias de la literatura, instancias generadas o casos reales. Es importante señalar que las opciones mencionadas en los puntos 6 y 3 se derivan de los trabajos revisados, en lugar de haber sido establecidas previamente a la revisión.

A continuación, se proporciona un resumen breve del proceso de hibridación de cada algoritmo, recopilado de los trabajos. En este contexto, los términos "algoritmo híbrido" y "metaheurística híbrida" se utilizan de manera intercambiable.

Autores	Algoritmos usados	Área	Metodología de hibridación	Tipo de FLP	Caso de estudio
Uddin 2015	GA, VNS	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura
Pourhassan y Raissi (2019)	GA, PSO	Académico	Ninguna	DFLP	Instancias generadas
Khajemahalle et al. (2021)	NP, SA	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura

Zha et al. (2020)	PSO, SA	Industria	Ninguna	DFLP	Taller de ensamblaje de aeronaves
Bozorgi et al. (2015)	TA, DEA	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura
Turanoğlu y Akkaya (2018)	SA, BFO	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura
Kulturel-Konak y Konak (2015)	SA, MIP	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura
Pourvaziri y Naderi (2014)	GA, SA	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura
Kaveh et al. (2014)	GA, SA	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura
Tayal y Singh (2015)	FA, CSA	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura
Hosseini-Nasab y Emami (2013)	PSO, SA	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura
S. Hosseini et al. (2014)	ICA, VNS, SA	Académico	Ninguna	DFLP	Instancias obtenidas

					de la literatura
Tayal y Singh (2017)	SA, DEA, TOPSIS	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura
S. S. Hosseini et al. (2021)	MGA, CB-SA	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura
Moslemipour et al. (2018)	AC, CS, SA	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura y generadas
Moslemipour (2018)	SA, CS		Ninguna	DFLP	Instancias obtenidas de la literatura y generadas
Pournaderi et al. (2019)	SA	Académico	Ninguna	DFLP	Instancias generadas
Pradeepmon et al. (2018)	PBILA, PWX	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura
Tayal et al. (2020)	DEA, ML	Académico	Ninguna	DFLP	Instancias obtenidas de la literatura
Paes et al. (2017)	GA	Académico	Ninguna	SFLP	Instancias obtenidas de la literatura

García-Hernández et al. (2015)	GA, EK	Académico	Ninguna	SFLP	Instancias obtenidas de la literatura
Wan et al. (2022)	mGRASP, LP	Académico	Ninguna	SFLP	Instancias obtenidas de la literatura
Garcia-Hernandez et al. (2020)	CRO, VNS	Académico	Ninguna	SFLP	Instancias obtenidas de la literatura
Gonçalves y Resende (2015)	BRKGA, LP	Académico	Ninguna	SFLP	Instancias obtenidas de la literatura y generadas
J. Guan y Lin (2016)	VNS, ACO	Académico	Ninguna	SFLP	Instancias obtenidas de la literatura
Ahmadi-Javid y Ardestani-Jaafari (2021)	SA, MA	Académico	Ninguna	SFLP	Instancias obtenidas de la literatura
Liu et al. (2020)	OP, TN	Académico	Ninguna	SFLP	Instancias obtenidas de la literatura
Jerin Leno et al. (2018)	GA, SA	Académico	Ninguna	SFLP	Instancias obtenidas de la literatura

Tabla 1. Descripción tabular de la revisión de la literatura

DFLP

En el trabajo realizado por Uddin (2015) se propone un algoritmo híbrido para resolver DFLP llamado GA-VNS, donde se tomaron en cuenta el Algoritmo Genético (GA) y la Búsqueda de Vecindario Variable (VNS) para componerlo. GA se utiliza para reorganizar el espacio de soluciones y evitar el estancamiento de la búsqueda, mejorando así la exploración del espacio de búsqueda, mientras que VNS se utiliza para explorar más a fondo las regiones prometedoras con la esperanza de encontrar soluciones mejores. El algoritmo híbrido propuesto se compara con algunos algoritmos encontrados en la literatura, donde este ofrece soluciones buenas o incluso mejores en la mayoría de los casos. Pourhassan y Raissi (2019) desarrollaron tres algoritmos híbridos metaheurísticos para resolver el problema de la distribución dinámica de instalaciones considerando múltiples transportadores. Los algoritmos se construyeron basándose en GA y el algoritmo de Optimización por Enjambre de Partículas (PSO). Además, se construyeron dos algoritmos híbridos, PSO1 y PSO2, que difieren entre sí debido a mejoras en la búsqueda local. GA utiliza y propone nuevos operadores de cruce y mutación como componentes de hibridación. Los algoritmos se comparan entre sí, donde los que son basados en PSO tienen los mejores resultados en cuanto a solución y tiempo computacional. Khajemahalle et al. (2021) construyeron un algoritmo híbrido utilizando un algoritmo voraz llamado Particiones Anidadas (NP) y la metaheurística Simulated Annealing (SA). NP se utiliza para obtener las regiones más prometedoras y realizar particiones en cada una de ellas; mientras que, SA se utiliza para obtener el índice más prometedor en cada región circundante. La solución obtenida por el algoritmo NP-SA se compara con las mejores soluciones obtenidas de algunos algoritmos presentes en la literatura. NP-SA obtiene los mejores valores de solución en el 42% de los casos de prueba y también muestra una alta eficiencia en tiempo de ejecución. Zha et al. (2020) proponen un algoritmo híbrido que combina PSO y SA para resolver DFLP con áreas desiguales y centrándose en la demanda aleatoria difusa. Este algoritmo combina SA en ciclos de PSO que mejoran dinámicamente la mejor solución global encontrada en cada iteración de PSO. Además, utilizan dos métodos de intercambio, dos métodos de búsqueda local y un método de cambio para mejorar la calidad de la solución en cada iteración. El algoritmo desarrollado se compara con tres algoritmos obtenidos de la literatura, obteniendo un mejor rendimiento en términos de solución y tiempo de ejecución.

Por otro lado, Bozorgi et al. (2015) generan un algoritmo híbrido para resolver DFLP. En el algoritmo propuesto, primero se genera una ubicación inicial de las instalaciones y luego se crea su vecindario mediante el intercambio por pares y la estrategia inversa. Aplicando el algoritmo de Búsqueda Tabú (TS) mediante una estrategia de diversificación, que incluye memoria basada en frecuencia, función de penalización y tamaño dinámico de la lista tabú

para el modelo DEA, se obtiene el diseño eficiente más deseado. Los experimentos computacionales muestran que el algoritmo propuesto supera a otros algoritmos presentes en la literatura relevante, obteniendo la distribución más eficiente. Turanoğlu y Akkaya (2018) proponen un nuevo algoritmo híbrido metaheurístico llamado SABFO para encontrar una solución óptima para DFLP, que combina los beneficios de la metaheurística SA junto con el algoritmo de Optimización de Búsqueda Bacteriana (BFO). El algoritmo BFO se utiliza para obtener una buena solución inicial y SA se utiliza para mejorar esta solución, ya que es una estrategia de búsqueda local robusta. Los vecinos de la solución inicial se descubren utilizando esta estrategia. El algoritmo se prueba y se compara con otros algoritmos obtenidos de la literatura, generando mejores valores de solución y tiempo de procesador que algunos otros algoritmos. SABFO obtiene la mejor solución en 29 de 48 problemas. Kulturel-Konak y Konak (2015) encontraron una solución para DFLP, considerando períodos de producción repetitivos, donde formaron un algoritmo híbrido utilizando SA para "reparar" las variables binarias del modelo matemático, pasar el resultado a un modelo MIP y generar una solución al problema. El algoritmo se probó con diversos tipos de FLP, mostrando que es eficaz cuando se trata de mejorar la calidad de la solución. Pourvaziri y Naderi (2014) desarrollaron un algoritmo híbrido de multipoblación utilizando GA, que utiliza un procedimiento heurístico para separar el espacio de soluciones en diferentes partes, garantizando una mejor diversificación por parte del algoritmo. Además, utilizaron un mecanismo de búsqueda local basado en SA para mejorar el proceso de búsqueda de soluciones. El algoritmo se probó con casos de prueba conocidos en la literatura y se comparó con 11 algoritmos, el cual muestra superioridad y mejor rendimiento sobre los demás algoritmos. Kaveh et al. (2014) presentaron un algoritmo híbrido de GA y SA para resolver DFLP con restricciones difusas. Al comparar estos dos algoritmos, los resultados mostraron que SA tiene un mejor rendimiento al resolver los casos de prueba.

Además, Tayal y Singh (2015) propusieron un algoritmo híbrido metaheurístico para resolver DFLP con demanda aleatoria. El algoritmo híbrido se compone del Algoritmo de Luciérnaga (FA) y la SA Caótica (CSA). FA se utiliza como un componente para explorar de manera más eficiente el espacio de soluciones y encontrar la solución inicial, mientras que CSA se utiliza como un componente de búsqueda local para mejorar la solución inicial. El algoritmo se compara con tres metaheurísticas, SA, CSA y ACOSA Híbrido, mostrándose más eficiente en términos del valor de la función objetivo. Hosseini-Nasab y Emami (2013) encontraron una solución para DFLP al desarrollar un algoritmo híbrido de metaheurística basado en la metaheurística de PSO y asociarla con SA. PSO se utiliza para explorar el espacio de soluciones y SA se utiliza como un algoritmo de búsqueda local rápido para ser aplicado a la

mejor solución encontrada y mejorarla. La eficiencia del algoritmo se evalúa utilizando problemas de prueba obtenidos de la literatura. Los resultados mostraron que el algoritmo propuesto obtiene la mejor solución en 37 de 48 problemas. S. Hosseini et al. (2014) compusieron un algoritmo híbrido de metaheurística utilizando tres metaheurísticas: el Algoritmo Competitivo Imperialista (ICA), la Búsqueda de Vecindario Variable (VNS) y SA, para resolver eficientemente DFLP. ICA se utiliza como un componente de diversificación utilizando componentes de mutación, intercambio e inserción, mientras que VNS se utiliza como un componente de intensificación, haciendo variaciones en la estructura de vecindario durante el proceso de búsqueda. SA se utiliza para mejorar la solución obtenida y la calidad de esta. Tayal y Singh (2017) proponen un algoritmo híbrido compuesto por SA para generar un conjunto de diseños iniciales, DEA para identificar el subconjunto de diseños eficientes y TOPSIS para clasificar los diseños en función de los criterios y la opinión de un experto en el área. S. S. Hosseini et al. (2021) construyeron un algoritmo híbrido metaheurístico combinando un GA modificado (MGA) y un algoritmo SA basado en la nube (CB-SA) para resolver el modelo DFLP propuesto. MGA utiliza un operador de mutación solo cuando los cromosomas son muy similares, evitando así la convergencia rápida del algoritmo. Por su parte, el algoritmo CB-SA utiliza la teoría de la nube para generar una temperatura de enfriamiento continuo y generar un proceso de enfriamiento más consistente en comparación con el proceso de enfriamiento físico. MGA y CB-SA trabajan juntos para generar la población inicial y seleccionar las soluciones iniciales respectivamente, luego CB-SA inicia un proceso de búsqueda local en las soluciones iniciales y MGA genera nuevas poblaciones aplicando los operadores de cruce y mutación. El algoritmo se compara con las metaheurísticas bases que componen el algoritmo híbrido, donde el algoritmo propuesto obtiene buenos resultados en un tiempo computacional razonable.

Finalizando, Moslemipour et al. (2018) utilizaron las metaheurísticas de Colonias de Hormigas, Selección Clonal y SimulatedAnnealing (AC-CS-SA) para desarrollar un algoritmo híbrido de metaheurística capaz de resolver DFLP en casos deterministas y estocásticos. El algoritmo híbrido propuesto consta de tres etapas: utiliza AC para construir una población de soluciones iniciales factibles, utiliza CS para seleccionar y clonar soluciones, y utiliza SA para mejorar las soluciones clonadas. El algoritmo se compara con SA, donde el algoritmo híbrido, en muchos casos, tiene un mejor rendimiento. Moslemipour (2018) realiza un trabajo similar al anterior utilizando solo CS y SA para generar el algoritmo híbrido. Los algoritmos utilizados para construir el algoritmo híbrido se centran en resolver DFLP con demanda de productos dependiente de variables aleatorias distribuidas normalmente; la diferencia radica en el hecho de que la población inicial se genera de forma aleatoria. Pournaderi et al. (2019) utilizan una

serie de algoritmos para encontrar una solución a DFLP, donde se tienen en cuenta los siguientes algoritmos: Algoritmo Genético de Ordenación No Dominada (NSGA-II), Algoritmo Genético de Rango No Dominado (NRGA) y Algoritmo de Simulated Annealing en la Nube Multiobjetivo. Este último es un algoritmo híbrido que combina SA junto con la teoría de la nube para producir una temperatura de enfriamiento continua. Pradeepmon et al. (2018) proponen resolver DFLP mediante un algoritmo híbrido de metaheurística que utiliza el algoritmo de Aprendizaje Incremental Basado en Probabilidades junto con un Algoritmo de Búsqueda Local de Intercambio por Pares (PBILA-PWX). El algoritmo híbrido se probó con instancias de ejemplo obtenidas de la literatura. Los resultados mostraron que el algoritmo obtiene buenas soluciones cercanas a la óptima en un tiempo muy corto. Tayal et al. (2020) propusieron una metodología híbrida novedosa que complementa el Análisis de Envoltura de Datos (DEA) con Aprendizaje Automático Supervisado y No Supervisado (ML) para medir y predecir la eficiencia de las distribuciones basados en los criterios conflictivos de un DFLP estocástico. La metodología consta de tres etapas: en primer lugar, se identifican los criterios que influyen en el DFLP, los cuales son utilizados en el aprendizaje no supervisado para eliminar criterios correlacionados/redundantes; luego se generaron algunos distribuciones utilizando SA, SA caótico y el algoritmo híbrido compuesto por el Algoritmo de Luciérnaga y SA Caótico, los cuales fueron evaluados con el uso de DEA; se generó un conjunto de datos para entrenar un modelo con varios algoritmos y obtener un modelo final. Como último paso, se realizó la clasificación de los diseños y las comparaciones de los resultados de varios algoritmos. En la última etapa, se predijo la eficiencia de las nuevas distribuciones.

SFLP

Por un lado, Paes et al. (2017) proponen dos enfoques algorítmicos para abordar el SFLP de áreas desiguales. El primero es una implementación del algoritmo genético clásico (GA), y el segundo es un algoritmo híbrido que combina el GA con cuadrantes de restricción (fase 1) y fases de descomposición (fase 2). En la primera fase se desarrolla la ejecución del GA clásico, donde se obtiene un conjunto de soluciones, y en la segunda fase se mejora la calidad de las soluciones obtenidas en la primera etapa. El algoritmo se prueba con ocho instancias clásicas obtenidas de la literatura, 25 nuevas instancias generadas aleatoriamente y 100 funciones del estado del arte, en donde el algoritmo híbrido es significativamente mejor que el GA clásico para problemas de escala mediana y grande. García-Hernández et al. (2015) se centran en el SFLP de áreas desiguales y en la inclusión del conocimiento experto (EK) en el diseño del problema. Proponen un algoritmo híbrido basado en un GA interactivo que se combina con dos métodos de nicho para permitir la interacción entre el algoritmo y el EK. La inclusión de técnicas de nicho en el algoritmo permite preservar la diversidad en las

soluciones, evitando presentar soluciones similares al experto en la misma iteración del algoritmo. El objetivo del algoritmo es satisfacer las preferencias del EK. Wan et al. (2022) presentan un algoritmo híbrido que combina una mejora del procedimiento de búsqueda aleatoria adaptativa codiciosa multiobjetivo (mGRASP) y la programación lineal (LP) para encontrar una solución al SFLP de múltiples filas. Se utiliza mGRASP para optimizar las secuencias de máquinas y obtener un conjunto de secuencias de máquinas no dominadas. Además, se propone un método de dominancia basado en segmentos para medir la proporción de dominancia de cualquier par de secuencias de máquinas. Por otro lado, LP se utiliza para optimizar las distancias de seguridad adicionales entre máquinas adyacentes. Este enfoque se prueba y se compara con un método exacto y dos heurísticas multiobjetivo, donde el algoritmo híbrido muestra eficacia y supera a los enfoques comparativos.

