UNIVERSIDAD DE CUENCA

Facultad de Ciencias Químicas

Carrera de Ingeniería Industrial

Optimización de costos en la cadena de suministro en empresas de ensamblaje: comparación de algoritmos evolutivos en el caso de estudio de ensamblaje de televisores

Trabajo de titulación, modalidad "Proyecto de Investigación", previo a la obtención del título de Ingeniero Industrial

Autora:

Josselin Jimena Orellana Ordoñez

CI: 0705631562

jimenaorellana96@hotmail.com

Director:

Ing. Juan Carlos Llivisaca Villazhanay, M.Sc.

CI: 0105627269

Cuenca, Ecuador

18-febrero-2020

“Optimización de costos en la cadena de suministro en empresas de ensamblaje: comparación de algoritmos evolutivos en el caso de estudio de ensamblaje de televisores”

Orellana Josselin[1], Llivisaca Juan[2]
[1] Universidad de Cuenca, jimena.orellana96@ucuenca.edu.ec
[2]Facultad de Ciencias Químicas, Universidad de Cuenca, juan.llivisaca@ucuenca.edu.ec

Fecha de entrega: 18 de febrero de 2020

ABSTRACT
Optimization of the different resources of the supply chains (CS) of the companies allows having efficient and effective management of them. Generally, this leads to a dilemma because of the different optimization objectives, which in many cases are in conflict. And due to the nature of assembly companies, optimization problems are complex, and metaheuristic techniques are necessary for their solution. The purpose of this work was to develop and solve a CS model for the assembly industry that seeks to optimize two objectives: profit maximization (considering the cost associated with provisioning, production, storage and distribution), and maximizing the level of customer service. The problem was modelled considering multiple products and periods, for two security inventory scenarios, a maximum and a minimum. For this purpose, three algorithms were compared: NSGA-II, MOPSO and MOMA. The multi-objective optimization tool PlatEMO was used to test these algorithms. As main results, the genetic algorithm NSGA-II presented the best results with a 97.87% service level, associated with an increase in profit of $ 391,556.66 concerning the worst result given by MOPSO in the best optimization scenario, with a level Minimum security inventory.

Keywords: Supply chain. Algorithms. PlatEMO. Multi-objective. NSGA-II. MOPSO. MPAES.

RESUMEN
La optimización de los diferentes recursos de las cadenas de suministro (CS) de las empresas permite tener una gestión eficiente y eficaz de los mismos. Generalmente, esto conlleva a un dilema debido a que los diferentes objetivos de optimización en muchos casos están en conflicto. Debido a la naturaleza de las empresas de ensamblaje, los problemas de optimización son complejos, y son necesarias técnicas metaheurísticas para su solución. El propósito de este trabajo fue desarrollar y resolver un modelo de CS para la industria de ensamblaje que busque optimizar dos objetivos: la maximización del beneficio (tomando en cuenta el costo asociado al aprovisionamiento, producción, almacenamiento y distribución), y la maximización del nivel de servicio al cliente. El problema se modeló considerando múltiples productos y periodos para dos escenarios de inventario de seguridad, un máximo y un mínimo. Para este fin, tres algoritmos fueron comparados: NSGA-II, MOPSO y MOMA. La herramienta de optimización multi-objetivo PlatEMO fue empleada para probar estos algoritmos. Como resultados principales, el algoritmo genético NSGA-II presentó un 97.87% de nivel de servicio, asociado a un incremento en el beneficio de $391,556.66 con respecto al MOPSO con un nivel mínimo de inventario de seguridad.

Palabras clave: Cadena de suministro. Algoritmos. PlatEMO. Multi-objetivo. NSGA-II. MOPSO. MPAES.

Índice

Cláusula de licencia y autorización para publicación en el Repositorio Institucional

Josselin Jimena Orellana Ordoñez en calidad de autora y titular de los derechos morales y patrimoniales del trabajo de titulación "Optimización de costos en la cadena de suministro en empresas de ensamblaje: comparación de algoritmos evolutivos en el caso de estudio de ensamblaje de televisores", de conformidad con el Art. 114 del CÓDIGO ORGÁNICO DE LA ECONOMÍA SOCIAL DE LOS CONOCIMIENTOS, CREATIVIDAD E INNOVACIÓN reconozco a favor de la Universidad de Cuenca una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos.

Asimismo, autorizo a la Universidad de Cuenca para que realice la publicación de este trabajo de titulación en el repositorio institucional, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Cuenca, 18 de febrero de 2020

Josselin Jimena Orellana Ordoñez

C.I: 0705631562

Cláusula de Propiedad Intelectual

Josselin Jimena Orellana Ordoñez, autora del trabajo de titulación "Optimización de costos en la cadena de suministro en empresas de ensamblaje: comparación de algoritmos evolutivos en el caso de estudio de ensamblaje de televisores", certifico que todas las ideas, opiniones y contenidos expuestos en la presente investigación son de exclusiva responsabilidad de su autora.

Cuenca, 18 de febrero de 2020

Josselin Jimena Orellana Ordoñez

C.I: 0705631562

1. Introducción

La Cadena de Suministro (CS) ha sido de gran interés en los últimos años, ya que en ésta se abarcan las funciones de planeación, provisión, producción, distribución y devolución. La existencia de numerosas variables de decisión y sus complejas interrelaciones, así como las limitaciones propias de cada sistema hacen de la CS un sistema de alta complejidad (Mendoza et al., 2015; Wu y Blackhurst, 2009). Según Gómez et al. (2008), las empresas no compiten individualmente sino sus cadenas de suministro, por lo que, mejorar el funcionamiento de la CS es importante para muchas empresas. Esto se logra a través de la optimización de costos asociados al abastecimiento, producción y comercialización de bienes y servicios, que proporcionan una ventaja competitiva a la CS.

Para determinar la eficiencia y/o eficacia de la CS es necesario establecer un conjunto de indicadores clave de desempeño (KPI por sus siglas en inglés), que pueden ser financieros y no financieros. Los KPIs permiten comparar y analizar el desempeño de las estrategias en toda la CS mediante su cuantificación, para aplicar procesos de mejora donde sea pertinente. Para alcanzar esto, hay varios modelos o herramientas que permiten gestionar los KPIs, los más conocidos y utilizados son: Balanced Scorecard (BSC) y SCOR. El BSC, propuesto por Norton y Kaplan en 1992, determina los indicadores de desempeño en base a los objetivos estratégicos propios de una organización y se clasifica en cuatro perspectivas (Clientes, Procesos Internos, Aprendizaje y Crecimiento, y Financiero). El modelo SCOR gestiona los indicadores que relacionan a todas las partes de CS a nivel estratégico, táctico y operacional. La combinación de estos dos modelos ha permitido una administración óptima de la cadena e identificar posibles áreas de mejora a través de un estudio detallado de KPIs (Kaplan y Norton, 1992; Stephens, 2001; Wu y Blackhurst, 2009).