Además, Garcia-Hernandez et al. (2020) utilizaron la metaheurística de Optimización de Arrecifes de Coral (CRO, por sus siglas en inglés) junto con la Búsqueda de Vecindarios Variables (VNS, por sus siglas en inglés) para componer el algoritmo metaheurístico híbrido. VNS se utilizó para intensificar la explotación del espacio de búsqueda con un costo computacional accesible, y se utilizó una nueva representación del problema llamada Estructura de Bahía Flexible Relajada (RFBS, por sus siglas en inglés), la cual permite un mejor manejo del espacio libre en el diseño de la planta. Gonçalves y Resende (2015) desarrollaron un algoritmo metaheurístico híbrido que combina un Algoritmo Genético de Clave Aleatoria Sesgada (BRKGA, por sus siglas en inglés) y un modelo de Programación Lineal (LP, por sus siglas en inglés) para mejorar las soluciones. Este algoritmo está diseñado para encontrar una solución al SFLP de áreas desiguales con requisitos de área predefinidos. BRKGA se utiliza junto con una estrategia de ubicación de instalaciones novedosa y altamente eficiente que ayuda a asignar una instalación a un espacio vacío, mientras que el modelo LP intenta mejorar las soluciones generadas por BRKGA en términos de costo y factibilidad. J. Guan y Lin (2016) generaron un algoritmo híbrido llamado VNSACO, que es la combinación de la metaheurística VNS y la Optimización por Colonia de Hormigas (ACO, por sus siglas en inglés), para encontrar una solución al SFLP de una sola fila. VNSACO aprovecha la intensificación lograda por VNS y la diversificación proporcionada por ACO para equilibrar la explotación local y la exploración global. Además, para mejorar la explotabilidad, se cambian tres estructuras de vecindario diferentes en la búsqueda; y, para acelerar la búsqueda local, se aplica el primer método de mejora a cada búsqueda de vecindario en lugar de aplicarlo a la mejor solución encontrada.

Por último, Ahmadi-Javid y Ardestani-Jaafari (2021) abordan el problema SFLP de áreas desiguales con un algoritmo híbrido basado en SA y un Algoritmo Memético (MA, por sus

siglas en inglés). Además, consideraron la ruta más corta y única en el modelo formulado, que tiene la longitud mínima entre todos los diseños obtenidos en un diseño de bahía. El algoritmo propuesto utiliza MA para determinar los diseños de bahía, y SA se mejora y se utiliza para determinar un bucle único para cada diseño de bahía. Además, los autores incorporan una heurística de construcción de bucles para encontrar una ruta de bucle única para cada diseño (cromosoma). La eficiencia del algoritmo se prueba con cada componente de este. Liu et al. (2020) proponen resolver el SFLP de áreas desiguales utilizando un algoritmo heurístico basado en la optimización de Pareto y la Tecnología de Nicho. La optimización de Pareto se utiliza para encontrar un conjunto de soluciones óptimas de Pareto, que se combina con una búsqueda óptima global basada en tecnología de nicho para mejorar la capacidad de búsqueda de dicho conjunto de soluciones. El algoritmo se prueba con dos conjuntos de instancias obtenidas de la literatura, donde los resultados experimentales demuestran la efectividad del algoritmo. Jerin Leno et al. (2018) desarrollaron un algoritmo metaheurístico híbrido que combina GA y SA para resolver el problema SFLP de áreas desiguales. GA se utiliza como el algoritmo base que genera la población inicial de soluciones, se utiliza una estrategia elitista para mantener soluciones de buena calidad e intercambiarlas con la solución encontrada por el algoritmo en una cierta iteración. Esto se hace para tener en cuenta las buenas soluciones en el proceso de búsqueda; SA se utiliza para mejorar la solución encontrada por el algoritmo.

En la revisión de la literatura, se ha observado que los algoritmos metaheurísticos híbridos desarrollados para abordar el FLP y sus variantes carecen de un enfoque sistemático en su composición. Además, una parte significativa de estos trabajos no valida sus algoritmos metaheurísticos híbridos utilizando escenarios de la vida real ni casos ya existentes en la literatura. Muchos de estos enfoques se centran en la presentación de un algoritmo híbrido diseñado para ser utilizado en instancias obtenidas de la literatura o generadas de manera aleatoria. En este contexto, el presente estudio busca cubrir esta brecha mediante la aplicación de una metodología para la construcción de un algoritmo metaheurístico híbrido. Además, se lleva a cabo la evaluación del algoritmo mediante pruebas que involucran casos tanto de la vida real como de la literatura. De esta manera, se busca aportar a la literatura una aproximación más completa y práctica en el desarrollo y validación de algoritmos metaheurísticos híbridos para el FLP.

Capitulo III – Metodología

Este capítulo describe el caso de estudio y el proceso que se llevó a cabo para componer la metaheurística híbrida, la cual está dividida en dos fases principalmente: 1) una revisión sistemática de los métodos de solución aplicados a FLP y 2) aplicación de una multi-metodología para la creación de la metaheurística híbrida.

Caso de estudio: ResilTEX

El objetivo principal de ResilTEX fue desarrollar un modelo resiliente de distribución de planta para las MiPYMES textiles del Ecuador con un enfoque en productividad y seguridad ocupacional. El modelo generado intenta minimizar el costo total de una distribución de planta tomando en cuenta una cantidad específica de secciones. Además, los aspectos de resiliencia y seguridad que se acoplan al modelo generado son obtenidos en base a la revisión de la literatura y reunión con expertos en el área. Este modelo fue validado tomando en cuenta una empresa textil anónima del Ecuador.

El modelo resiliente se obtiene de Llivisaca et al. (2022) que utiliza un método de asignación cuadrática como base. El modelo es representado por la Eq. *16*.

$$Min\ TC(a_q) = \sum_{i=1}^{q} c_{ia_q} + \sum_{i=1}^{q-1}\sum_{j=1}^{q-1} w_{ij}d(a_i,a_j) + \sum_{j=q+1}^{M}[c_{ja_a} + \sum_{i=1}^{q} w_{ij}d(a_i,a_j)] + 0.5\sum_{i=q+1}^{M-1}\sum_{i>1}^{M-1} w_{ij}d(a_i,a_j) - U_{a_q} \quad \textit{Eq. 16}$$

Sujeto a

$$\sum_{i=1}^{M} x_{ij}\forall j \quad \textit{Eq. 17}$$

$$\sum_{j=1}^{M} x_{ij}\forall i \quad \textit{Eq. 18}$$

En las ecuaciones mencionadas anteriormente, $d(a_i,a_j)$ representa la distancia lineal entre secciones; w_{ij} es una variable de flujo entre la sección i y la sección j; M es el número de secciones, x es una variable binaria (1 si se utiliza una sección en la distribución, de lo contrario 0); c_i representa el costo fijo generado por los espacios vacíos de la distribución de

la sección; C son los costos originados por las soluciones; a, q son variables utilizadas para analizar las secciones q, $q \leq M$; y, U_{a_q} se representa mediante la Eq. *19*. Esta ecuación incorpora la función $s(a_q)$, que devuelve un valor según cuántos indicadores satisfagan la distribución.

$$U_{a_q} = \sum_{i=1}^{q-1} \sum_{i \leq j \leq q}^{q-1} w_{ij} s(a_q) \qquad \textit{Eq. 19}$$

Además, cada termino en la Eq. *16* tiene un propósito, los cuales son descritos en la Tabla 2.

$Min\, TC(a_q)$	Minimizar el costo total de la distribución de planta basado en la solución a con las diferentes secciones q.
$\sum_{i=1}^{q} c_{ia_q} + \sum_{i=1}^{q-1} \sum_{j=1}^{q-1} w_{ij} d(a_i, a_j)$	Este término analiza los costos fijos de cada departamento para generar una decisión basado en los flujos y las distancias lineales.
$\sum_{j=q+1}^{M} [c_{ja_a} + \sum_{i=1}^{q} w_{ij} d(a_i, a_j)]$	Se analizan los costos variables de cada sección que dependen del número de visitas, o flujo de material entre secciones. El número promedio de viajes (w) entre departamentos es usado si no hay manera de acceder al costo.
$0.5 \sum_{i=q+1}^{M-1} \sum_{i>1}^{M-1} w_{ij} d(a_i, a_j) - U_{a_q}$	Este término es responsable de analizar el costo por asignación de cada sección en un valor promedio. El valor de 0.5 es usado debido a que los valores de la matriz de distancia son sumados dos veces.
$U_{a_q} = \sum_{i=1}^{q-1} \sum_{i \leq j \leq q}^{q-1} w_{ij} s(a_q)$	Este valor es propuesto como premio/castigo. El costo total se reduce si los indicadores se cumplen y son relevantes, lo cual afecta al costo total directamente.
$\sum_{i=1}^{M} x_{ij} \forall j$	Este término autoriza secciones de origen a destino y evita repeticiones.

$\sum_{j=1}^{M} x_{ij} \forall i$	Este término autoriza secciones de origen a destino y evita repeticiones.

Tabla 2. Descripción de los términos del modelo resiliente

La Figura 6 muestra la distribución actual de la planta del caso de estudio. Las áreas negras hacen referencia a el área restringida que ninguna instalación debe ser colocada. Las medidas están en metros.

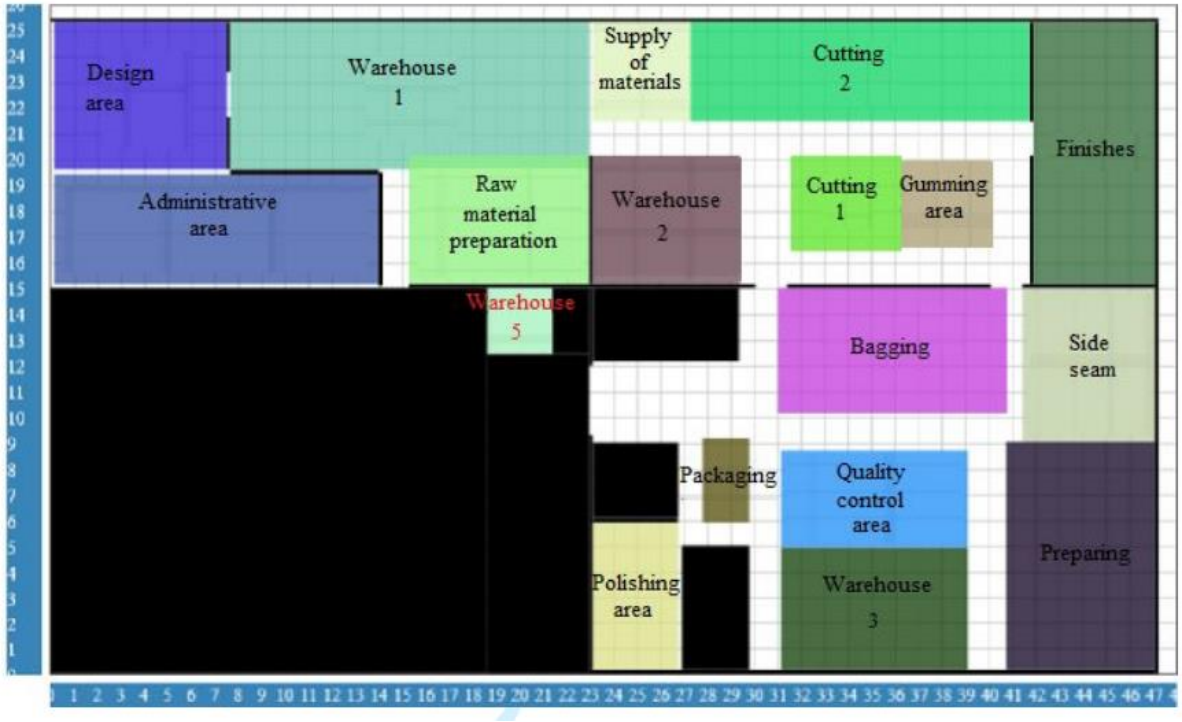


Figura 6. Distribución de instalaciones del caso de estudio de Llivisaca et al. (2022).

Mayores detalles del modelo se ofrecen en el artículo de Llivisaca et al. (2022). Además, se puede acceder a los estudios de Vázquez et al. (2022), para encontrar detalles acerca de las tecnologías aplicadas a FLP considerando factores de resiliencia, Flores-Siguenza et al. (2022a) para profundizar el diseño de distribución dinámico de las MIPYMES acopladas al contexto ecuatoriano, y Flores-Siguenza et al. (2022b), para conocer los factores de resiliencia y problemas de distribución de planta en el contexto industrial.

Revisión Sistemática de los Métodos de Solución Aplicados a FLP

La elección del método de solución adecuado para resolver determinado FLP depende de muchos factores considerando el campo en el que sea acoplado. FLP puede volverse un problema muy complejo de resolver a medida que aumenta el número de restricciones que tiene que cumplir una distribución de planta. Debido a esto, varios autores que han tratado

con este problema han contribuido con soluciones aceptables en tiempos de cálculo realistas utilizando diversos métodos de solución (Pérez-Gosende et al., 2021). Esto nos motiva a revisar la literatura acerca de trabajos recientemente desarrollados. Por esta razón, una revisión sistemática de la literatura fue llevada a cabo en este trabajo. Debido a que una revisión de este tipo abarca un estudio extenso y detallado, una investigación fue desarrollada (Sotamba et al., 2023). Para una lectura más detallada, referirse a los Anexos A y B.

La guía de informes más común para revisiones sistemáticas es el Conjunto de Elementos para Informar Revisiones Sistemáticas y Meta-Análisis Preferidos (PRISMA, por sus siglas en inglés), que contiene una lista de verificación de 27 elementos para informar en revisiones sistemáticas (Gunnell et al., 2022). Esta metodología fue creada como una guía que describe en detalle el proceso de preparación y mantenimiento de revisiones sistemáticas sobre los efectos de las intervenciones sanitarias (Fleming et al., 2014). Dado que la revisión sistemática se llevó a cabo en el campo de la ingeniería, se seleccionó la metodología de Kitchenham (2004), ya que intenta adaptar las pautas médicas a las necesidades de los investigadores en ingeniería. Esta consta de tres etapas: Planificación, Realización e Informe de la revisión. La etapa de planificación está asociada con el desarrollo de un protocolo de revisión; la etapa de realización se ocupa de aplicar el protocolo de revisión desde la etapa de planificación; y, la etapa de informe está relacionada con la publicación del documento.

Las preguntas de investigación a responder fueron:

1. ¿Qué metodologías son usadas para encontrar una solución óptima al problema DFLP?
2. ¿Cómo se evalúa el rendimiento de la metodología propuesta en las metodologías de solución utilizadas para resolver DFLP?
3. Cuando un algoritmo híbrido es desarrollado, ¿Cuál es el proceso de selección llevado a cabo para elegir el conjunto de algoritmos que lo compondrán?

La búsqueda de los artículos siguió una estrategia de búsqueda la cual consistía en armar una cadena de texto utilizando operadores AND y OR, lo cual permitió identificar los estudios relevantes en las bases de datos Scopus, Scielo y Web of Science. La cadena de texto utilizada fue: "*facility layout problem" AND (" optimization" OR" algorithm") AND (" Heuristic" OR "Metaheuristic" OR "Hybrid" OR "deterministic" OR "exact")*. Además, fueron definidos criterios de extracción, que son mostrados en la Tabla 3, con el fin de clasificar los estudios primarios seleccionados y obtener la información necesaria.