Según Movahedipour et al. (2016), la investigación de la CS ha tenido una tendencia de crecimiento exponencial, y señala a la "optimización y el modelado matemático" como el principal tema de investigación. Por ejemplo, los algoritmos evolutivos (AE), que se basan en el comportamiento social y la evolución biológica natural de las especies (Farahani y Elahipanah, 2008; Kannan et al., 2009), han sido uno de los métodos más estudiados en la última década. Considerando la complejidad de la CS y la búsqueda de la optimización de costos y beneficios, se han desarrollado varias metaheurísticas, de las que destacan los Algoritmos Genéticos (AG), Enjambre de Partículas (PSO, por sus siglas en inglés), y Algoritmos Meméticos (AM), que han permitido encontrar soluciones próximas a la global. El Algoritmo Genético de Clasificación No dominado (NSGA-II, por sus siglas en inglés) es el AG más aplicado a los estudios de optimización de CS donde el costo total de la CS se encuentra en conflicto con otros objetivos como el "lead time" (Mendoza et al., 2015), el nivel de servicio del cliente (Farahani y Elahipanah, 2008), la distribución (Altiparmak et al., 2006; Zapata Cortés, 2016), los inventarios (Amodeo et al., 2008; Guerrero et al., 2016; Pinto, 2004), y CS con enfoque "just in time", JIT (Velasco Bermeo, 2009). Por otro lado, la optimización por PSO es compatible con NSGA-II; sin embargo, pocos estudios han optado por este método para la optimización de objetivos de la CS (Latha Shankar et al., 2013). Algunos de éstos, además de optimizar el costo de la cadena han optimizado la logística y logística inversa en CS cerradas (Kannan et al., 2009), problemas de programación (Xiong et al., 2018), y retrasos en las órdenes (Javanshir et al., 2012). En cuanto a la optimización por AM, se dice que presenta una mejora a los algoritmos anteriores en términos de calidad de resultados; también se le conoce como algoritmo híbrido debido a que se combinan dos o varios algoritmos para mejorar cualquiera de ellos. Los AM en CS se han empleado para la optimización de costos y objetivos como la capacidad de respuesta de la cadena (Pishvaee et al., 2010) y efectos ambientales (Jamshidi et al., 2012). Estos algoritmos han sido utilizados en muchos campos de aplicación y han dado respuesta a varios problemas de manufactura, logística y ensamblaje. Sin embargo, la mayoría de los casos mencionados consideran un solo

objetivo de optimización (Altiparmak et al., 2006; Jamshidi et al., 2012; Moscato y Cotta, 2003), y han sido pocos los trabajos que consideran múltiples objetivos, y menos aún en toda la CS, debido a la alta complejidad que conlleva probar múltiples objetivos en este entorno (Farahani y Elahipanah, 2008; Mendoza et al., 2015; Pishvaee et al., 2010; Zapata Cortés, 2016).

La investigación realizada propone contar con KPIs guiados en los modelos BSC y SCOR, que permitan optimizar varias funciones objetivo mediante la aplicación de tres algoritmos evolutivos de optimización. Estos algoritmos son probados con datos del caso de estudio de una empresa de ensamble de televisores, utilizando la herramienta PlatEMO, programada en Matlab para resolución de problemas multi-objetivo. Las siguientes secciones muestran una revisión de literatura relacionada con modelos y algoritmos de optimización basados en la evolución y afines a la CS. Se presenta además, una identificación de las variables y criterios de la CS que permiten formular dos funciones objetivo a optimizar, considerando que se buscó incrementar el valor de la empresa para los accionistas, lo cual se centra en la minimización de costos y el aumento de los ingresos, manteniendo su nivel de servicio alto (Sánchez Gómez, 2008). Se definió el entorno para aplicar la herramienta PlatEMO. Se realizó una discusión de los principales resultados y por último se realizó las respectivas conclusiones.

2. Revisión de literatura

En esta Sección se presenta una breve revisión de la optimización de problemas multiobjetivo y los algoritmos utilizados para la optimización del caso de estudio.

2.1. Problema de optimización multi-objetivo

Un problema multi-objetivo (MOP, por sus siglas en inglés) es aquel donde se da respuesta a varias incógnitas planteadas al mismo tiempo en un problema. La solución de un problema de optimización multi-objetivo, presenta un conjunto de puntos óptimos de solución, conocidos como Frontera de Pareto o Pareto Óptimo, que se apoya en un análisis de dominancia, en donde se puede encontrar las soluciones no-dominadas o dominantes. Una solución no-dominada es aquella en la que no existe otra solución que mejore una función objetivo sin perjudicar al menos una del resto de objetivos (Zapata Cortés, 2016). En la Imagen 1, se muestra una clasificación de la solución que puede ser la mejor, la pero, indiferente (ni dominante, ni dominada) con respecto a los valores objetivo (Coello Coello, 2005).

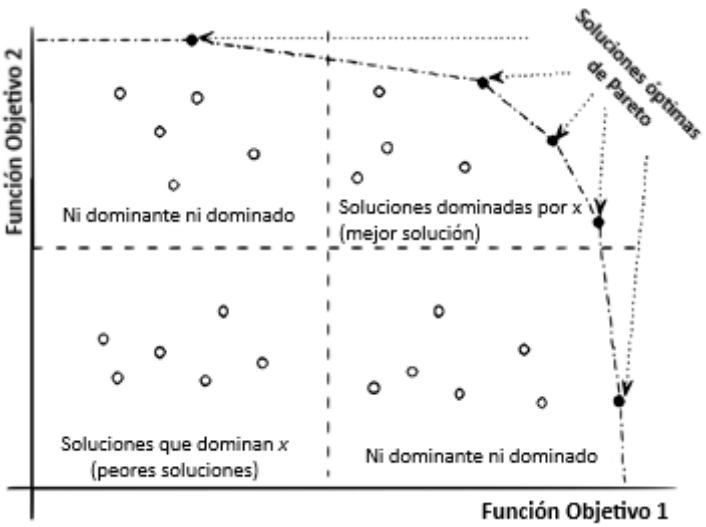

Imagen 1. Concepto de Pareto Óptimo. Fuente: Coello Coello (2005)

Por otro lado, de forma general un problema de optimización multi-objetivo incluye un conjunto de $\boldsymbol{n}$ variables de decisión, un conjunto de $\boldsymbol{k}$ funciones objetivo, y un conjunto de $\boldsymbol{m}$ restricciones. Donde, $\boldsymbol{x} = [\boldsymbol{x_1}, \boldsymbol{x_2}, \dots, \boldsymbol{x_n}]^{\boldsymbol{T}}$ es el vector de decisión, mientras que el vector objetivo de los espacios de decisión son X, y Y respectivamente, y $\boldsymbol{g_i}$ son las restricciones de las funciones del problema. Talavera et al. (2004) plantean el problema multi-objetivo según la ecuación (1):

$$\begin{aligned} &\text{Optimizar:} && y = f(x) = \big(f_1(x), f_2(x), \dots, f_k(x)\big) \\ &\text{Sujeto a:} && g_i(x) = \big(g_1(x), g_2(x), \dots, g_m(x)\big) \geq 0 \\ &\text{Donde:} && x = (x_1, x_2, \dots, x_n) \in X; \\ & && y = (y_1, y_2, \dots, y_n) \in Y \end{aligned} \qquad (1)$$

2.2. Algoritmos Evolutivos aplicados a la Cadena de Suministro

Principalmente, los Algoritmos Evolutivos Multi-objetivo (MOEA, por sus siglas en inglés) que han sido utilizados en CS son: algoritmo genético NSGA-II (Deb et al., 2002), optimización multi-objetivo por enjambre de partículas MOPSO (Kennedy y Eberhart, 1995), y algoritmo memético M-PAES (Knowles y Corne, 2000). A continuación, se hace una revisión de estos algoritmos.

2.2.1. Optimización por algoritmos genéticos

Los AG se fundamentan principalmente en la imitación de los fenómenos de adaptación y evolución de las especies, realizando procesos de transformación y selección simulando la genética natural (Ruiz Lizama, 2014). El algoritmo comienza con el planteamiento de una población inicial, que está conformada por individuos que compiten por la supervivencia; cada uno de los individuos representa una posible solución al problema de optimización. Luego de un proceso de evaluación se selecciona a los individuos más fuertes (padres), que tendrán la oportunidad de "procrear" nuevos individuos (hijos), a través de procesos de cruce y/o mutación. Los nuevos individuos heredarán características de sus padres o incluso podrán mejorarlas; de esta manera, la población inicial evoluciona, y se repite nuevamente el ciclo. Cada ciclo o bucle del algoritmo se lo conoce como generación (Konak et al., 2006). La evolución de la población ocurre principalmente por tres procesos que se conocen como operadores genéticos o naturales que son (Hernández Romero et al., 2012; Velasco Bermeo, 2009): *Selección,* este proceso es el encargado de escoger de entre una población los individuos más aptos para transmitir su código genético a futuras generaciones. Existe la selección directa, por torneo y proporcional (Hernández Romero et al., 2012). *El Cruce o Recombinación,* consiste en cruzar o "aparear" dos individuos (padres), con el fin de combinar su información genética para generar nuevos individuos (hijos). *La Mutación,* básicamente consiste en seleccionar un individuo al azar y cambiar uno o varios genes actuales por alelos (valor que toma un gen en una posición determinada) nuevos.

- *Evolución de los AG Multi-objetivo*

Schaffer (1984) presentó un AG llamado Vector Evaluated Genetic Algorithm (VEGA) o Algoritmo genético evaluado por vector, desafortunadamente esta técnica no resultó eficiente (Coello Coello, 2006). Luego, Goldberg (1989) planteó nuevos conceptos para tratar a los MOEA's para mejorar la calidad de la optimización, y fue quien por primera vez habló sobre la optimalidad de Pareto, métodos de clasificación y selección no-dominadas. La *primera generación* de los AG se caracteriza principalmente por la simplicidad de sus algoritmos y la falta de metodología para validarlos. Los más importantes son: Algoritmo Genético Multi-objetivo (MOGA), Algoritmo Genético de Nicho de Pareto (NPGA) y el Algoritmo Genético de Clasificación no dominado (NGSA). MOGA fue presentado por Fonseca y Fleming (1993), esta técnica describe el rango de un

individuo específico igual al número de cromosomas en la población por la que está dominado (Pinto, 2004). NPGA es un AG propuesto por Horn et al. (1994) que se enfoca en la selección por torneo basado en el dominio de Pareto. También, se desarrolló un "fitness sharing", que es una técnica para conservar la diversidad en los AG (Coello Coello et al., 2007b). NGSA desarrollado por Srinivas y Deb (1994), tiene la misma estructura del MOGA y se diferencia en la asignación de la función de aptitud. El NSGA consiste en que todos los individuos cerca del frente de Pareto tengan valores altos de aptitud. Se define una medida de distancia σ_{share}, para medir el grado en que los individuos afectan su función de aptitud (Coello Coello, 2006; Pinto, 2004).

La *segunda generación* de AG se caracterizó por la búsqueda de la eficiencia en la optimización. NSGA-II es uno de los AG desarrollados más recientes y fue presentado por Deb et al. (2002), se basa en una técnica de selección "elitista" y de "torneos atestados". El elitismo básicamente consiste en asegurar que los individuos con mejor valor de la función de adaptación continúen en las siguientes iteraciones con el fin de no perder el ajuste obtenido, y la selección por torneos atestados, donde el ganador de un torneo de individuos es juzgado en base al nivel que su función de ajuste aporte al torneo (Pinto, 2004). Deb et al. (2002) presentan el siguiente pseudocódigo para NSGA-II.