Criterio	Nombre

¿Qué metodologías son usadas para encontrar una solución óptima al problema DFLP?	
C01	Metodologías de solución
C02	Nombre del algoritmo
C03	Año
C04	País
¿Cómo se evalúa el rendimiento de la metodología propuesta en las metodologías de solución utilizadas para resolver DFLP?	
C05	Métodos de evaluación
C06	Proceso de validación del algoritmo
C07	Algoritmos seleccionados para la comparación
C08	Métodos utilizados para la selección de parámetros
Cuando un algoritmo híbrido es desarrollado, ¿Cuál es el proceso de selección llevado a cabo para elegir el conjunto de algoritmos que lo compondrán?	
C09	Metodología aplicada para desarrollar el algoritmo híbrido

Tabla 3. Criterios de extracción utilizados para los estudios primarios

El flujo de la metodología seleccionada, basándonos en las preguntas mencionadas previamente, puede ser visto en la Figura 7. Los artículos que fueron considerados en la revisión sistemática pasaron por los siguientes criterios de inclusión – exclusión:

- Criterios de inclusión:
  - Artículos enfocados en DFLP
  - Artículos en español e inglés
  - Artículos de revistas científicas, libros, secciones de libros y conferencias
- Criterios de exclusión:
  - Artículos duplicados
  - Artículos teóricos

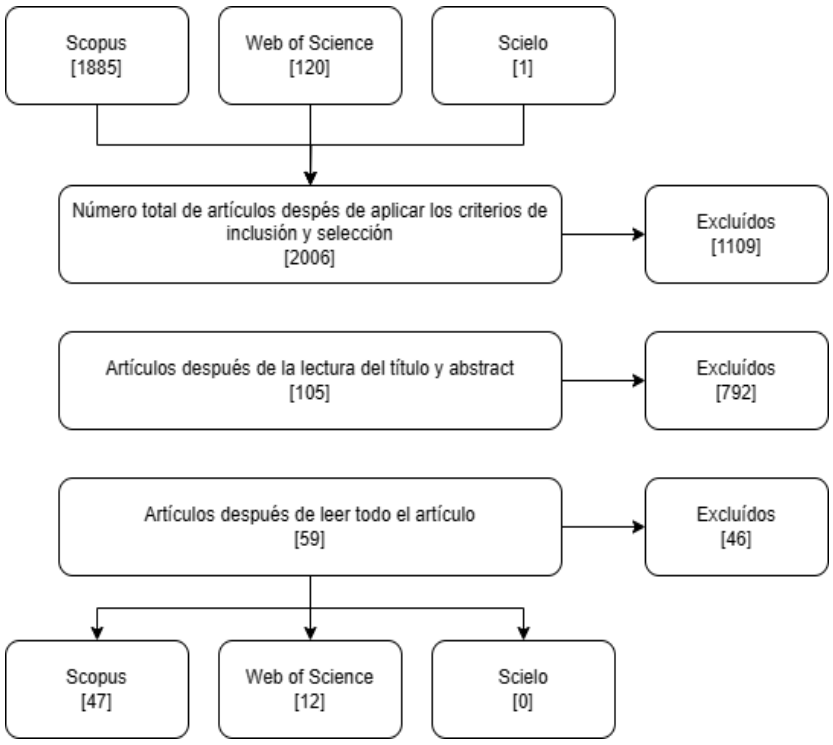


Figura 7. Diagrama de flujo de la metodología aplicada

Los resultados mostraron que los investigadores están más interesados en aplicar algoritmos metaheurísticos y combinarlos para obtener mejores resultados, donde el 66.10% de los estudios utilizan enfoques metaheurísticos (ver Figura 8). En los enfoques metaheurísticos, los tres más usados son: SA (18 ocurrencias), GA (8 ocurrencias) y PSO (4 ocurrencias). Además, los estudios fueron etiquetados en tres categorías para comprender la evaluación de rendimiento: 1) comparación estadística (utilizan algún método estadístico para comparar los algoritmos), 2) comparación cuantitativa (toman en cuenta solo el valor de la función a minimizar/maximizar y el tiempo de ejecución) y 3) sin comparación (No realizan ninguna comparación con otros algoritmos, ni estadística, ni de tiempo de ejecución). En el primer grupo se encuentran el 16.94% de los artículos, en el segundo grupo están el 45.76% de los artículos, y en el tercer grupo están el 37.3% de los artículos. Además, la primera categoría registró algunos métodos estadísticos utilizados para evaluar los algoritmos desarrollados: prueba t de Student, ANOVA, prueba de Tukey, prueba de signos, prueba de Mann-Whitney y prueba Wilcoxon signed-rank. Además, el 11.84% de los estudios utilizan el método Taguchi para ajustar los parámetros de sus algoritmos y obtener los mejores valores. Desafortunadamente, los resultados de este trabajo no mostraron estudios que explicaran el proceso o la metodología aplicada para componer los algoritmos híbridos.

Por lo tanto, debido a que los resultados obtenidos en la revisión sistemática mostraron que las metaheurísticas son los enfoques más usados, siendo SA, GA y PSO las metaheurísticas

líderes, este trabajo toma dichas metaheurísticas, junto con ACO y TS mencionados en la revisión de la literatura de Pérez-Gosende et al. (2021), para ser analizados, implementados y comparados con el fin de componer la metaheurística híbrida que resolverá el FLP de ResilTEX.

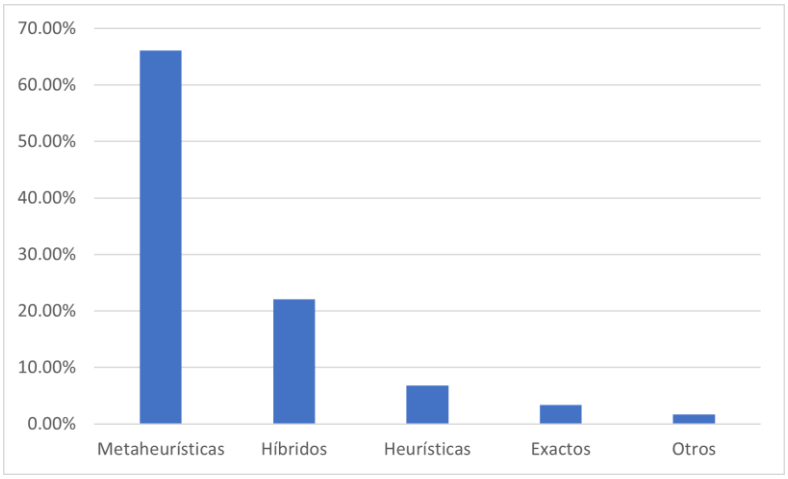


Figura 8. Metodologías de solución

Metodología de Composición de la Metaheurística Híbrida

La metodología propuesta por Ferreira (2013) fomenta el uso de un análisis FODA (Fortalezas, Debilidades, Oportunidades y Amenazas) para seleccionar un método adecuado de un conjunto de metaheurísticas (en nuestro caso son SA, PSO, ACO, TS y GA), utiliza el Enfoque de Elección Estratégica (SCA, por sus siglas en inglés) para manejar la comparación y selección de las metaheurísticas, y promueve el Pensamiento Convergente/Divergente para guiar, ajustar y ayudar en la implementación de la metaheurística híbrida. Estas metodologías juntas, proporcionan una buena guía para elegir las metaheurísticas bases y los componentes que conformarán la metaheurística híbrida.

Análisis FODA

El análisis FODA nos permite identificar las características de las metaheurísticas que se han considerado para componer el algoritmo metaheurístico híbrido. Estas características se recopilaron por el presente autor al momento de la lectura de los estudios que se tomaron en cuenta en la revisión sistemática mencionada anteriormente y del marco teórico presentado en el capítulo anterior. Vale la pena indicar que el análisis FODA es subjetivo y depende de los individuos que participan en la sesión del análisis (Phadermrod et al., 2019). Los resultados del análisis FODA para los algoritmos SA, GA, TS, PSO y ACO se presentan en la Tabla 4, Tabla 5, Tabla 6, Tabla 7 y Tabla 8 respectivamente. Al examinar la literatura en

busca de características FODA, ciertas metaheurísticas no exhiben todas estas características. Sin embargo, esto no implica que estas metaheurísticas carezcan de estas cualidades; más bien, refleja que se adaptan bien al problema FLP.

Fortalezas	**Debilidades**
- Fácil implementación - Uso de una función de probabilidad para escapar del óptimo local - Mejor exploración del espacio de búsqueda al aceptar peores soluciones que la actual cuando la temperatura es alta - Una disminución de la temperatura de forma logarítmica converge la solución al mínimo global	- El método de enfriamiento con disminución de temperatura logarítmica es demasiado lento - No toma en cuenta estados previos cuando se escoge una solución, solamente considera la solución actual
Oportunidades	**Amenazas**

Tabla 4. Análisis FODA de la metaheurística SA

Fortalezas	**Debilidades**
- El proceso de selección puede ser basado en la aleatorización, evitando caer en óptimos locales	- Después de que una nueva población es creada a través de los operadores de mutación y cruce, una evaluación es necesaria para confirmar que los cromosomas generan una solución al problema
Oportunidades	**Amenazas**
- El operador de cruce puede ser extendido a más de dos padres	- El mapeo de fenotipo puede ser difícil cuando es requerido y puede consumir mucho tiempo en algunos problemas - El tamaño de la población es dependiente del problema

Tabla 5. Análisis FODA de la metaheurística GA

Fortalezas	**Debilidades**

- Incorpora memoria para prevenir la repetición de movimientos recientes para evitar la re-selección de soluciones	- La generación de listas candidatas suele llevar a crear una lista dejando soluciones buenas de lado si no se implementa correctamente
Oportunidades	**Amenazas**
- Múltiples listas tabúes pueden ser utilizadas para tener un mejor control de los movimientos ejecutados - El uso de criterios de aceptación permite prevenir la exclusión de buenos movimientos	

Tabla 6. Análisis FODA de la metaheurística TS

Fortalezas	**Debilidades**
- Requiere solamente operadores matemáticos primitivos - Es computacionalmente más eficiente en términos de memoria y velocidad	- Durante la búsqueda, todas las partículas se mueven hacia la región donde $gbest$ está ubicada, el cual lleva a quedar atrapado en óptimos locales
Oportunidades	**Amenazas**

Tabla 7. Análisis FODA de la metaheurística PSO

Fortalezas	**Debilidades**
- La inclusión de información heurística puede mejorar el rendimiento de ACO	- Ciertos componentes de solución, en promedio, reciben actualizaciones más de unas soluciones que de otras
Oportunidades	**Amenazas**
- La solución completa puede ser construida extendiendo una solución parcial	

Tabla 8. Análisis FODA de la metaheurística ACO

Enfoque de Elección Estratégica (SCA)

El SCA es un proceso de seleccionar una opción particular de varias alternativas para lograr un objetivo, que en este contexto ayudará a evaluar y seleccionar el conjunto de metaheurísticas que serán utilizadas para componer una híbrida. Por lo tanto, para lograr una

mejor comprensión de este proceso, esta sección es dividida en tres etapas: representación del problema, evaluación y selección.

Representación del problema

Las metaheurísticas SA, GA, TS, PSO y ACO son algoritmos de propósito general que deben ser acopladas al problema que se intenta resolver al momento de ser implementadas. Esto da lugar a algunas codificaciones que se realizan a las soluciones del problema para que puedan seguir el proceso de búsqueda que describe cada metaheurística. A continuación, se describe el proceso que sigue cada metaheurística para encontrar una solución al problema FLP de las MIPYMES.

SA

Al inicio, una solución inicial es obtenida y el costo total es calculado. La solución inicial es una distribución de instalaciones que respeta las restricciones del problema, tanto de superposición de cada instalación, como del modelo matemático. En cada iteración, se genera una solución vecina moviendo aleatoriamente las posiciones de dos instalaciones de la solución inicial. Para determinar si el intercambio resulta en una nueva solución, es necesario verificar que las instalaciones permanezcan dentro de los límites de la planta y no se superpongan con otras facilidades o áreas restringidas. Después de estas verificaciones, la solución se guarda como la nueva solución si se considera buena o si la probabilidad supera un umbral aleatorio que ronda entre $[0,1]$. El esquema de enfriamiento se basa en una distribución de probabilidad exponencial, que ha mostrado buenos resultados en comparación con otros esquemas de enfriamiento (Tayal & Singh, 2019).

GA

En esta representación usando GA, cada cromosoma representa una instalación y los individuos están compuestos por múltiples cromosomas. Por lo tanto, cada individuo representa una solución candidata al problema, es decir, una distribución de instalaciones. Estos individuos forman una población y se aplican los operadores genéticos: cruce y mutación. El operador de cruce utilizado es el de un solo punto, donde se genera un número aleatorio basado en el tamaño de los cromosomas del individuo, lo cual determina el punto de cruce para combinar dos cromosomas. La selección de padres sigue el método de selección de la ruleta, que asigna probabilidades a los individuos proporcionales a sus valores de aptitud. Durante el cruce, la descendencia se forma tomando los cromosomas del padre desde el punto medio hacia la izquierda y los cromosomas de la madre desde el punto medio hacia la derecha. Es importante tener en cuenta que el punto medio no necesariamente es el punto medio de los cromosomas del individuo, sino más bien un valor aleatorio determinado

en función del número de cromosomas, como se muestra en la Figura 9. El operador de mutación utiliza una tasa de mutación, que típicamente se encuentra en el rango de [0.8, 0.99] según fuentes de la literatura (Kramer, 2017). Para determinar si un individuo sufrirá una mutación, se genera un número aleatorio entre [0,1]. Si este valor aleatorio es mayor que la tasa de mutación, se intercambian aleatoriamente los cromosomas de dos individuos. De lo contrario, el individuo permanece sin cambios. En caso de que el cambio pase, es necesario verificar que el individuo generado cumpla con las restricciones del problema, que no se superponga con otros individuos y que respete las áreas restringidas. Este proceso se aplica a todos los individuos dentro de la población y se repite durante un número especifico de generaciones.

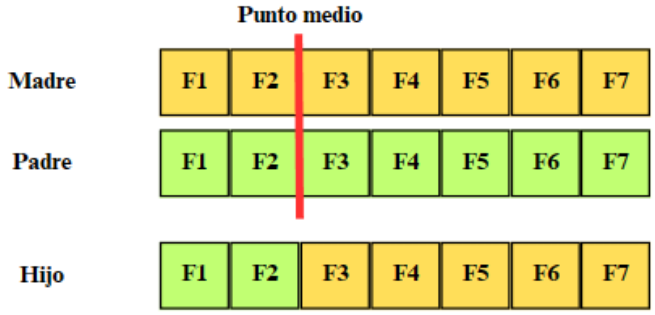


Figura 9. Cruce de los individuos

TS

Inicialmente, un conjunto de soluciones permitidas es generada. Una solución permitida representa una distribución de planta que cumpla con las restricciones de superposición de instalaciones y de áreas restringidas. Durante la generación de soluciones permitidas, se evalúa el criterio de aceptación y se verifica su elegibilidad en la lista tabú. Además, el criterio de aceptación evalúa si la solución tiene un coste menor que la solución actual. Por otro lado, almacenar la solución completa en la lista tabú aumentaría significativamente el tiempo de cálculo y ejecución. Por este motivo, para optimizar la eficiencia, se utilizan los movimientos utilizados para obtener soluciones vecinas como movimientos tabúes. Por ejemplo, si una solución vecina es generada con el intercambio de una instalación a con una b, entonces el movimiento (a, b) y (b, a) es ingresado a la lista tabú. Este enfoque captura eficientemente las características de la solución, lo que resulta en un proceso mucho más rápido. Además, si una solución muestra un buen costo total, se aplica el criterio de aceptación. Si el costo total de la solución es menor que el costo total actual, se almacena la solución en el conjunto permitido, evitando la lista tabú. La lista tabú se actualiza incorporando los movimientos de las mejores soluciones encontradas en cada iteración. Cada movimiento se inserta en la lista

tabú sin tener en cuenta su presencia, lo que permite reemplazos y facilita una mejor exploración del espacio de búsqueda. Evitar este paso restringiría el espacio de búsqueda, aumentando el riesgo de quedar atrapado en un óptimo local. El criterio de aspiración se actualiza reemplazando el costo mínimo previamente identificado con el costo mínimo de la solución actual. Vale la pena señalar que, en la implementación actual, la solución actual no se incluye en el conjunto de movimientos permitidos para la próxima iteración. Esta omisión intencional evita quedar atrapado en un óptimo local y fomenta una mayor diversificación en el proceso de búsqueda.

ACO

En una representación con ACO, cada hormiga construye una solución candidata para el FLP intercambiando aleatoriamente las instalaciones. La selección de los intercambios de facilidades se basa en las probabilidades definidas en la Eq. *20*

$$P_{ihl}^{t} = \frac{[\tau_{ihl}(t)]^{\alpha}[h_{ihl}]^{\beta}}{\sum_{f_{lkt}}[\tau_{ifl}^{\alpha}][H_{ifl}]^{\beta}} \qquad \textit{Eq. 20}$$

que considera factores como los niveles de feromonas, la información heurística, la concentración total de feromonas y la información heurística total. La información heurística representa las relaciones de proximidad entre las facilidades, proporcionando información sobre sus distancias relativas. El FLP se caracteriza además por una matriz de diagrama de relaciones, que guía la ubicación de las facilidades. Esta matriz especifica qué facilidades deben colocarse lo más cerca posible entre sí y cuáles deben mantenerse lo más separadas posible.

Una vez que cada hormiga ha construido una solución, se identifica la mejor solución y se registra el resultado, manteniendo un seguimiento de qué distribución produjo un coste menor. Durante este proceso, cada hormiga construye su propia solución y la compara con la mejor solución encontrada hasta ese momento. Después de que todas las hormigas han construido sus soluciones, tiene lugar la actualización de feromonas. Esta actualización sigue la Eq. *21*

$$\tau_{ihl}(t) \leftarrow \rho\tau_{ihl}(t) + \sum_{k=1}^{m} \delta\tau_{ihl}^{k} \qquad \textit{Eq. 21}$$

que incluye la degradación y la adición de un valor de feromona cuando una instalación ha sido movida, lo que resulta en un costo menor en comparación con la solución actual. La cantidad de feromona añadida durante la actualización depende de la calidad de la solución encontrada por la hormiga. Si la solución es de mayor calidad, se añade una mayor cantidad de feromona. La regla de actualización se aplica una vez que todas las hormigas han completado su proceso de construcción, lo que permite que la calidad colectiva de las soluciones encontradas por las colonias de hormigas se refleje en los niveles de feromonas.

PSO

El problema se representa en PSO de la siguiente manera. En primer lugar, se obtiene la solución inicial al problema, junto con su costo total. Luego, cada partícula genera un conjunto de posiciones y velocidades aleatorias. Cada partícula representa una solución al problema, y las velocidades representan las posiciones del centro de masa de cada facilidad. Dentro del enjambre, cada partícula evalúa su aptitud y realiza un seguimiento de su mejor posición y velocidad alcanzada hasta el momento. El enjambre mantiene un registro de la mejor posición y velocidad encontrada entre todas las partículas hasta la iteración actual. Después, se actualiza la velocidad y posición de cada partícula. Para lograr esto, se calcula la velocidad y se agrega a la posición actual de la partícula. Para asegurar que la nueva posición de cada instalación conduzca a una nueva solución, se verifican las restricciones. Si se viola una restricción, se agregan pequeños valores aleatorios que van desde [-10, 10] a la posición, con el objetivo de encontrar una posición válida donde la instalación pueda encajar. Si no se puede encontrar una posición válida, la partícula conserva su posición anterior. Este proceso se repite para cada partícula.

Evaluación

Debido que para SCA es necesario un conjunto de opciones que ayuden a discriminar a cada metaheurística, Ferreira (2013) propone evaluar metaheurísticas en base a un conjunto de características que permiten seleccionar a las mejores en base a puntajes dados. De esta manera, las metaheurísticas pasan por un proceso de evaluación en donde se verifica qué concepto logran satisfacer en base al problema FLP a ser resuelto y se les asigna un puntaje. Las puntuaciones se representan con los valores 1, 2 o 3, que indican valores bajos, moderados y altos, respectivamente. Las características propuestas son:

- **Búsqueda en el vecindario:** se refiere a la capacidad de la metaheurística para explorar el vecindario con respecto a la solución actual.
- **Capacidad de escapar de óptimos locales:** se refiere a la capacidad que tiene la metaheurística de escapar de óptimos locales cuando se encuentre en uno de ellos.

- **Intensificación / Diversificación:** se refiere a la capacidad del algoritmo en buscar y explorar regiones que contienen soluciones candidatas en el espacio de búsqueda.
- **Simplicidad de implementación:** se refiere a que tan fácil se puede implementar el algoritmo utilizando un lenguaje de programación. Para esto se debe considerar las restricciones a respetar por el algoritmo al momento de buscar una solución.
- **Adaptación al caso de estudio:** se refiere a qué tan bien se adapta la metaheurística al problema que se intenta resolver, en este caso es al FLP de las MIPYMES.

La Tabla 9 presenta las puntuaciones de cada metaheurística, las mismas que fueron asignadas después de realizar la implementación y resolver el FLP de las MIPYMES, ya que de esta manera se da un puntaje realista considerando la complejidad del problema. Vale la pena mencionar que el puntaje colocado es bajo la perspectiva del diseñador al momento de implementar y analizar el comportamiento de las metaheurísticas, el cual se explica a detalle a continuación.

Características/Metaheurística	GA	TS	SA	PSO	ACO
Búsqueda en el vecindario	2	3	3	2	1
Capacidad de escapar de óptimos locales	3	3	3	2	1
Intensificación / Diversificación	3	3	2	2	1
Simplicidad de implementación	1	3	3	1	1
Adaptación al caso de estudio	2	3	3	1	1

Tabla 9. Comparación de las metaheurísticas utilizando la herramienta de Ferreira (2013)

Selección

La selección y comparación de las metaheurísticas es realizado a través del uso de redes de trayectoria de búsqueda (STN, por sus siglas en inglés). Las STNs proporcionaron una representación visual del comportamiento de cada metaheurística durante el proceso de identificación de la mejor solución dentro del espacio de búsqueda. Este enfoque contribuye a obtener una comprensión más profunda del comportamiento de las metaheurísticas basado en sus procesos de búsqueda de la mejor solución. En este estudio, adoptamos un modelo de STN basado en grafos y datos propuesto por Ochoa et al. (2021), que utiliza datos recopilados durante el proceso de búsqueda para mapear el comportamiento del algoritmo. El marco de STN requiere una solución representativa para el algoritmo, que refleje su estado de búsqueda según criterios predefinidos, una ubicación que corresponda a una partición predefinida del espacio de búsqueda, una trayectoria de búsqueda que represente la secuencia en la que se descubren las soluciones representativas, y nodos y aristas que representen las ubicaciones y las conexiones entre ellas. Dado que cada solución de FLP

involucra centros de masa que se definen por valores continuos, nos vemos obligados a dividir el espacio de búsqueda utilizando hipercubos de longitud predefinida. Para calcular los hipercubos, Ochoa et al. (2021) presentan las siguientes ecuaciones para extraer resultados significativos del modelo de STN, donde D representa el número de dimensiones en el problema y X, Y denotan el dominio de cada variable dentro de sus valores mínimos y máximos.

$$(X_{max} - X_{min}) \times D \geq 10^n \quad \textit{Eq. 22}$$

$$(Y_{max} - Y_{min}) \times D \geq 10^n \quad \textit{Eq. 23}$$

Además, la longitud del hipercubo se define como 10^{pf}, donde pf es un parámetro de factor de partición calculado como $pf = n - 2$. En nuestro caso de estudio específico, las variables X, Y tienen dominios diferentes, lo que significa que la longitud del hipercubo puede afectar la granularidad del espacio de búsqueda de manera diferente en cada dimensión. Esta disparidad es inevitable debido a las características específicas de diseño del problema. Debido a esto, el tamaño del hipercubo en nuestro caso de estudio es de 1×1.

Por lo tanto, la codificación de la solución en una representación de cadena para la visualización en el STN se realizó representando cada variable de decisión como una instalación, y el espacio de búsqueda continuo se limitó a todas las posibles soluciones candidatas considerando hipercubos de longitud 1×1. El rango de valores que cada variable (instalación) puede tomar se definió como [0, 48] para la coordenada X y [0, 26] para la coordenada Y. Por lo tanto, los centros de masa de cada instalación se pueden codificar utilizando cuatro caracteres por coordenada, con dos caracteres para la coordenada X y dos caracteres para la coordenada Y. Por ejemplo, si una facilidad tiene un centro de masa en $(14.2, 20.2)$, pertenecería al hipercubo de dos dimensiones 15×21, lo que resultaría en la cadena codificada 1521. Este proceso se aplica a cada variable, y una vez que todas las instalaciones están codificadas, se concatenan para obtener la representación final de la cadena como se muestra en la Figura 10.

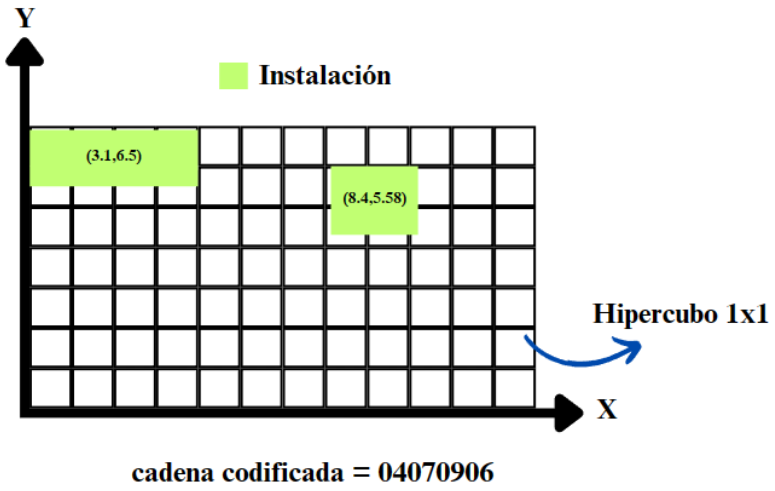


Figura 10. Representación gráfica de hipercubos en dos dimensiones

Con esto en mente, se muestra el comportamiento de las metaheurísticas basadas en población en la Figura 11. Se puede observar que GA demuestra una cobertura superior en la exploración del espacio de búsqueda y encuentra un mayor número de soluciones en comparación con los otros algoritmos, los cuales muestran una exploración relativamente limitada. La Figura 12 muestra el comportamiento de las metaheurísticas basadas en trayectoria. Es evidente que SA muestra un mayor grado de exploración en el espacio de búsqueda, mientras que TS exhibe una exploración reducida pero un enfoque más fuerte en la intensificación para lograr resultados óptimos. Además, en la Figura 11 y Figura 12 se puede apreciar un punto rojo como la mejor solución, la cual es encontrada por los algoritmos PSO y SA. Sin embargo, GA es el algoritmo que se acerca más a la mejor solución encontrada por PSO; mientras que TS está algo alejada de la mejor solución que encuentra SA.

Si nos fijamos a la Tabla 9, la suma de los puntajes obtenidos por los algoritmos muestra a PSO en el puesto cuatro y a SA en el puesto dos. Por su parte, TS, GA y ACO, están en el puesto uno, tres y cinco respectivamente. Además, apoyados con el análisis FODA, que fue el punto de partida para entender qué podemos esperar y mejorar de cada metaheurística antes de que sean implementados y acoplados al caso de estudio; y al ser un enfoque subjetivo, podemos tomar la decisión de utilizar SA, GA y TS para que trabajen en conjunto y compongan la metaheurística híbrida.

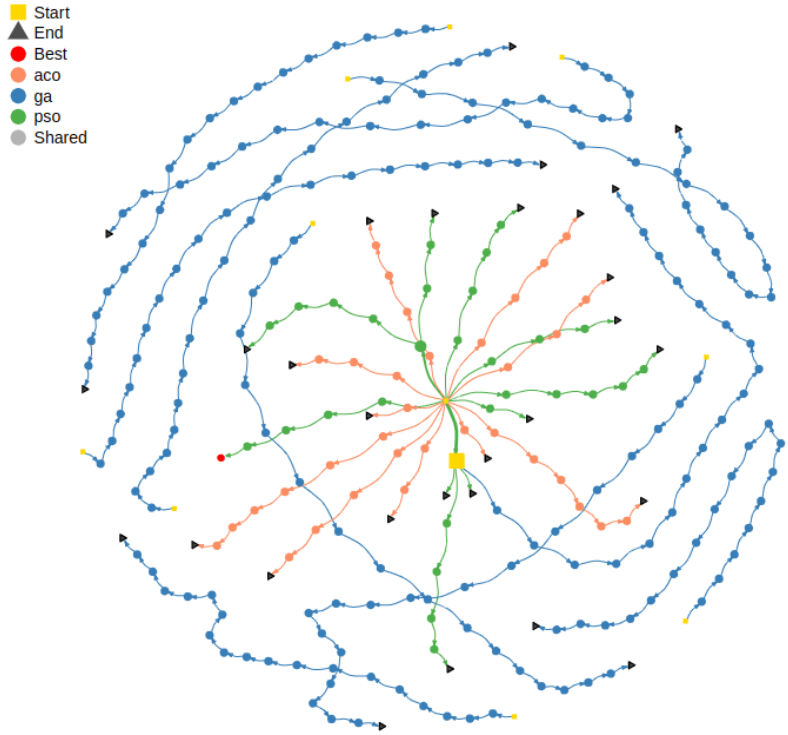


Figura 11. Red de trayectoria de búsqueda de las metaheurísticas basadas en población

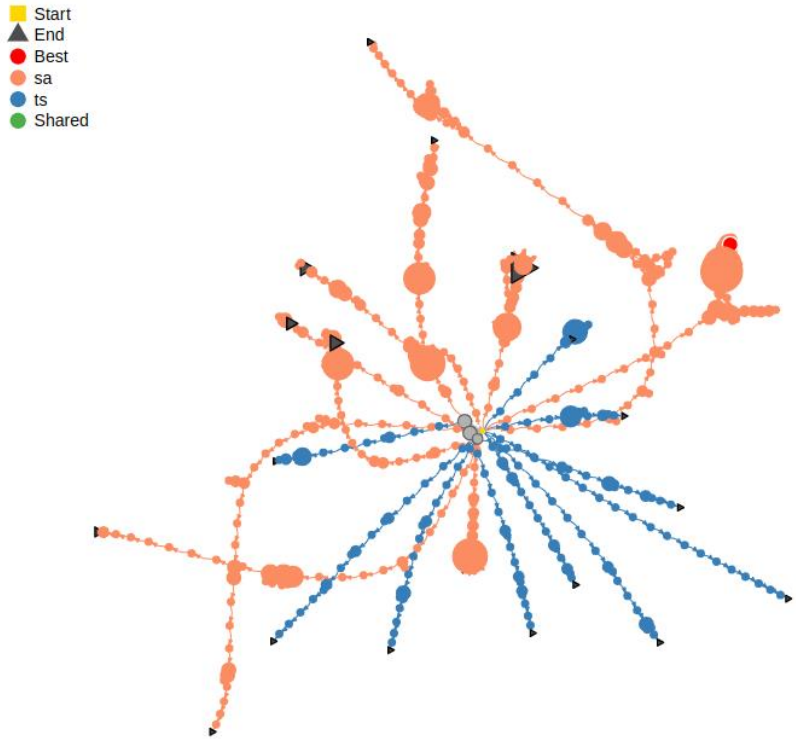


Figura 12. Red de trayectoria de búsqueda de las metaheurísticas basadas en trayectoria

Pensamiento Convergente/Divergente

El pensamiento convergente y el pensamiento divergente son procesos cognitivos asociados con la resolución de problemas y la creatividad, donde el primero está relacionado con la capacidad de analizar y evaluar información de manera lógica para encontrar una solución. Por otra parte, el segundo está relacionado con generar múltiples ideas o soluciones explorando diferentes posibilidades. Debido a que el desarrollo de una metaheurística híbrida es un proceso de toma de decisiones incierto y de prueba, estos procesos cognitivos son adecuados. El pensamiento divergente se utilizó en conjunto con el análisis FODA para generar ideas sobre cómo componer la metaheurística híbrida, teniendo en cuenta sus características claves. Además, a través de la implementación de cada algoritmo, se identificaron diferencias en el desarrollo de cada una, lo que proporcionó una mejor comprensión de sus componentes. Adicionalmente, los gráficos obtenidos con STN ayudaron a analizar el comportamiento de cada metaheurística, facilitando la visualización de la intensificación y diversificación de cada una. Con todo este proceso, se aplicó el pensamiento convergente para excluir y discriminar ciertas metaheurísticas que podrían encajar bien en un algoritmo compuesto, ya que cada uno cubre las debilidades de los demás. Por lo tanto, los algoritmos elegidos para componer el algoritmo híbrido fueron SA, GA y TS. La Figura 13 representa el flujo completo del algoritmo híbrido metaheurístico GENTSA; el nombre es compuesto por los nombres de las metaheurísticas bases: *GENetic-TabuSearch-simulatedAnnealing.*

El proceso de GENTSA comienza con SA, que es utilizado para generar una población inicial que se encarga de almacenar soluciones no tan buenas inicialmente y luego mejorarlas para mantener una población diversa. La selección de soluciones se determina mediante un rango, que se obtiene considerando el número deseado de soluciones en la población inicial y el número de iteraciones alcanzadas cuando la temperatura alcanza un valor mínimo definido. Por ejemplo, suponiendo que se requiere una población inicial de 20 y las temperaturas inicial y final son 100 y 1, respectivamente, a medida que la temperatura disminuye exponencialmente $T_i = T_0\,\alpha^n$ (donde α es la tasa de enfriamiento con un valor de 0.9 y n es el número de iteraciones), el número de iteraciones obtenidas hasta alcanzar la temperatura final es 44, por lo que el rango es igual a 2 ($\lfloor 44/20 \rfloor$), lo que significa que cada dos iteraciones se guarda una solución como individuo en la población inicial. GA se utiliza para mejorar las soluciones en cada generación aplicando operadores de cruzamiento y mutación, como se mencionó anteriormente en la implementación de GA. Después de aplicar estos operadores, se selecciona el mejor individuo de la población y se pasa a TS para la búsqueda local. TS se utiliza para mejorar aún más el mejor individuo obtenido por GA en cada generación y

guardarlo en una piscina de soluciones. Finalmente, cuando se alcanza el número máximo de generaciones, se recorren las mejores soluciones para obtener el mejor individuo general registrado en todo el proceso. En el siguiente capitulo se evalúa y compara el rendimiento de GENTSA con el de las otras metaheurísticas aplicado al problema FLP de las MIPYMES y funciones del estado del arte.