1. Crear población aleatoria inicial P0 (N individuos) y evaluar;
2. Crear una población de descendencia Qt, mediante cruce y mutación;
3. Mientras no se cumpla con el criterio de parada, hacer:
4. Unir las poblaciones Pt y Qt para crear la población Rt;
5. Realizar una clasificación no-dominada a Rt, e identificar mejores frentes no dominados Fi;
6. Hasta que el tamaño de Pt+1 $<$ N, hacer:
7. Calcular la distancia de aglomeración en Fi
8. Ordenar Fi de forma descendente
9. Actualizar Pt+1 (Pt+1 = Pt+1 $\cup$ Fi), se asignan los mejores elementos de Fi
10. Aplicar operadores genéticos a Pt+1 para crear nueva población Qt+1

2.2.2. Optimización por Enjambre de Partículas

La Optimización por enjambre de partículas (PSO, por sus siglas en inglés) es una técnica de cálculo evolutivo presentada por Kennedy y Eberhart (1995). Este algoritmo de búsqueda está basado en la población y la simulación de la interacción o comportamiento social y de agrupamiento de las aves y peces (Reyes-Sierra y Coello Coello, 2006). La diferencia principal de PSO con AG es que no busca la supervivencia del más apto por lo que no adopta un proceso de selección de individuos; sin embargo, se asemeja a AG porque trabaja en base a una población, la cual se evalúa mediante una o varias funciones de ajuste, para luego actualizar la población y encontrar la solución óptima (Kao y Zahara, 2008). Se inicia el algoritmo con un grupo de partículas aleatorias, donde cada individuo es tratado como una partícula sin volumen en un espacio de búsqueda multidimensional y a una velocidad aleatoria; entonces, un conjunto de partículas son "lanzadas" dentro de un espacio de búsqueda con una posición y velocidad, en el cual cada partícula conoce y recuerda su posición así como el valor de la función objetivo en esa posición, también su mejor posición anterior y la de sus vecinos. De esta manera, todas las partículas ajustan constantemente su dirección y velocidad de búsqueda según las dos mejores posiciones dependiendo del algoritmo (Kannan et al., 2009).

Los principales términos de PSO presentados por Reyes-Sierra y Coello Coello (2006), son: *Velocidad (v),* un vector que controla la dirección en la que la partícula debe moverse "volar" para mejorar su posición. *Peso de inercia (W),* se emplea para controlar el impacto del historial anterior

de velocidades en la velocidad actual de una partícula. *Factor de aprendizaje*, son dos constantes: C1, es factor de aprendizaje cognitivo y es la atracción que una partícula tiene hacia su propio éxito y C2, el aprendizaje social y es la atracción de una partícula hacia sus vecinos. Mesa Alvarez (2017) describe un pseudocódigo para resolver un problema de optimización por enjambre de partículas:

1. Inicializar enjambre, velocidades y mejores posiciones;
2. Inicializar registro externo (inicialmente vacío);
3. Mientras no se satisfaga el criterio de parada (número de iteraciones o variación entre la solución actual y la anterior), hacer:
4. Para cada partícula, hacer:
5. Seleccionar un miembro del registro externo (si es necesario);
6. Actualización de velocidad y posición;
7. Evaluar nueva posición:
8. Actualización de mejor posición y del registro externo

2.2.3. Optimización por Algoritmos Meméticos

Los Algoritmos Meméticos (AM) tienen su origen a finales de los años ochenta, cuando la computación evolutiva se encontraba en auge y técnicas metaheurísticas para la optimización eran utilizadas y estudiadas. Los algoritmos meméticos surgen de la combinación de conceptos y estrategias de distintas metaheurísticas para mejorar el desempeño de éstas. El término "meme" es análogo al término gen de los algoritmos genéticos y se le atribuye a R. Dawkins (Moscato y Cotta, 2003). Los AM son de naturaleza poblacional ya que han heredado características de los algoritmos evolutivos (Moscato y Cotta, 2003). Corne et al. (2000) desarrollaron el algoritmo memético de Pareto con Estrategia Evolutiva Archivada (M-PAES). Este algoritmo utiliza un archivo externo, que genera un nuevo hijo con un operador Gaussiano mutado y selecciona la siguiente generación basada en una comparación no dominada de los padres e hijos evaluados en buen estado (Coello Coello et al., 2007a).

A continuación se presenta una comparación de los aspectos más distintivos de los algoritmos genéticos y meméticos (Tabla 1).

Tabla 1. Comparación de los aspectos más distintivos de AG y AM

Algoritmos Genéticos (AG)	Algoritmos Meméticos (AM)
Codificación, esquemas, cadenas lineales, alfabetos predefinidos	***Representación,*** formas, no-linealidad, cercanía al problema
Individuo, solución al problema	***Agente,*** solución/es al problema más mecanismo de mejora local
Cruce, intercambio no-guiado de información	***Recombinación***, intercambio guiado de información
Mutación, introducción aleatoria de nueva información	***Mutación,*** introducción sensible de nueva información
No hay mejora local	***Mejora local***, aprendizaje lamarckiano

Fuente: (Moscato y Cotta, 2003)

Moscato y Cotta (2003) hablan sobre los operadores reproductivos de los algoritmos meméticos que son: *Recombinación,* donde los agentes seleccionados construyen agentes nuevos con la información o características extraídas del grupo de agentes recombinados de la nueva población en cada generación. *Mutación,* donde al proceso de modificación de un agente existente con este tipo de información se le conoce como mutación. *Mejora local,* el algoritmo explora de manera autónoma el espacio de búsqueda para mejorar el agente o resultado.

Knowles y Corne (2000) presentan el siguiente pseudocódigo de M-PAES:

1. Generar una población inicial P de n soluciones aleatorias y evaluar;
2. Ubicar cada miembro no-dominado de P en un archivo global G;
3. Mientras criterio de parada no se satisface, hacer;
4. Para cada solución c en P, hacer;
5. Crear un archivo local H, inicialmente en 0;
6. Llenar el archivo H con soluciones de G que no dominen c;
7. Copiar la solución c desde P a H;
8. Ejecutar la búsqueda local usando el proceso PAES (c, G,H);
9. Reemplazar las soluciones mejoradas de c en P;
10. Crear una población intermedia (P') y un ni (inicialmente en 0);
11. Mientras ni < n, hacer:
12. Poner el número de recombinaciones de prueba r=0;
13. Mientras (((c es dominado por G) ∨ (c está en la locación más llena de la malla))∧(r<recomb_trials_max)), hacer:
14. Aleatoriamente seleccionar 2 pares de P∪G y recombinar para formar la descendencia c;
15. Comparar c con las soluciones en G;
16. Cambiar G con c si es necesario;
17. Si c es dominado por G, descartar c y usar torneo binario para seleccionar una nueva solución c de G;
18. Colocar c en la población intermedia P';
19. Actualizar la población P con la población intermedia P'.