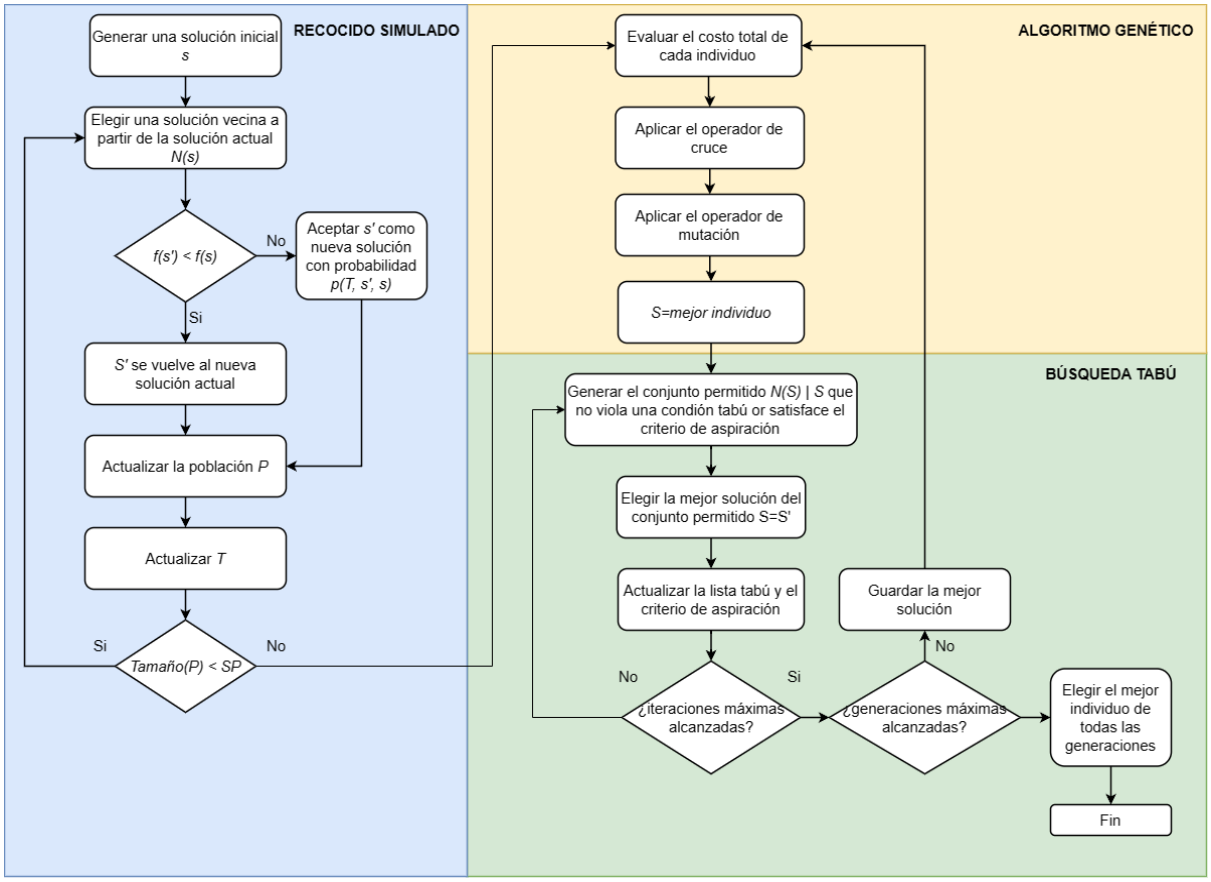


Figura 13. Diagrama de flujo de la metaheurística híbrida GENTSA

Capítulo IV – Evaluación y comparación de rendimiento de las metaheurísticas

La evaluación fue llevada a cabo con las metaheurísticas SA, TS, PSO, GA, ACO y GENTSA, las cuales intentaron encontrar una solución al modelo FLP generado en el proyecto de investigación ResilTEX que se puede observar en la Eq. *16*, sujeto a las restricciones mostradas en las Ecuaciones Eq. *17* y Eq. *18*. Además, GENTSA fue evaluada con funciones del estado del arte y comparada con los algoritmos metaheurísticos HSSJAYA, SSA y JAYA utilizados en el trabajo de Erdemir y Alpaslan Altun (2022), quienes evalúan dichas metaheurísticas con las mismas funciones que se utilizan en este trabajo. GENTSA, junto con las metaheurísticas GA, SA, ACO, PSO y TS, se desarrollaron utilizando el lenguaje de programación C++, ver el Anexo C. Además, los lenguajes de programación Python y R fueron usados para realizar las pruebas estadísticas de los resultados. Los algoritmos se ejecutaron en una computadora HP con un procesador Intel Core i7 8ª generación de 2.20GHz, 12GB de RAM y el sistema operativo Ubuntu 22.04.

Evaluación con el caso de estudio analizado en el proyecto ResilTEX

Para la evaluación, cada algoritmo se ejecutó 10 veces con el fin de eliminar el sesgo de que los resultados obtenidos en una primera ejecución sean aleatorios; además, se usaron los parámetros de configuración mostrados en la Tabla 10. Los valores de los parámetros fueron elegidos a través de un diseño experimental, es decir, se realizó varias ejecuciones de los algoritmos con diferentes parámetros para encontrar los valores más adecuados. La Tabla 11 muestra que GENTSA logró el diseño de menor costo en comparación con los otros algoritmos, con el costo más bajo obtenido en la primera ejecución del algoritmo. GENTSA obtuvo el mejor costo en seis de las diez ejecuciones, con el costo de 306.814u siendo el más bajo. Cabe mencionar que el costo de distribución puede involucrar múltiples factores y variables por lo que no puede ser cuantificado bajo una única unidad de medida. Por lo que el costo de distribución se puede entender como un valor que compara una cierta distribución con respecto del original considerando varias restricciones del modelo. Por otro lado, SA obtuvo los mejores costos en las ejecuciones 4ª y 6ª, ACO en la 7ª ejecución y TS en la 3ª ejecución. La Figura 14 muestra la distribución obtenida con el costo más bajo logrado por GENTSA. Para revisar las distribuciones generadas por las metaheurísticas restantes y en cada ejecución, referirse a los Anexos D-I.

Considerando el tiempo de ejecución de cada metaheurística en segundos, la Tabla 12 presenta la desviación estándar, la media, los valores mínimos y máximos. Se puede observar que SA y TS son las metaheurísticas con tiempos de ejecución más cortos en comparación con los otros algoritmos. Además, SA exhibe la desviación estándar más baja entre ellos.

Parámetro	Valor
SA	
Temperatura inicial	100
Tasa de enfriamiento	0.995
GA	
Tamaño de la población	50
Tasa de cruce	0.8
Tasa de mutación	0.5
Generaciones	20
TS	
Tamaño de la lista tabú	5
Longitud del conjunto permitido	15
ACO	
Alpha	0.6
Beta	0.8
Rho	0.2
Hormigas	20
Q	500
PSO	
Partículas	19
Dimensiones	2
C1	0.3
C2	0.7
Peso de inercia	0.3
Velocidad máxima	10
GENTSA	
Temperatura inicial	50
Tamaño de la población	15
Tasa de enfriamiento	0.98
Temperatura mínima	0.5
Generaciones	20
Tasa de cruce	0.8
Tasa de mutación	0.5
Longitud de la lista tabú	7
Máximas iteraciones tabú	15

Tamaño del conjunto permitido tabú	15

Tabla 10. Configuración de parámetros

Ejecución	GENTSA	SA	GA	PSO	ACO	TS
1	**306.814**	365.389	399.757	371.832	369.022	336.031
2	**306.898**	370.07	416.583	380.732	352.227	347.757
3	340.787	367.882	407.215	382.209	380.959	**331.268**
4	328.925	**324.98**	381.414	394.167	334.51	356.415
5	**311.297**	344.978	350.433	367.234	376.352	333.194
6	325.342	**309.014**	387.759	332.598	345.792	373.743
7	340.571	336.584	395.353	367.66	**333.623**	371.344
8	**311.658**	339.575	340.607	371.712	365.889	324.352
9	**319.84**	358.846	394.703	367.776	357.306	367.119
10	**319.92**	327.122	400.721	346.756	359.862	377.437

Tabla 11. Costo total obtenido por cada metaheurística

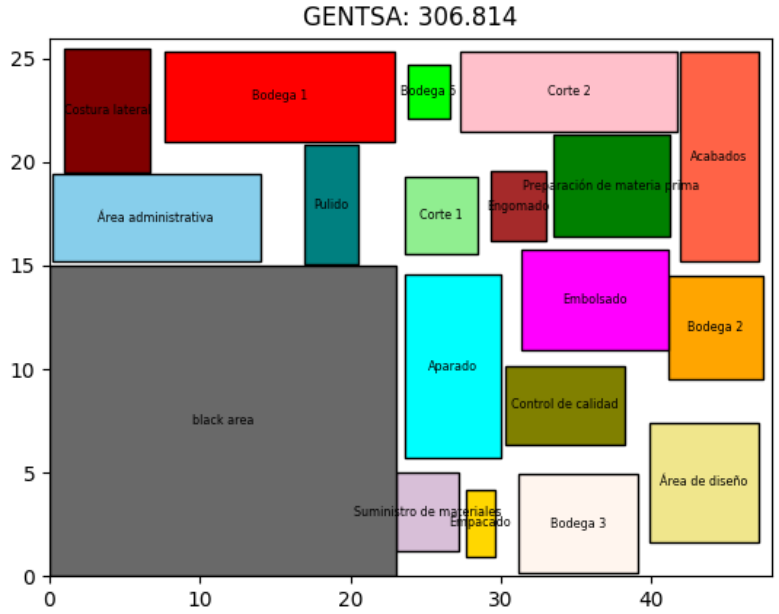


Figura 14. Mejor distribución con el costo más bajo

Metaheurística	Std. Dev	Media	Min	Max
SA	0.43729381173048626	11.1939	10.526	11.987
GA	4.053463676632341	77.621	71.923	83.141

TS	1.254908004596353	12.8561	11.007	15.119
ACO	3.8916196014284616	49.8297	42.805	57.662
PSO	0.5586875095555056	19.9682	19.154	21.241
GENTSA	11.918254199149953	213.3956	199.0	237.515

Tabla 12. Métricas de tiempo de cada metaheurística en segundos

Además, se realizó un análisis estadístico de las metaheurísticas para comparar mejor su rendimiento. Se aplicó la prueba de Shapiro-Wilk (Razali & Wah, 2011) para determinar si la muestra sigue una distribución normal (hipótesis nula) o no (hipótesis alternativa). El valor p obtenido fue $2.3169482965710254e^{-35}$, el cual es menor al nivel de significancia 0.05, lo que proporciona evidencia suficiente para rechazar la hipótesis nula y elegir la alternativa. Con esto en mente, podemos determinar que las pruebas seleccionadas para comparar el rendimiento de una metaheurística con otra deben ser pruebas estadísticas no paramétricas. La comparación de múltiples metaheurísticas se realiza utilizando la prueba de Friedman y la prueba post-hoc de Nemenyi., La prueba de Friedman clasifica las metaheurísticas por separado para cada ejecución y asume la hipótesis nula de que todos las metaheurísticas son equivalentes y, por lo tanto, sus clasificaciones deben ser iguales. El resultado de la prueba de Friedman arroja un valor p de $1.400397912709822e^{-05}$, el cual es menor al nivel de significancia 0.05, lo que proporciona evidencia suficiente para rechazar la hipótesis nula y aceptar la hipótesis alternativa, que establece que todos los metaheurísticos no son equivalentes. La prueba post-hoc de Nemenyi es utilizada para comparar cada metaheurística con las demás y proporcionar un análisis más detallado (Demsar, 2006). En la prueba de Nemenyi, el rendimiento de dos metaheurísticas se considera significativamente diferente si las clasificaciones promedio correspondientes difieren al menos por una diferencia crítica. La Figura 15 ilustra visualmente la comparación de las metaheurísticas utilizando la prueba de Nemenyi, en donde la recta numérica hace referencia al ranking de cada algoritmo, la línea CD representa el valor de la diferencia crítica (en este caso la prueba de Nemenyi arrojó un valor de 2.3842342u), y las líneas negras unen los algoritmos que tienen rendimientos equivalentes. Además, las líneas negras tienen una longitud máxima a la de la diferencia crítica, por lo que los algoritmos que están a una distancia menor son considerados equivalentes.

GENTSA es el algoritmo clasificado en la 1era posición y GA en la 6ta posición, lo que significa que GENTSA en muy superior a GA. Además, GENTSA también supera significativamente a PSO y ACO. Sin embargo, no podemos concluir que GENTSA tenga un rendimiento significativamente mejor que SA y TS. Por otro lado, el gráfico muestra que no hay evidencia suficiente para concluir que SA tiene un mejor rendimiento que ACO, TS que

PSO y ACO que GA, debido a que la distancia entre pares de algoritmos no supera la diferencia crítica (CD).

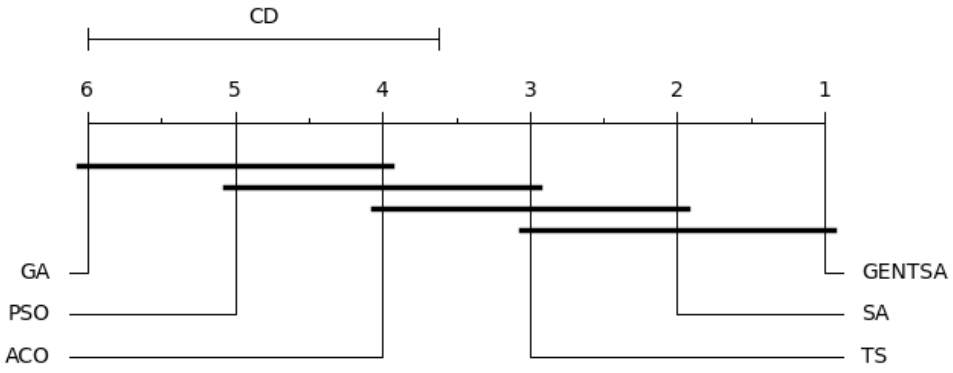


Figura 15. Comparación de rendimiento de las metaheurísticas

Funciones del estado del arte

Se realizó una evaluación del rendimiento de la metaheurística híbrida GENTSA en cuatro funciones del estado del arte: *Ackley, Sphere, Rastrigin y Griewank*. Se consideró un espacio de búsqueda de 20 dimensiones y se aplicó GENTSA para encontrar las soluciones óptimas dentro de los rangos: [-32, 32] para Ackley, [-100, 100] para Sphere, [-5.12, 5.12] para Rastrigin y [-600, 600] para Griewank. Para evaluar el rendimiento de GENTSA, se obtuvo la media y la desviación estándar de los resultados obtenidos en cada función. Esto fue realizado para comparar GENTSA con los algoritmos HSSJAYA, SSA y JAYA. GENTSA se configuró para ejecutarse 30 veces, siendo el mismo número de veces que se ejecutaron los algoritmos de comparación. La Tabla 13 presenta los resultados obtenidos por cada metaheurística. Se puede observar que GENTSA no supera el rendimiento de los otros algoritmos. Estos resultados concuerdan con el hecho de que GENTSA es una metaheurística híbrida diseñado específicamente para resolver problemas de FLP en el contexto ecuatoriano de las MIPYMES textiles que se acoplen al modelo ResilTEX propuesto; y no exhibe un rendimiento superior en las funciones del estado del arte ya que difieren de su dominio previsto. No obstante, cabe mencionar que, basándonos en los resultados, GENTSA presenta un rendimiento competitivo y logra resultados que no están significativamente alejados de los mejores valores obtenidos por las metaheurísticas de comparación. Esta observación sugiere que GENTSA muestra versatilidad y se adapta razonablemente bien a dominios de problemas más allá de su alcance de diseño original.

f_x		HSSJAYA	SSA	JAYA	GENTSA
Sphere	Media	0.00E+00	2.51E-02	2.21E-02	1.53E-10
	Std. dev	0.00E+00	6.69E-02	5.56E-02	2.62E-11
Rastrigin	Media	0.00E+00	3.39E-01	7.99E-01	5.70E+00
	Std. dev	0.00E+00	5.14E-01	1.00E+00	2.58E+00
Griewank	Media	0.00E+00	4.18E-02	4.23E-02	0.09E+00
	Std. dev	0.00E+00	5.52E-02	7.91E-02	0.05E+00
Ackley	Media	0.00E+00	3.43E-01	3.89E-01	1.38E+00
	Std. dev	0.00E+00	4.47E-01	1.00E+00	0.71E+00

Tabla 13. Valores obtenidos por GENTSA y otros al resolver las funciones del estado del arte

Capítulo V – Conclusiones y Recomendaciones

Este trabajo presentó el desarrollo de la metaheurística híbrida GENTSA para resolver el modelo FLP que fue construido tomando en cuenta el contexto de las empresas MIPYMES textiles ecuatorianas; el cual fue probado con un caso de estudio específico. El desarrollo de GENTSA fue llevado a cabo mediante el uso de una metodología de hibridación, la misma que describe el proceso de construcción haciendo uso de metodologías subjetivas. Previo a esto, para la elección del conjunto de metaheurísticas a analizar, una revisión de la literatura fue realizada con el fin de conocer los métodos de solución más comunes para resolver DFLP.

En la revisión sistemática llevada a cabo, se han analizado los algoritmos metaheurísticos híbridos diseñados para abordar el problema FLP y sus diversas variaciones. Se destaca que, en muchos de estos enfoques, la composición de estos algoritmos carece de un enfoque sistemático y no se validan adecuadamente utilizando escenarios del mundo real o casos presentes en la literatura. La mayoría de los trabajos se centran en la presentación de algoritmos híbridos para ser aplicados en instancias obtenidas de la literatura o generadas de manera aleatoria.

En respuesta a esta brecha en la literatura, en este estudio se ha aplicado una metodología para la construcción del algoritmo metaheurístico híbrido. Además, se ha llevado a cabo una evaluación del rendimiento de este utilizando casos tanto reales como extraídos de la literatura. Esto ha permitido una comprensión más profunda y rigurosa de la efectividad de los algoritmos en diversos contextos y situaciones prácticas.

Es importante resaltar que se emprendió una investigación individual para abordar el proceso extenso que conlleva una revisión sistemática de la literatura. A partir de esta base, seleccionamos las metaheurísticas SA, GA y PSO, junto con TS y PSO obtenidas de revisiones literarias previas. El grupo final se compuso de cinco metaheurísticas, en un enfoque que busca aprovechar lo mejor de cada una.

En la aplicación de la multi-metodología, se empleó enfoques variados para la comparación y selección de las mejores metaheurísticas. Desde el análisis FODA hasta el Pensamiento Convergente y Divergente, se ha trabajado para aprovechar la experiencia adquirida en las etapas anteriores y mejorar la implementación de GENTSA mediante críticas constructivas.

GENTSA, como producto de esta investigación, es la culminación de la combinación de los mejore aspectos de cada metaheurística, cuidadosamente analizados a través del modelo STN. Este enfoque ha permitido evaluar y optimizar la capacidad de intensificación y diversificación de cada metaheurística seleccionada, compensando así las debilidades

individuales y aumentando su poder para encontrar soluciones óptimas y evitar mínimos locales.

Cabe destacar que GENTSA fue específicamente desarrollada para abordar el modelo FLP en el contexto de las empresas MiPYMES textiles del proyecto de investigación ResilTEX. En este sentido, GENTSA ha demostrado su capacidad para lograr una distribución de instalaciones efectiva, con un costo de distribución significativamente menor en el 60% de las ejecuciones. No obstante, se debe considerar que, debido a las restricciones del problema y las metaheurísticas empleadas, GENTSA presenta un tiempo de ejecución más alto en comparación con las metaheurísticas bases, con una diferencia promedio de tiempo de 179.1010 segundos.

Para una evaluación más precisa del rendimiento del algoritmo, se ha realizado pruebas estadísticas rigurosas. La prueba de Shapiro-Wilk ha demostrado que los resultados de las metaheurísticas no siguen una distribución normal, lo que indica la necesidad de utilizar pruebas estadísticas no paramétricas. La prueba de Friedman ha resaltado diferencias significativas entre las metaheurísticas y sus rangos. En base a estos resultados, hemos empleado la prueba post-hoc de Nemenyi para realizar comparaciones detalladas y obtener un análisis exhaustivo. Los resultados obtenidos confirman que GENTSA exhibe un rendimiento notablemente superior a ACO, GA y PSO. Sin embargo, es importante señalar que no contamos con suficiente evidencia para afirmar que GENTSA supera significativamente a SA y TS en términos de rendimiento.

Finalmente, las pruebas adicionales realizadas en GENTSA utilizando cuatro funciones de referencia han demostrado que, aunque no fue diseñada con propósitos generales, puede adaptarse de manera razonable a problemas en dominios más allá de su diseño original.

Por lo tanto, se pueden sugerir algunas direcciones para investigaciones futuras. En el caso de problemas con restricciones altamente complejas, puede resultar beneficioso comenzar eligiendo o estudiando las metaheurísticas más simples, aquellas con menos parámetros. Además, al combinar metaheurísticas para crear una metaheurística híbrida, es fundamental considerar tanto la intensificación como la diversificación como componentes esenciales y asegurarse de que estén equilibrados. Para lograr esto, se recomienda utilizar el enfoque de visualización STN, que permite comprender mejor el comportamiento de las metaheurísticas. De esta manera, se podrán tomar decisiones más informadas al seleccionar las metaheurísticas a fusionar o simplemente obtener una mejor comprensión de cómo se comportan diferentes metaheurísticas u otros métodos.

Otro aspecto para tener en cuenta es la elección del lenguaje de programación al implementar los algoritmos. En este trabajo, se utilizó el lenguaje de programación C++. No obstante, también se pueden considerar otros lenguajes como R o Python, que ofrecen una amplia gama de bibliotecas y pueden agilizar el proceso de implementación. Incluso, se podrían utilizar varios lenguajes de una misma implementación para evaluar las diferencias en tiempos de ejecución y uso de recursos computaciones. Es importante tener en cuenta la complejidad del problema, ya que la verificación de ciertas restricciones puede aumentar el tiempo de ejecución del algoritmo.

Limitaciones

El presente trabajo tiene algunas limitaciones que es importante mencionar. En primer lugar, el estado del arte se basó en un número reducido de funciones, limitándose a solo cuatro. Sin embargo, es importante destacar que el algoritmo no fue diseñado como una metaheurística híbrida de propósito general, sino que se modeló para un contexto específico. Esto significa que sus resultados podrían no ser tan buenos en otros posibles contextos en los que se pueda probar. Además, en el caso de estudio, la distribución actual no incluía medidas explícitas de las áreas restringidas, lo que obligó al estudio a estimar estas medidas. Asimismo, los datos obtenidos para el caso de estudio son privados y no se pueden compartir de manera pública, lo cual representa un obstáculo para visualizar los resultados al intentar reproducir el algoritmo.

Otra limitación se relaciona con el modelo STN utilizado para visualizar el comportamiento de las metaheurísticas. Este modelo fue programado en el lenguaje de programación R y estaba diseñado para admitir un máximo de tres metaheurísticas. Dado que en este trabajo se utilizaron cinco metaheurísticas, no fue posible probar conjuntamente todos los algoritmos y obtener una visión más completa de su comportamiento colectivo.

Referencias

Aarts, E., Korst, J., & Michiels, W. (2005). Simulated Annealing. In E. K. Burke & G. Kendall (Eds.), *Search Methodologies* (pp. 187–210). Springer US. https://doi.org/10.1007/0-387-28356-0_7

Abdel-Basset, M., Abdel-Fatah, L., & Sangaiah, A. K. (2018). Metaheuristic Algorithms: A Comprehensive Review. In *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications* (pp. 185–231). Elsevier. https://doi.org/10.1016/B978-0-12-813314-9.00010-4

Abdinnour-Helm, S., & Hadley, S. W. (2000). Tabu search based heuristics for multi-floor facility layout. *International Journal of Production Research*, *38*(2), Article 2. https://doi.org/10.1080/002075400189464

Ahmadi, A., Pishvaee, M. S., & Akbari Jokar, M. R. (2017). A survey on multi-floor facility layout problems. *Computers & Industrial Engineering*, *107*, 158–170. https://doi.org/10.1016/j.cie.2017.03.015

Ahmadi-Javid, A., & Ardestani-Jaafari, A. (2021). The unequal area facility layout problem with shortest single-loop AGV path: How material handling method matters. *International Journal of Production Research*, *59*(8), Article 8. https://doi.org/10.1080/00207543.2020.1733124

Akojwar, S. G., & Kshirsagar, P. R. (2016). *Performance Evolution of Optimization Techniques for Mathematical Benchmark Functions*. *1*.

Aleksić, A., Stefanović, M., Arsovski, S., & Tadić, D. (2013). An assessment of organizational resilience potential in SMEs of the process industry, a fuzzy approach. *Journal of Loss Prevention in the Process Industries*, *26*(6), 1238–1245. https://doi.org/10.1016/j.jlp.2013.06.004

Baghel, M., Agrawal, S., & Silakari, S. (2012). Survey of Metaheuristic Algorithms for Combinatorial Optimization. *International Journal of Computer Applications*, *58*(19), Article 19. https://doi.org/10.5120/9391-3813

Beheshti, Z., & Shamsuddin, S. M. H. (2013). A Review of Population-based Meta-Heuristic Algorithm. *Int. J. Advance. Soft Comput. Appl.*, *5*(1), Article 1.

Blum, C. (2003). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*, *35*(3), Article 3.

Blum, C., Puchinger, J., Raidl, G., & Roli, A. (2010). *A BRIEF SURVEY ON HYBRID METAHEURISTICS*.

Bozorgi, N., Abedzadeh, M., & Zeinali, M. (2015). Tabu search heuristic for efficiency of dynamic facility layout problem. *The International Journal of Advanced Manufacturing Technology*, *77*(1–4), Article 1–4. https://doi.org/10.1007/s00170-014-6460-9

Burkard, R. E. (1984). Quadratic assignment problems. *European Journal of Operational Research*, *15*, 283–289.

Chiang, W.-C., & Kouvelis, P. (1996). An improved tabu search heuristic for solving facility layout design problems. *International Journal of Production Research*, *34*(9), Article 9. https://doi.org/10.1080/00207549608905045

Curry, D. M., & Dagli, C. H. (2014). Computational Complexity Measures for Many-objective Optimization Problems. *Procedia Computer Science*, *36*, 185–191. https://doi.org/10.1016/j.procs.2014.09.077

De Armas, J., Lalla-Ruiz, E., Tilahun, S. L., & Voß, S. (2022). Similarity in metaheuristics: A gentle step towards a comparison methodology. *Natural Computing*, *21*(2), Article 2. https://doi.org/10.1007/s11047-020-09837-9

Demsar, J. (2006). *Statistical Comparisons of Classifiers over Multiple Data Sets*. *7*, 1–30.

Derakhshan Asl, A., & Wong, K. Y. (2017). Solving unequal-area static and dynamic facility layout problems using modified particle swarm optimization. *Journal of Intelligent Manufacturing*, *28*(6), Article 6. https://doi.org/10.1007/s10845-015-1053-5

Dorigo, M. (Ed.). (2004). *Ant colony optimization and swarm intelligence: 4th international workshop, ANTS 2004, Brussels, Belgium, September 5-8, 2004: proceedings*. Springer.

Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine*, *1*(4), Article 4. https://doi.org/10.1109/MCI.2006.329691

Dorigo, M., & Blum, C. (2005). Ant colony optimization theory: A survey. *Theoretical Computer Science*, *344*(2–3), Article 2–3. https://doi.org/10.1016/j.tcs.2005.05.020

Drira, A., Pierreval, H., & Hajri-Gabouj, S. (2006). Facility Layout Problems: A Literature Analysis. *IFAC Proceedings Volumes*, *39*(3), Article 3. https://doi.org/10.3182/20060517-3-FR-2903.00208

Du, K.-L., & Swamy, M. N. S. (2016a). Particle Swarm Optimization. In K.-L. Du & M. N. S. Swamy, *Search and Optimization by Metaheuristics* (pp. 153–173). Springer International Publishing. https://doi.org/10.1007/978-3-319-41192-7_9

Du, K.-L., & Swamy, M. N. S. (2016b). Simulated Annealing. In K.-L. Du & M. N. S. Swamy, *Search and Optimization by Metaheuristics* (pp. 29–36). Springer International Publishing. https://doi.org/10.1007/978-3-319-41192-7_2

El-Shorbagy, M. A., & Hassanien, A. E. (2018). Particle Swarm Optimization from Theory to Applications: *International Journal of Rough Sets and Data Analysis*, *5*(2), Article 2. https://doi.org/10.4018/IJRSDA.2018040101

Erdemir, E., & Alpaslan Altun, A. (2022). A New Metaheuristic Approach to Solving Benchmark Problems: Hybrid Salp Swarm Jaya Algorithm. *Computers, Materials & Continua*, *71*(2), Article 2. https://doi.org/10.32604/cmc.2022.022797

Ferreira, J. S. (2013). Multimethodology in Metaheuristics. *Journal of the Operational Research Society*, *64*(6), Article 6. https://doi.org/10.1057/jors.2012.88

Fidanova, S. (2021). *Ant Colony Optimization and Applications* (Vol. 947). Springer International Publishing. https://doi.org/10.1007/978-3-030-67380-2

Fleming, P. S., Koletsi, D., & Pandis, N. (2014). Blinded by PRISMA: Are Systematic Reviewers Focusing on PRISMA and Ignoring Other Guidelines? *PLoS ONE*, *9*(5), Article 5. https://doi.org/10.1371/journal.pone.0096407

Flores-Siguenza, P., Guamán, R., Rosero-Mantilla, C., Siguenza-Guzman, L., & Jadan-Avilés, D. (2022a). Textile Micro, Small and Medium Enterprises (MSME) Layout Dynamics in the Ecuadorian Context. In M. Botto-Tobar, S. Montes León, P. Torres-Carrión, M. Zambrano Vizuete, & B. Durakovic (Eds.), *Applied Technologies* (Vol. 1535, pp. 265–276). Springer International Publishing. https://doi.org/10.1007/978-3-031-03884-6_20

Flores-Siguenza, P., Siguenza-Guzman, L., Lema, F., Tigre, F., Vanegas, P., & Aviles-González, J. (2022b). A Systematic Literature Review of Facility Layout Problems and Resilience Factors in the Industry. In M. Botto-Tobar, S. Montes León, P. Torres-Carrión, M. Zambrano Vizuete, & B. Durakovic (Eds.), *Applied Technologies* (Vol. 1535, pp. 252–264). Springer International Publishing. https://doi.org/10.1007/978-3-031-03884-6_19

García-Hernández, L., Arauzo-Azofra, A., Salas-Morera, L., Pierreval, H., & Corchado, E. (2015). Facility layout design using a multi-objective interactive genetic algorithm to support the DM. *Expert Systems*, *32*(1), Article 1. https://doi.org/10.1111/exsy.12064

García-Hernández, L., Palomo-Romero, J. M., Salas-Morera, L., Arauzo-Azofra, A., & Pierreval, H. (2015). A novel hybrid evolutionary approach for capturing decision maker knowledge into the unequal area facility layout problem. *Expert Systems with Applications*, *42*(10), Article 10. https://doi.org/10.1016/j.eswa.2015.01.037

Garcia-Hernandez, L., Salas-Morera, L., Carmona-Munoz, C., Abraham, A., & Salcedo-Sanz, S. (2020). A Hybrid Coral Reefs Optimization—Variable Neighborhood Search Approach for the Unequal Area Facility Layout Problem. *IEEE Access*, *8*, 134042–134050. https://doi.org/10.1109/ACCESS.2020.3010577

Garden, R. W., & Engelbrecht, A. P. (2014). Analysis and classification of optimisation benchmark functions and benchmark suites. *2014 IEEE Congress on Evolutionary Computation (CEC)*, 1641–1649. https://doi.org/10.1109/CEC.2014.6900240

Geman, S., & Geman, D. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-6*(6), Article 6.