3. Planteamiento del problema de Optimización

3.1. Funciones objetivo y restricciones usadas para la optimización de la CS

La presente investigación utiliza tres diferentes algoritmos de optimización que fueron evaluados usando los datos de un caso de estudio de una empresa de ensamblaje de televisores ubicada en Cuenca - Ecuador, no se menciona el nombre de dicha empresa por términos de confidencialidad; y por el mismo motivo, no se presenta todos los datos de entrada del problema de optimización. La CS del caso de estudio, está formada por tres niveles (proveedores, planta de ensamblaje, y clientes) y dos etapas (producción y distribución). Debido a la situación actual de la empresa, se busca determinar la mejor opción de producción y distribución que logre cumplir con la demanda de los clientes bajo las restricciones de capacidad propias de cada nivel. Además, optimizar los niveles de inventarios que mantiene la empresa al final de cada periodo, en especial el inventario de seguridad. La CS es servida por seis proveedores (S1-S6) que suministran más de 170 ítems. Tiene una planta de ensamblaje de televisores (P1), que para el periodo de estudio 2017 ensambló 14 modelos de televisores. La distribución se limita hacia cinco centros de distribución (CD) que representan el 82% de las ventas anuales (C1-C5) (Figura 1).

Para el planteamiento del problema contamos con los siguientes supuestos:

a. Se conoce el número de proveedores, plantas y centros de distribución.
b. La demanda de cada centro de distribución es conocida.
c. Se conoce la capacidad de planta.
d. La capacidad de abastecimiento de los proveedores se considera infinita; es decir, que los proveedores pueden atender las demandas de la planta por completo. Además, en la etapa de abastecimiento no hay selección de proveedores por lo que se puede simplificar este término en la función objetivo.
e. Los clientes reciben productos de un solo centro de distribución.

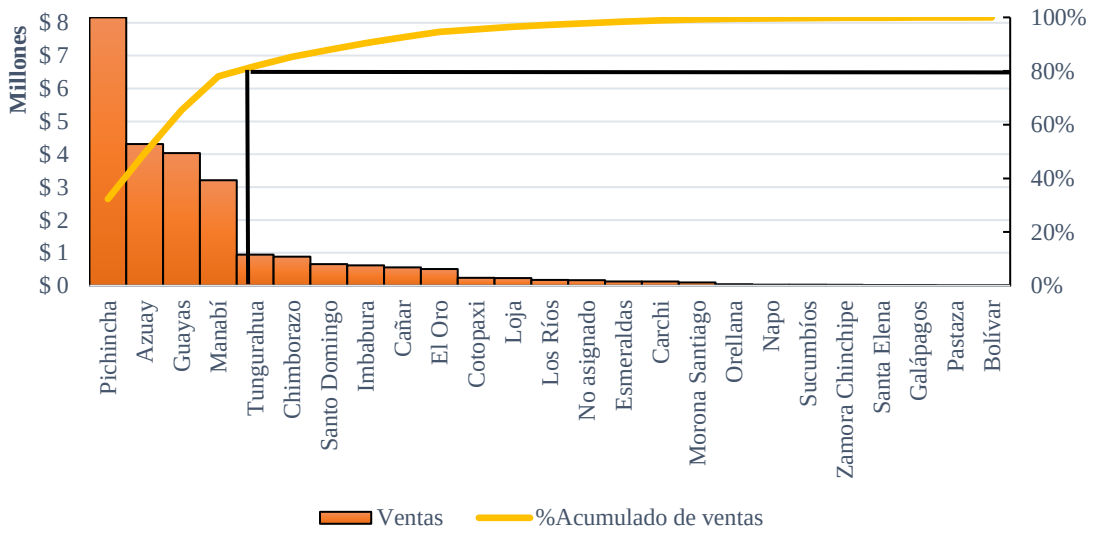

Figura 1. Total de ventas anuales por mercado. Fuente: Elaboración Propia.

Por otro lado, en la Tabla 2 se muestran los índices utilizados para identificar productos y las partes o niveles del problema planteado.

Tabla 2. Notación de índices utilizados en el modelo matemático

Índice	Significado
b	Productos
j	Plantas
k	Mercados
z	Periodos

En la Tabla 3 se indican los datos de entrada, que resultan del levantamiento de información en la empresa, y correspondientes a los costos asociados a la CS y las restricciones de capacidad del sistema.

Tabla 3. Notación de los datos del modelo matemático

Notación	Significado
CM_{bjz}	Costo unitario de materia prima abastecida para ensamblar *b* en la planta *j*, en el periodo *z*
CE_{bjz}	Costo de ensamblaje (conversión) del producto *b* en la planta *j*, en el periodo *z*
CT_{bkjz}	Costo unitario de transporte de producto *b* desde la planta *j* hasta el mercado *k*, en el periodo *z*
CI_{bjz}	Costo de unitario de mantenimiento inventario de producto *b* en la planta *j*, en el periodo *z*
PV_{bk}	Precio de venta del producto *b* en el mercado *k*
D_{bkz}	Demanda del producto *b* del cliente *k*, en el periodo *z*
I_{b0}	Nivel de inventario inicial del producto *b*, en el periodo z=0
pt_{bj}	Tiempo de requerido para producir una unidad de *b* en planta *j*
tt_{jz}	Tiempo disponible para producir en la planta *j* por periodo *z*
SS_{bjz}	Stock de seguridad del producto *b* en la planta *j*, en el periodo *z*

En la Tabla 4 se muestra la notación para las variables de decisión o de salida del problema de optimización propuesto.

Tabla 4. Notación de las variables de decisión usadas en el modelo matemático

Variable	Significado
P_{zjb}	Cantidad total de producto *b* ensamblado en la planta *j*, en el periodo *z*.
PD_{zjkb}	Cantidad de producto *b* despachado desde la planta *j* hasta el mercado *k*, en el periodo *z*.
I_{zjb}	Cantidad de inventario final de producto *b* en la planta *j*, en el periodo *z*.

Con los datos planteados en las Tablas 2 - 4, se pueden generar las funciones objetivos para la optimización de la CS. Éstas se describen en las ecuaciones (2) y (4):

Objetivo 1: Maximizar la utilidad (Max U)

$$\boldsymbol{Max\ U} = \sum_{z}\sum_{j}\sum_{k}\sum_{b}(PD_{zjkb} * PV_b) - CT \tag{2}$$

Donde:

$$\boldsymbol{CT} = \sum_{z}\sum_{j}\sum_{b} P_{zjb} * (CM_{zjib} + CE_{jzb}) + \sum_{z}\sum_{j}\sum_{k}\sum_{b}(PD_{zjkb} * CT_{zjkb}) + \sum_{z}\sum_{j}\sum_{b}(I_{zjb} * CI_{zjb}) \tag{3}$$

Objetivo 2: Maximizar el nivel de servicio al cliente (Max NS)

$$\boldsymbol{Max\ NS} = \frac{\sum_z \sum_j \sum_k \sum_b PD_{zjkb}}{\sum_z \sum_k \sum_b D_{zkb}} \tag{4}$$

Las ecuaciones (2) y (4) tienen el objetivo de maximizar la utilidad de la CS y maximizar el nivel de servicio al cliente. La ecuación (2) está formada por la diferencia entre las ventas (primer término) y el costo total (CT), segundo término; a su vez, el costo total (3) se conforma por el costo total de materia prima más el costo de ensamblaje; el costo de distribución y transporte de los productos; y, el costo de mantenimiento del inventario al final de un periodo en la planta. Mientras que la ecuación (4) representa el nivel de servicio al cliente como una relación entre los productos distribuidos y su demanda. El problema por optimizar requiere de niveles de servicio al cliente altos o mayores al 90%. Por otro lado, las restricciones que se identificaron para el caso de estudio se presentan a continuación en las ecuaciones 5 – 9.