Gendreau, M., & Potvin, J.-Y. (2014). Tabu Search. In E. K. Burke & G. Kendall (Eds.), *Search Methodologies* (pp. 243–263). Springer US. https://doi.org/10.1007/978-1-4614-6940-7_9

Ghadikolaei, Y. K., & Shahanaghi, K. (2013). Multi-floor dynamic facility layout: A simulated annealing-based solution. *International Journal of Operational Research*, *16*(4), Article 4. https://doi.org/10.1504/IJOR.2013.052711

Ghaheri, A., Shoar, S., Naderan, M., & Hoseini, S. S. (2015). The Applications of Genetic Algorithms in Medicine. *Oman Medical Journal*, *30*(6), Article 6. https://doi.org/10.5001/omj.2015.82

Glover, F. (1990). Tabu Search: A Tutorial. *Interfaces*, *20*(4), Article 4. https://doi.org/10.1287/inte.20.4.74

Gonçalves, J. F., & Resende, M. G. C. (2015). A biased random-key genetic algorithm for the unequal area facility layout problem. *European Journal of Operational Research*, *246*(1), Article 1. https://doi.org/10.1016/j.ejor.2015.04.029

Guan, C., Zhang, Z., Liu, S., & Gong, J. (2019). Multi-objective particle swarm optimization for multi-workshop facility layout problem. *Journal of Manufacturing Systems*, *53*, 32–48. https://doi.org/10.1016/j.jmsy.2019.09.004

Guan, J., & Lin, G. (2016). Hybridizing variable neighborhood search with ant colony optimization for solving the single row facility layout problem. *European Journal of Operational Research*, *248*(3), Article 3. https://doi.org/10.1016/j.ejor.2015.08.014

Gunnell, K. E., Belcourt, V. J., Tomasone, J. R., & Weeks, L. C. (2022). Systematic review methods. *International Review of Sport and Exercise Psychology*, *15*(1), Article 1. https://doi.org/10.1080/1750984X.2021.1966823

Hosseini, S., Khaled, A. A., & Vadlamani, S. (2014). Hybrid imperialist competitive algorithm, variable neighborhood search, and simulated annealing for dynamic facility layout problem. *Neural Computing and Applications*, *25*(7–8), Article 7–8. https://doi.org/10.1007/s00521-014-1678-x

Hosseini, S. S., Azimi, P., Sharifi, M., & Zandieh, M. (2021). A new soft computing algorithm based on cloud theory for dynamic facility layout problem. *RAIRO - Operations Research*, *55*, S2433–S2453. https://doi.org/10.1051/ro/2020127

Hosseini-Nasab, H., & Emami, L. (2013). A hybrid particle swarm optimisation for dynamic facility layout problem. *International Journal of Production Research*, *51*(14), Article 14. https://doi.org/10.1080/00207543.2013.774486

Hosseini-Nasab, H., Fereidouni, S., Fatemi Ghomi, S. M. T., & Fakhrzad, M. B. (2018). Classification of facility layout problems: A review study. *The International Journal of Advanced Manufacturing Technology*, *94*(1–4), Article 1–4. https://doi.org/10.1007/s00170-017-0895-8

Hu, B., & Yang, B. (2019). A particle swarm optimization algorithm for multi-row facility layout problem in semiconductor fabrication. *Journal of Ambient Intelligence and Humanized Computing*, *10*(8), Article 8. https://doi.org/10.1007/s12652-018-1037-3

Hunagund, I. B., Madhusudanan Pillai, V., & Kempaiah, U. N. (2018). A simulated annealing algorithm for unequal area dynamic facility layout problems with flexible bay structure. *International Journal of Industrial Engineering Computations*, 307–330. https://doi.org/10.5267/j.ijiec.2017.8.004

Hussain, K., Mohd Salleh, M. N., Cheng, S., & Naseem, R. (2017). Common Benchmark Functions for Metaheuristic Evaluation: A Review. *JOIV : International Journal on Informatics Visualization*, *1*(4–2), 218. https://doi.org/10.30630/joiv.1.4-2.65

Jerin Leno, I., Saravana Sankar, S., & Ponnambalam, S. G. (2018). MIP model and elitist strategy hybrid GA–SA algorithm for layout design. *Journal of Intelligent Manufacturing*, *29*(2), Article 2. https://doi.org/10.1007/s10845-015-1113-x

Jongen, H., Meer, K., & Triesch, E. (2004). *Optimization Theory*. Kluwer Academic Publishers.

Kalita, Z., & Datta, D. (2018). A constrained single-row facility layout problem. *The International Journal of Advanced Manufacturing Technology*, *98*(5–8), Article 5–8. https://doi.org/10.1007/s00170-018-2370-6

Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: Past, present, and future. *Multimedia Tools and Applications*, *80*(5), Article 5. https://doi.org/10.1007/s11042-020-10139-6

Kaveh, M., Dalfard, V. M., & Amiri, S. (2014). A new intelligent algorithm for dynamic facility layout problem in state of fuzzy constraints. *Neural Computing and Applications*, *24*(5), Article 5. https://doi.org/10.1007/s00521-013-1339-5

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks*, *4*, 1942–1948. https://doi.org/10.1109/ICNN.1995.488968

Khajemahalle, L., Emami, S., & Keshteli, R. N. (2021). A hybrid nested partitions and simulated annealing algorithm for dynamic facility layout problem: A robust optimization approach. *INFOR: Information Systems and Operational Research*, *59*(1), Article 1. https://doi.org/10.1080/03155986.2020.1788328

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science*, *220*(4598), Article 4598. https://doi.org/10.1126/science.220.4598.671

Kitchenham, B. (2004). Procedures for Performing Systematic Reviews. *08/2004*, *33*, 34.

Komarudin, & Wong, K. Y. (2010). Applying Ant System for solving Unequal Area Facility Layout Problems. *European Journal of Operational Research*, *202*(3), Article 3. https://doi.org/10.1016/j.ejor.2009.06.016

Koopmans, T. C., & Beckmann, M. (1957). Assignment Problems and the Location of Economic Activities. *Econometrica*, *25*(1), 53. https://doi.org/10.2307/1907742

Kothari, R., & Ghosh, D. (2013). Tabu search for the single row facility layout problem using exhaustive 2-opt and insertion neighborhoods. *European Journal of Operational Research*, *224*(1), Article 1. https://doi.org/10.1016/j.ejor.2012.07.037

Kovacs, A. (2018). On the Computational Complexity of Tariff Optimization for Demand Response Management. *IEEE Transactions on Power Systems*, *33*(3), 3204–3206. https://doi.org/10.1109/TPWRS.2018.2802198

Kramer, O. (2017). Genetic Algorithms. In O. Kramer, *Genetic Algorithm Essentials* (Vol. 679, pp. 11–19). Springer International Publishing. https://doi.org/10.1007/978-3-319-52156-5_2

Kulkarni, Mr. N. K., Patekar, Ms. S., Bhoskar, Ms. T., Kulkarni, Mr. O., Kakandikar, G. M., & Nandedkar, V. M. (2015). Particle Swarm Optimization Applications to Mechanical Engineering- A Review. *Materials Today: Proceedings*, *2*(4–5), Article 4–5. https://doi.org/10.1016/j.matpr.2015.07.223

Kulturel-Konak, S., & Konak, A. (2015). A large-scale hybrid simulated annealing algorithm for cyclic facility layout problems. *Engineering Optimization*, *47*(7), Article 7. https://doi.org/10.1080/0305215X.2014.933825

Kusiak, A., & Heragu, S. S. (1987). The facility layout problem. *European Journal of Operational Research*, *29*(3), 229–251. https://doi.org/10.1016/0377-2217(87)90238-4

Lee, C. K. H. (2018). A review of applications of genetic algorithms in operations management. *Engineering Applications of Artificial Intelligence*, *76*, 1–12. https://doi.org/10.1016/j.engappai.2018.08.011

Lenin, N., Siva Kumar, M., Ravindran, D., & Islam, M. N. (2014). A Tabu Search for Multi-Objective Single Row Facility Layout Problem. *Journal of Advanced Manufacturing Systems*, *13*(01), Article 01. https://doi.org/10.1142/S0219686714500024

Lin, Z., & Yingjie, Z. (2019). Solving the Facility Layout Problem with Genetic Algorithm. *2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA)*, 164–168. https://doi.org/10.1109/IEA.2019.8715148

Liu, J., & Liu, J. (2019). Applying multi-objective ant colony optimization algorithm for solving the unequal area facility layout problems. *Applied Soft Computing*, *74*, 167–189. https://doi.org/10.1016/j.asoc.2018.10.012

Liu, J., Liu, J., Yan, X., & Peng, B. (2020). A heuristic algorithm combining Pareto optimization and niche technology for multi-objective unequal area facility layout problem. *Engineering Applications of Artificial Intelligence*, *89*, 103453. https://doi.org/10.1016/j.engappai.2019.103453

Liu, J., Zhang, H., He, K., & Jiang, S. (2018). Multi-objective particle swarm optimization algorithm based on objective space division for the unequal-area facility layout problem. *Expert Systems with Applications*, *102*, 179–192. https://doi.org/10.1016/j.eswa.2018.02.035

Llivisaca, J. C., Siguenza-Guzman, L., & Aviles-González, J. (2022). *Facility layout model under resilience and productivity requirements*.

Loiola, E. M., De Abreu, N. M. M., Boaventura-Netto, P. O., Hahn, P., & Querido, T. (2007). A survey for the quadratic assignment problem. *European Journal of Operational Research*, *176*(2), 657–690. https://doi.org/10.1016/j.ejor.2005.09.032

Macuzić, I., Tadić, D., Aleksić, A., & Stefanović, M. (2016). A two step fuzzy model for the assessment and ranking of organizational resilience factors in the process industry. *Journal of Loss Prevention in the Process Industries*, *40*, 122–130. https://doi.org/10.1016/j.jlp.2015.12.013

Matai, R. (2015). Solving multi objective facility layout problem by modified simulated annealing. *Applied Mathematics and Computation*, *261*, 302–311. https://doi.org/10.1016/j.amc.2015.03.107

Mirjalili, S. (2019). Ant Colony Optimisation. In S. Mirjalili, *Evolutionary Algorithms and Neural Networks* (Vol. 780, pp. 33–42). Springer International Publishing. https://doi.org/10.1007/978-3-319-93025-1_3

Moslemipour, G. (2018). A hybrid CS-SA intelligent approach to solve uncertain dynamic facility layout problems considering dependency of demands. *Journal of Industrial Engineering International*, *14*(2), Article 2. https://doi.org/10.1007/s40092-017-0222-x

Moslemipour, G., Lee, T. S., & Loong, Y. T. (2018). Solving stochastic dynamic facility layout problems using proposed hybrid AC-CS-SA meta-heuristic algorithm. *International Journal of Industrial and Systems Engineering*, *28*, 31. https://doi.org/10.1504/IJISE.2018.088561

Neumann, F., & Witt, C. (2010). Combinatorial Optimization and Computational Complexity. In F. Neumann & C. Witt, *Bioinspired Computation in Combinatorial Optimization* (pp. 9–19). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-16544-3_2

Ochoa, G., Malan, K. M., & Blum, C. (2021). Search trajectory networks: A tool for analysing and visualising the behaviour of metaheuristics. *Applied Soft Computing*, *109*, 107492. https://doi.org/10.1016/j.asoc.2021.107492

Osaba, E., Carballedo, R., Diaz, F., Onieva, E., Masegosa, A. D., & Perallos, A. (2018). Good practice proposal for the implementation, presentation, and comparison of metaheuristics for solving routing problems. *Neurocomputing*, *271*, 2–8. https://doi.org/10.1016/j.neucom.2016.11.098

Paes, F. G., Pessoa, A. A., & Vidal, T. (2017). A hybrid genetic algorithm with decomposition phases for the Unequal Area Facility Layout Problem. *European Journal of Operational Research*, *256*(3), Article 3. https://doi.org/10.1016/j.ejor.2016.07.022

Palomo-Romero, J. M., Salas-Morera, L., & García-Hernández, L. (2017). An island model genetic algorithm for unequal area facility layout problems. *Expert Systems with Applications*, *68*, 151–162. https://doi.org/10.1016/j.eswa.2016.10.004

Palubeckis, G. (2017). Single row facility layout using multi-start simulated annealing. *Computers & Industrial Engineering*, *103*, 1–16. https://doi.org/10.1016/j.cie.2016.09.026

Peng, Y., Zeng, T., Fan, L., Han, Y., & Xia, B. (2018). An Improved Genetic Algorithm Based Robust Approach for Stochastic Dynamic Facility Layout Problem. *Discrete Dynamics in Nature and Society*, *2018*, 1–8. https://doi.org/10.1155/2018/1529058

Peres, F., & Castelli, M. (2021). Combinatorial Optimization Problems and Metaheuristics: Review, Challenges, Design, and Development. *Applied Sciences*, *11*(14), 6449. https://doi.org/10.3390/app11146449

Pérez-Gosende, P., Mula, J., & Díaz-Madroñero, M. (2021). Facility layout planning. An extended literature review. *International Journal of Production Research*, *59*(12), Article 12. https://doi.org/10.1080/00207543.2021.1897176

Phadermrod, B., Crowder, R. M., & Wills, G. B. (2019). Importance-Performance Analysis based SWOT analysis. *International Journal of Information Management*, *44*, 194–203. https://doi.org/10.1016/j.ijinfomgt.2016.03.009

Pourhassan, M. R., & Raissi, S. (2019). A Hybrid Genetic and Particle Swarm Optimization Algorithms for Dynamic Facility Layout Problem with Multiple Transporters. *2019 15th Iran International Industrial Engineering Conference (IIIEC)*, 92–98. https://doi.org/10.1109/IIIEC.2019.8720630

Pournaderi, N., Ghezavati, V. R., & Mozafari, M. (2019). Developing a mathematical model for the dynamic facility layout problem considering material handling system and optimizing it using cloud theory-based simulated annealing algorithm. *SN Applied Sciences*, *1*(8), Article 8. https://doi.org/10.1007/s42452-019-0865-x

Pourvaziri, H., & Naderi, B. (2014). A hybrid multi-population genetic algorithm for the dynamic facility layout problem. *Applied Soft Computing*, *24*, 457–469. https://doi.org/10.1016/j.asoc.2014.06.051

Pradeepmon, T. G., Panicker, V. V., & Sridharan, R. (2018). A heuristic algorithm enhanced with probability-based incremental learning and local search for dynamic facility layout problems. *International Journal of Applied Decision Sciences*, *11*(4), Article 4. https://doi.org/10.1504/IJADS.2018.10012583

Raidl, G. R. (2006). A Unified View on Hybrid Metaheuristics. In F. Almeida, M. J. Blesa Aguilera, C. Blum, J. M. Moreno Vega, M. Pérez Pérez, A. Roli, & M. Sampels (Eds.), *Hybrid Metaheuristics* (Vol. 4030, pp. 1–12). Springer Berlin Heidelberg. https://doi.org/10.1007/11890584_1

Ralston, A., Reilly, E. D., & Hemmendinger, D. (Eds.). (2003). *Encyclopedia of computer science* (4th ed). Wiley.

Razali, N. M., & Wah, Y. B. (2011). *Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests*. *2*(1), 21–33.

Rifai, A. P., Windras Mara, S. T., Kusumastuti, P. A., & Wiraningrum, R. G. (2020). A Genetic Algorithm for the Double Row Layout Problem. *Jurnal Teknik Industri*, *22*(2), Article 2. https://doi.org/10.9744/jti.22.2.85-92

Safarzadeh, S., & Koosha, H. (2017). Solving an extended multi-row facility layout problem with fuzzy clearances using GA. *Applied Soft Computing*, *61*, 819–831. https://doi.org/10.1016/j.asoc.2017.09.003

Samarghandi, H., & Eshghi, K. (2010). An efficient tabu algorithm for the single row facility layout problem. *European Journal of Operational Research*, *205*(1), Article 1. https://doi.org/10.1016/j.ejor.2009.11.034

Schrijver, A. (2005). On the History of Combinatorial Optimization (Till 1960). In *Handbooks in Operations Research and Management Science* (Vol. 12, pp. 1–68). Elsevier. https://doi.org/10.1016/S0927-0507(05)12001-5

Sepehrirad, R., Azar, A., & Dabestani, R. (2015). *Methodology of Soft Operational Research*. *22*.