Restricciones:

$$\sum_{b} pt_{jb} * P_{zjb} \leq tt_{zj} \quad ; \ \forall\, j, z \tag{5}$$

$$\sum_{j} PD_{zjkb} \leq D_{zkb}\ ; \ \forall\, z, k, b \tag{6}$$

$$I_{zjb} = I_{(z-1)jb} + P_{zjb} - \sum_{k} PD_{zjkb} \; ; \; \forall j, b, z \quad (7)$$

$$I_{zjb} \geq SS_{bjz} \quad (8)$$

$$P_{zjb}, PD_{zjkb}, I_{zjb} \geq 0 \quad (9)$$

La inecuación (5) garantiza que el tiempo total requerido para producir los productos no exceda el tiempo total disponible de la planta. La inecuación (6) limita a la cantidad de productos enviados en el periodo para que no sea mayor a la demanda. La restricción (7) es una ecuación de balance de inventario final en la planta. La restricción (8) es la restricción que indica el inventario mínimo que se debe mantener en planta a final de cada periodo. Finalmente, la restricción (9) indica la no negatividad de las variables de decisión, las cuales tienen que tomar un valor mayor o igual a 0.

3.2. Datos de entrada para el modelo de optimización en la CS

La información utilizada para el presente estudio corresponde al periodo 2017, y se obtuvo de los trabajos realizados por Guamán Ochoa et al. (2020) y Guerrero (2018) . En la Tabla 5 se presentan los costos unitarios de los productos de la empresa para el periodo 2017 y su respectivo tiempo de ensamblaje.

Tabla 5. Tiempo y costos unitarios, por modelo de televisor

Nro.	Producto	Costo de materia prima CM_{bjz}	Costo de ensamblaje CE_{bjz}	Costo de mantener inventario CI_{bjz}	Costo de transporte CT_{bkjz}	Tiempo de ensamblaje pt_{bj} (min)
1	Modelo 1	$ 358,59	$ 14,77	$ 6,29	$ 4,00	12,058
2	Modelo 2	$ 522,93	$ 15,20	$ 6,29	$ 5,00	12,435
3	Modelo 3	$ 446,95	$ 12,82	$ 6,29	$ 5,00	12,435
4	Modelo 4	$ 174,44	$ 10,83	$ 6,29	$ 3,00	8,272
5	Modelo 5	$ 269,41	$ 17,08	$ 6,29	$ 3,00	8,272
6	Modelo 6	$ 349,17	$ 15,51	$ 6,29	$ 3,00	12,167
7	Modelo 7	$ 401,06	$ 15,13	$ 6,29	$ 5,00	12,167
8	Modelo 8	$ 758,89	$ 25,23	$ 6,29	$ 7,00	18,426
9	Modelo 9	$ 965,05	$ 32,38	$ 6,29	$ 7,00	24,517
10	Modelo 10	$ 297,14	$ 20,05	$ 6,29	$ 3,00	18,426
11	Modelo 11	$ 402,99	$ 13,51	$ 6,29	$ 4,00	12,435
12	Modelo 12	$ 529,38	$ 15,58	$ 6,29	$ 4,00	18,519
13	Modelo 13	$ 604,26	$ 17,23	$ 6,29	$ 5,00	18,519
14	Modelo 14	$ 268,43	$ 8,33	$ 6,29	$ 3,00	9,728

Fuente: Guamán Ochoa et al. (2020) y Guerrero (2018)

Otros datos de entrada para el modelo de optimización son D_{bkz} y PV_{bk} como se indica en la Tabla 3; sin embargo no se presentan debido a términos de confidencialidad con la empresa.

Además, se requiere mantener un inventario de seguridad (SS_{bjz}), que se calcula mediante la desviación estándar del producto (σ) y el número de desviaciones estándar (Z), ecuación (10). Para esto se considera como tiempo de reposición un mes, y el inventario al inicio de cada periodo I_{b0}, como cero (0) debido a la limitación en la información de aprovisionamiento.

$$SS = Z * \sigma_{dLT} \quad (10)$$

Generalmente se busca obtener un nivel de servicio del 95% por lo que para este valor se obtiene un número de desviaciones estándar (Z) de 1,645.

Por otro lado, dentro de la investigación, se consideran dos escenarios con el inventario de seguridad; uno con un inventario mínimo de seguridad y otro con un inventario máximo de seguridad; para esto se utiliza un "buffer" que también funciona como amortiguador, el cual permite controlar los inventarios en zonas presentados en la Tabla 6 (Cuesta et al., 2020). Estos escenarios permiten analizar las posibles variaciones que se den en la CS y cuál es su impacto en los objetivos de optimización planteados. El primer escenario (color verde), es la parte del nivel de inventario en el que se podrá consumir los productos necesarios para cumplir con la demanda y debe ser mayor a 2/3 del Nivel Objetivo de Inventario (NOI), que para el caso de estudio se calcula con la ecuación (10), mientras que el segundo escenario (color rojo), representa una alarma, la cual indica que se debe producir para rellenar el inventario de tal forma que este no se consuma totalmente. Este inventario es menor a 1/3 del NOI.

Tabla 6. Niveles de inventario de seguridad para los dos escenarios de análisis

Nro.	Modelo	NOI	Mínimo 1/3 del NOI	Máximo 2/3 del NOI
1	Modelo 1	815	272	543
2	Modelo 2	0	0	0
3	Modelo 3	878	293	585
4	Modelo 4	1518	506	1012
5	Modelo 5	0	0	0
6	Modelo 6	274	91	183
7	Modelo 7	344	115	229
8	Modelo 8	68	23	45
9	Modelo 9	44	15	30
10	Modelo 10	313	104	209
11	Modelo 11	506	169	337
12	Modelo 12	36	12	24
13	Modelo 13	45	15	30
14	Modelo 14	268	89	179

Nota: NOI= Nivel óptimo de inventario

Fuente: Elaboración propia.

3.3. Herramienta PlatEMO para optimización de problemas Multi-objetivo

PlatEMO es una plataforma de código abierto de MATLAB para la optimización evolutiva de múltiples objetivos. El objetivo de los algoritmos multi-objetivo (MOEA) en PlatEMO, es hacer que la población se aproxime al conjunto de Pareto con buena convergencia y diversidad (Tian et al., 2017). Para utilizar la herramienta, se debe conocer previamente el tamaño de la población, número de objetivos y variables de decisión, que son variables de ingreso para el programa. Además, el algoritmo a utilizar debe ser seleccionado previamente así como sus parámetros (Tian et al., 2017) .

3.3.1. Generación del problema en PlatEMO

La herramienta PlatEMO requiere el uso de un archivo .m que defina el problema, éste va ubicado en la carpeta de problemas del programa. Aquí se pueden realizar las modificaciones necesarias para el problema planteado en este estudio. La estructura resumida del archivo .m generado es la siguiente:

```
%% Inicialización
    obj.Global.lower    = valores mínimos de las variables independientes
    obj.Global.upper    = valores máximos de las variables independientes
%% Cálculo de valor de las funciones objetivo
    Function PopObj = CalObj (obj, X)
   Introducción de las variables a ser usadas, en este caso P, PD e I.
    PopObj (:,1)=introducción de la FO1
    PopObj (:,2)= introducción de la FO2
%% Cálculo de Restricciones
    Function PopCon = CalCon (obj, X)
   Introducción de las variables a ser usadas, en este caso P, PD y I.
   Introducción de las restricciones en la variable PopCon (: i)
```

PlatEMO es una plataforma utilizada especialmente para la minimización, entonces, se ingresa las funciones objetivo de maximización con un *signo negativo*. Este signo permite transformar a objetivos de minimización y trabajar normalmente con el problema presentado. Para el ingreso de las funciones objetivo se utiliza el vector PopObj, en el cual se expresa de manera lineal las funciones objetivo de maximización. La restricción de capacidad de planta se presenta en la ecuación (5) y es interpretada en la ecuación (11) para su ingreso correcto en la herramienta. Esto para los 12 periodos de estudio (12 restricciones).

$$PopCon = pt_{jb} x P_{zjb} - tt_{zj} \quad (11)$$

En el programa, se empieza definiendo el número de variables que se utilizan para el problema de estudio. En este caso son 1176. Este número se obtiene por el uso de las tres variables del problema que son:

- PD = PD_{zjkb} , 840 variables independientes.
- I = I_{zjb} , 168 variables independientes.
- P = P_{zjb}, 168 variables dependientes.

3.3.2. Determinación del número de iteraciones y parámetros para los algoritmos

Determinar el número de iteraciones o evaluaciones en la utilización de un algoritmo es crucial ya que permite aminorar el costo computacional sin descuidar el desempeño normal del algoritmo. Para este estudio se han probado 10,000 y 100,000 iteraciones.

Para la optimización mediante NSGA-II, las recomendaciones de Reed, Minsker y Goldberg (2000), han sido tomadas y se presentan en la Tabla 7.

Tabla 7. Parámetros del NSGA-II

Parámetro	Valor
Tamaño de la población (N)	100
Probabilidad de cruce (Pc)	0,9
Probabilidad de mutación (Pm)	0,01

En cuanto al algoritmo MOPSO, los parámetros se basan en la propuesta de los autores Coello Coello et al. (2004), Javanshir et al. (2012), Reyes-Sierra y Coello Coello (2006), mostrados en la Tabla 8. En este algoritmo aumentar las iteraciones no mejoró significantemente sus resultados.

Tabla 8. Parámetros del algoritmo MOPSO

Parámetro	Valor
Población Inicial	100
Peso de inercia W	0,4
Coeficiente de aprendizaje local, C1	1
Coeficiente de aprendizaje global, C2	1

Finalmente, para el algoritmo memético (M-PAES), con el incremento de número de iteraciones el resultado en función del beneficio (f1) y nivel de servicio (f2) mejora considerablemente, y se reduce el tamaño de la población cuando inicialmente se había indicado N=100. Los parámetros para considerar la implementación del algoritmo se basan en recomendaciones de los autores (Knowles y Corne, 2000) y se indican en la Tabla **9**.

Tabla 9. Parámetros del algoritmo M-PAES

Parámetro	Símbolo	Valor
Número máximo de fallos consecutivos de movimientos locales	L_fails	5
Número máximo de movimientos locales	L_opt	10
Número de cruzamientos	C_trials	20

- L_fails, número de movimientos locales que una partícula realiza dentro de su población, se requiere un menor número de fallos para un mejor resultado.
- L_opt, número máximo de movimientos locales para obtener un conjunto de resultados.
- C_trial, número de cruzamientos permite tener un resultado global más general entre todos los vecinos de cada partícula.

En la Figura 2 se presenta un ejemplo de los datos visualizados en el programa PlatEMO para el algoritmo NSGA-II con 100,000 evaluaciones y para el caso de inventario mínimo.

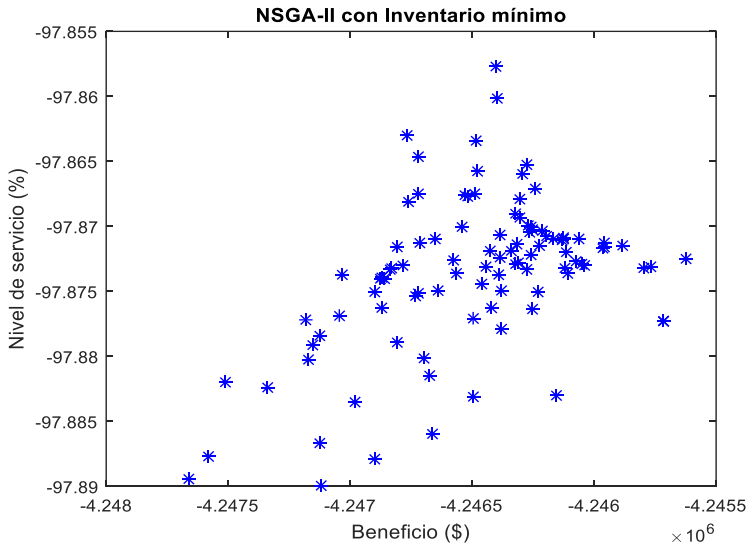

Figura 2. Simulación en PlatEMO con NSGA-II. Fuente: Elaboración propia

4. Resultados

Con base en los datos y las funciones objetivos de la Sección 2, se presentan los resultados obtenidos por los tres algoritmos con inventario de seguridad mínimo y máximo, y con un número de iteraciones de 10,000 y 100,000. En la Tabla 10 y
Tabla **11** se presentan el tiempo de ejecución de cada algoritmo y los valores obtenidos de las funciones objetivo.