Servicio Nacional de Gestión de Riesgos y Emergencias. (2020). Resoluciones COE Nacional 22 de Mayo 2020. https://www.gestionderiesgos.gob.ec/resoluciones-coe-nacional-22-de-mayo-2020/

Shi, Y., & Eberhart, R. (1998). A modified particle swarm optimizer. *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, 69–73. https://doi.org/10.1109/ICEC.1998.699146

Siddique, N., & Adeli, H. (2016). Simulated Annealing, Its Variants and Engineering Applications. *International Journal on Artificial Intelligence Tools*, *25*(06), Article 06. https://doi.org/10.1142/S0218213016300015

Silva, A., Coelho, L. C., & Darvish, M. (2021). Quadratic assignment problem variants: A survey and an effective parallel memetic iterated tabu search. *European Journal of Operational Research*, *292*(3), 1066–1084. https://doi.org/10.1016/j.ejor.2020.11.035

Singh, S. P. (2009). *Solving Facility Layout Problem: Three-level Tabu Search Metaheuristic Approach* (1). *1*(1), Article 1.

Singh, S. P., & Sharma, R. R. K. (2006). A review of different approaches to the facility layout problems. *The International Journal of Advanced Manufacturing Technology*, *30*(5–6), 425–433. https://doi.org/10.1007/s00170-005-0087-9

Sipser, M. (1992). The history and status of the P versus NP question. *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing - STOC '92*, 603–618. https://doi.org/10.1145/129712.129771

Sivanandam, S. N., & Deepa, S. N. (2007). *Introduction to genetic algorithms*. Springer.

Slowik, A., & Kwasnicka, H. (2020). Evolutionary algorithms and their applications to engineering problems. *Neural Computing and Applications*, *32*(16), Article 16. https://doi.org/10.1007/s00521-020-04832-8

Sotamba, L., Peña, M., & Siguenza-Guzman, L. (2023). Driver Analysis to solve Dynamic Facility Layout Problems: A Literature Review. *Lecture Notes in Mechanical*

Engineering. Accepted: Flexible Automation and Intelligent Manufacturing International Conference.

SRI. (2020). https://srienlinea.sri.gob.ec/saiku-ui/

Suman, B., & Kumar, P. (2006). A survey of simulated annealing as a tool for single and multiobjective optimization. *Journal of the Operational Research Society*, *57*(10), Article 10. https://doi.org/10.1057/palgrave.jors.2602068

Tayal, A., Kose, U., Solanki, A., Nayyar, A., & Saucedo, J. A. M. (2020). Efficiency analysis for stochastic dynamic facility layout problem using meta-heuristic, data envelopment analysis and machine learning. *Computational Intelligence*, *36*(1), Article 1. https://doi.org/10.1111/coin.12251

Tayal, A., & Singh, S. P. (2015). *ME-TM-18: FLEXIBLE LAYOUT DESIGN FOR UNCERTAIN PRODUCT DEMAND BY INTEGRATING FIREFLY AND CHAOTIC SIMULATED ANNEALING APPROACH*.

Tayal, A., & Singh, S. P. (2017). Integrated SA-DEA-TOPSIS-based solution approach for multi objective stochastic dynamic facility layout problem. *International Journal of Business and Systems Research*, *11*(1/2), Article 1/2. https://doi.org/10.1504/IJBSR.2017.080839

Tayal, A., & Singh, S. P. (2019). Analysis of simulated annealing cooling schemas for design of optimal flexible layout under uncertain dynamic product demand. *International Journal of Operational Research*, *34*, 85–103. https://doi.org/10.1504/IJOR.2019.10017909

Tewani, K. (2017). Ant colony optimization algorithm: Advantages, applications and challenges. *Mathematical and Computer Modelling*.

Tian, D., & Shi, Z. (2018). MPSO: Modified particle swarm optimization and its applications. *Swarm and Evolutionary Computation*, *41*, 49–68. https://doi.org/10.1016/j.swevo.2018.01.011

Ting, T. O., Yang, X.-S., Cheng, S., & Huang, K. (2015). Hybrid Metaheuristic Algorithms: Past, Present, and Future. In X.-S. Yang (Ed.), *Recent Advances in Swarm Intelligence and Evolutionary Computation* (Vol. 585, pp. 71–83). Springer International Publishing. https://doi.org/10.1007/978-3-319-13826-8_4

Turanoğlu, B., & Akkaya, G. (2018). A new hybrid heuristic algorithm based on bacterial foraging optimization for the dynamic facility layout problem. *Expert Systems with Applications*, *98*, 93–104. https://doi.org/10.1016/j.eswa.2018.01.011

Turgay, S. (2018). Multi Objective Simulated Annealing Approach for Facility Layout Design. *International Journal of Mathematical, Engineering and Management Sciences*, *3*(4), Article 4. https://doi.org/10.33889/IJMEMS.2018.3.4-026

Uddin, M. S. (2015). Hybrid Genetic Algorithm and Variable Neighborhood Search for Dynamic Facility Layout Problem. *Open Journal of Optimization*, *04*(04), Article 04. https://doi.org/10.4236/ojop.2015.44015

Vázquez, J., Llivisaca, J. C., Ortiz, D., Naranjo, I., & Siguenza-Guzman, L. (2022). Technologies Applied to Solve Facility Layout Problems with Resilience – A Systematic Review. In *Applied Technologies* (Vol. 1535, pp. 67–79). Springer International Publishing. https://doi.org/10.1007/978-3-031-03884-6_5

Wan, X., Zuo, X., Li, X., & Zhao, X. (2022). A hybrid multiobjective GRASP for a multi-row facility layout problem with extra clearances. *International Journal of Production Research*, *60*(3), Article 3. https://doi.org/10.1080/00207543.2020.1847342

Wang, D., Tan, D., & Liu, L. (2018). Particle swarm optimization algorithm: An overview. *Soft Computing*, *22*(2), Article 2. https://doi.org/10.1007/s00500-016-2474-6

Wang, S., Zuo, X., & Zhao, X. (2014). Solving dynamic double-row layout problem via an improved simulated annealing algorithm. *2014 IEEE Congress on Evolutionary Computation (CEC)*, 1299–1304. https://doi.org/10.1109/CEC.2014.6900352

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), Article 1. https://doi.org/10.1109/4235.585893

Xiao, X., Hu, Y., Wang, W., & Ren, W. (2019). A robust optimization approach for unequal-area dynamic facility layout with demand uncertainty. *Procedia CIRP*, *81*, 594–599. https://doi.org/10.1016/j.procir.2019.03.161

Xiao, Y. J., Zheng, Y., Zhang, L. M., & Kuo, Y. H. (2016). A combined zone-LP and simulated annealing algorithm for unequal-area facility layout problem. *Advances in Production Engineering & Management*, *11*(4), Article 4. https://doi.org/10.14743/apem2016.4.225

Yu, M., Zuo, X., & Murray, C. C. (2014). A tabu search heuristic for the single row layout problem with shared clearances. *2014 IEEE Congress on Evolutionary Computation (CEC)*, 819–825. https://doi.org/10.1109/CEC.2014.6900353

Zha, S., Guo, Y., Huang, S., Wu, Q., & Tang, P. (2020). A hybrid optimization approach for unequal-sized dynamic facility layout problems under fuzzy random demands. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *234*(3), Article 3. https://doi.org/10.1177/0954405419883046

Zhang, Y., Wang, S., & Ji, G. (2015). A Comprehensive Survey on Particle Swarm Optimization Algorithm and Its Applications. *Mathematical Problems in Engineering*, *2015*, 1–38. https://doi.org/10.1155/2015/931256

Zhou, J.-L., Wang, J.-S., Zhang, Y.-X., Guo, Q.-S., Li, H., & Lu, Y.-X. (2020). Particle Swarm Optimization Algorithm with Variety Inertia Weights to Solve Unequal Area Facility Layout Problem. *2020 Chinese Control And Decision Conference (CCDC)*, 4240–4245. https://doi.org/10.1109/CCDC49329.2020.9163977

Zhu, T., Balakrishnan, J., & Cheng, C. H. (2018). Recent advances in dynamic facility layout research. *INFOR: Information Systems and Operational Research*, *56*(4), Article 4. https://doi.org/10.1080/03155986.2017.1363591

Zolfi, K., Jouzdani, J., & Shirouyehzad, H. (2023). A novel memory-based simulated annealing algorithm to solve multi-line facility layout problem. *Decision Science Letters*, *12*(1), Article 1. https://doi.org/10.5267/j.dsl.2022.10.005

Zouein, P. P., & Kattan, S. (2022). An improved construction approach using ant colony optimization for solving the dynamic facility layout problem. *Journal of the Operational Research Society*, *73*(7), Article 7. https://doi.org/10.1080/01605682.2021.1920345

Zuo, X., Li, B., Huang, X., Zhou, M., Cheng, C., Zhao, X., & Liu, Z. (2019). Optimizing Hospital Emergency Department Layout via Multiobjective Tabu Search. *IEEE Transactions on Automation Science and Engineering*, *16*(3), Article 3. https://doi.org/10.1109/TASE.2018.2873098

Anexo A Revisión sistemática
Revisión sistemática acerca de las metodologías de solución aplicadas a DFLP, versión enviada a FAIM y la versión extendida.

Enlace:
https://drive.google.com/file/d/17zOEvbpEAyqx6wZcOLbCAJMt12cO8vxj/view?usp=sharing

Primera versión extendida:
https://drive.google.com/file/d/1m4uuDrC8cLaPCfwgCTm_Kzf2CRb5b9IJ/view?usp=sharing

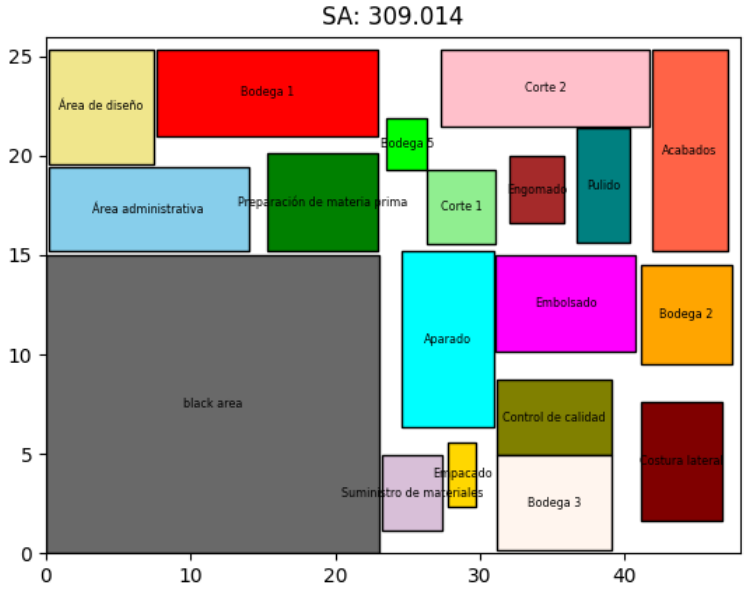
Anexo B Poster conferencia FAIM
Poster presentado en la conferencia FAIM.

Enlace: https://docs.google.com/presentation/d/1qiMqApzjPQ5U64g_HrG8bd3Z-7Gsj4Hk/edit?usp=sharing&ouid=107049036208194622682&rtpof=true&sd=true

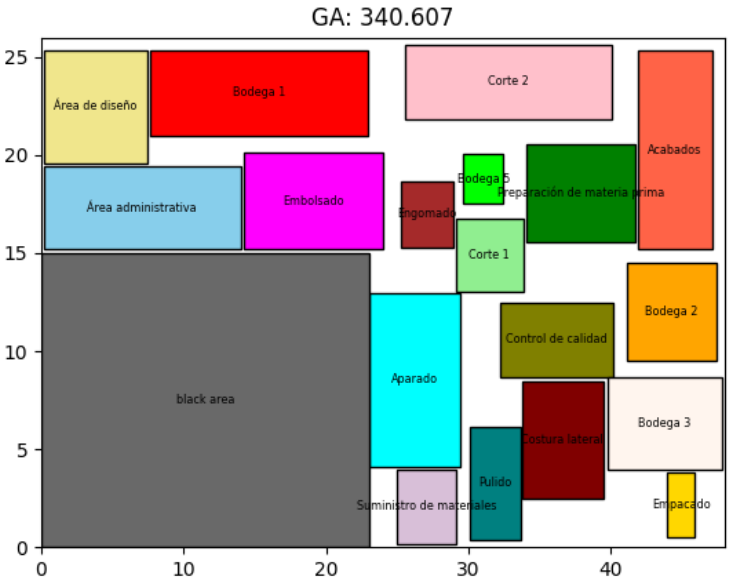
Anexo C Código Fuente de GENTSA
Código fuente de GENTSA.

Enlace:
https://drive.google.com/file/d/1hP6dzlM5Xz2WyZYJfz9n5brzi3klvhkb/view?usp=sharing

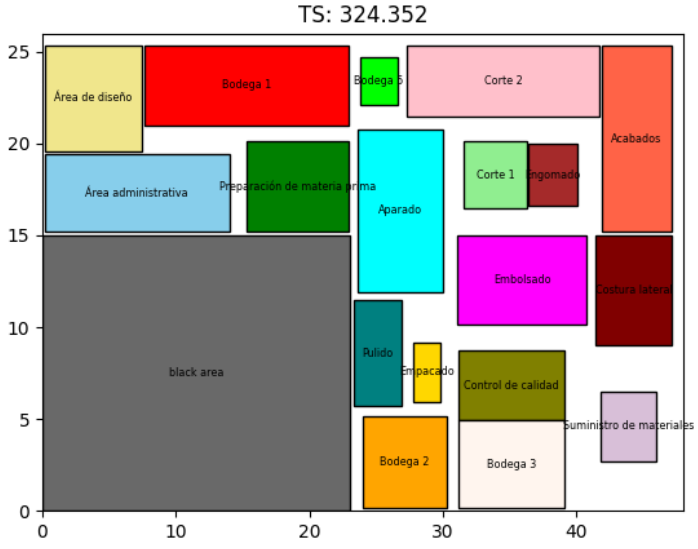
Anexo D Mejores distribuciones de instalaciones obtenidas por los algoritmos
Mejor distribución obtenido por SA.



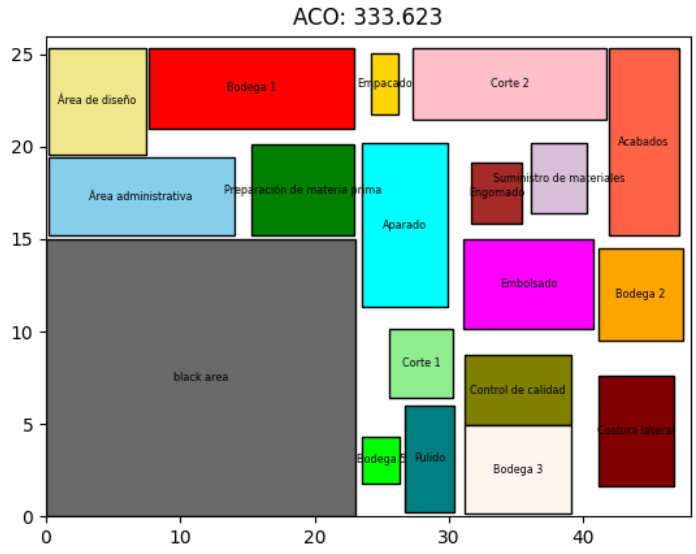
A. Mejor distribución obtenido por GA.



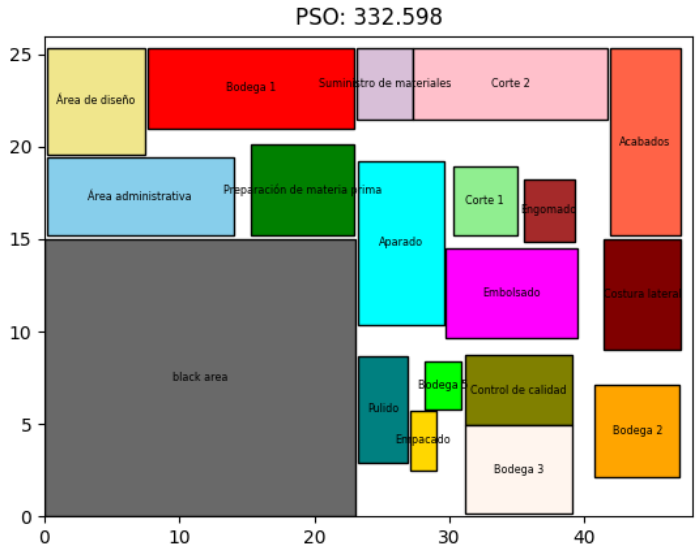
B. Mejor distribución obtenida por TS.



C. Mejor distribución obtenida por ACO.



D. Mejor distribución obtenida por PSO.



E. Imágenes de todas las distribuciones obtenidas por los algoritmos en cada ejecución. Enlace: https://drive.google.com/drive/folders/1IYHcq0voyMUDOgSie5lJxdrEAIVQyT_E?usp=sharing