Tabla 10. Resultados obtenidos de los tres algoritmos con inventario mínimo

10,000 It.	**F OBJ. 1**	**F OBJ. 2**	**TIEMPO (s)**
NSGA-II	$ 3,624,514.87	86.18	11.26
MOPSO	**$ 3,728,841.21**	**88.75**	**7.84**
M-PAES	$ 2,905,654.15	72.10	129.48
100,000 It.			
NSGA-II	**$ 4,272,669.41**	**97.87**	**89.94**
MOPSO	$ 3,881,112.75	90.33	55.61
M-PAES	$ 4,185,197.22	95.26	420

Nota: Función objetivo 1= FOBJ1, Función objetivo 2= FOBJ2, Iteraciones = It.

Tabla 11. Resultados obtenidos de los tres algoritmos con inventario máximo

10,000 It	**F OBJ. 1**	**F OBJ. 2**	**TIEMPO (s)**
NSGA-II	$ 2,856,451.52	85.26	9.55
MOPSO	**$ 3,074,152.51**	**87.47**	**8.19**
M-PAES	$ 2,135,499.08	70.31	48.68
100,000 It			
NSGA-II	**$ 3,576,619.52**	**97.73**	**75.95**
MOPSO	$ 3,271,055.75	91.93	61.33
M-PAES	$ 3,475,744.91	95.49	385

Nota: Función objetivo 1= FOBJ1, Función objetivo 2= FOBJ2, Iteraciones = It.

En las Tablas 10 y 11 se muestra como tendencia general que a mayor número de iteraciones, mejor será el nivel de respuesta, pero también mayor tiempo de procesamiento computacional. Esto se cumple para los tres algoritmos. En la Tabla 12 y
Tabla **13** se presentan las demandas anuales obtenidas por cada uno de los tres algoritmos con cada tipo de inventario y se busca compararlas con la demanda actual (periodo 2017).

Tabla 12. Demanda anual para inventario mínimo, periodo 2017

Nro.	Modelo	Inventario mínimo			
		Actual	NSGA-II	MOPSO	M-PAES
1	Modelo 1	8626	8500	8002	8431
2	Modelo 2	387	352	188	339
3	Modelo 3	6509	6443	5948	6331
4	Modelo 4	14261	14120	13349	13697
5	Modelo 5	323	301	305	301
6	Modelo 6	4562	4419	3966	4199
7	Modelo 7	4563	4442	4108	4437
8	Modelo 8	683	598	553	642
9	Modelo 9	388	337	304	357
10	Modelo 10	4052	3954	3568	3822
11	Modelo 11	3426	3338	3167	3073
12	Modelo 12	317	284	218	236
13	Modelo 13	360	318	234	310
14	Modelo 14	2455	2430	2080	2342
Ventas		$ 22,674,134.42	$ 22,086,178.98	$ 20,293,175.17	$ 1,593,097.83
Costo		$ 18,288,272.76	$ 17,813,509.58	$ 16,412,062.42	$17,407,900.60
Beneficio		$ 4,385,861.66	$ 4,272,669.41	$ 3,881,112.75	$ 4,185,197.22
Nivel Servicio		100%	97.87%	90.33%	95.26%

Tabla 13. Demanda anual para inventario máximo, periodo 2017

Nro.	Modelo	Inventario máximo			
		Actual	NSGA-II	MOPSO	M-PAES
1	Modelo 1	8626	8541	8275	8454
2	Modelo 2	387	345	314	327
3	Modelo 3	6509	6422	6115	6280
4	Modelo 4	14261	14073	13735	13900
5	Modelo 5	323	284	16	288
6	Modelo 6	4562	4428	4305	4253
7	Modelo 7	4563	4466	4279	4244
8	Modelo 8	683	632	564	644
9	Modelo 9	388	332	287	357
10	Modelo 10	4052	3937	3385	3765
11	Modelo 11	3426	3341	2707	3228
12	Modelo 12	317	246	257	268
13	Modelo 13	360	341	299	313
14	Modelo 14	2455	2383	2266	2298
Ventas		$ 22,674,134.42	$ 22,091,409.99	$ 20,677,482.06	$ 1,600,226.20
Costo		$ 18,992,079.71	$ 18,514,790.47	$ 17,406,426.31	$18,124,481.29
Beneficio		$ 3,682,054.71	$ 3,576,619.52	$ 3,271,055.75	$ 3,475,744.91
Nivel. Servicio		100%	97.73%	91.93%	95.49%

Se puede notar que, a un mayor nivel de inventario de seguridad, se necesita un gasto extra en el costo de mantenimiento de inventario por lo que disminuye el beneficio de la empresa. El algoritmo

MOPSO es el que presenta el menor beneficio, seguido de M-PAES con una diferencia mínima de aumento en su beneficio y por último NSGA-II que presenta el mayor beneficio de entre los tres algoritmos. Por otro lado, con 100,000 evaluaciones y para los dos niveles de inventario, el MOPSO presenta el nivel de servicio más bajo entre los tres casos, con un 90.33 y 91.93% respectivamente, seguido de M-PAES con un 95.26 y 95.49% y culminando con NSGA-II que presenta el mejor nivel de servicio de 97.87 y 97.73% respectivamente, presentado en la Figura 3.

5. Discusión

En la investigación llevada a cabo, dos escenarios para la optimización fueron planteados. Estos dos escenarios son un favorable y un pesimista. Como se observa en la Figura 3, el comportamiento del nivel de servicio con relación al beneficio según los diferentes escenarios muestra una mejora según el algoritmo que se utilice. El algoritmo NSGA-II es el que se desempeña de mejor manera bajo las condiciones establecidas, esto coincide con lo que Mendoza et al. (2015) mencionan en su trabajo, donde el NSGA-II es el algoritmo más utilizado para optimización multiobjetivo. Sin embargo, estos autores compararon sus resultados solo con un algoritmo, mientras que en la investigación propuesta lo realiza con tres algoritmos.

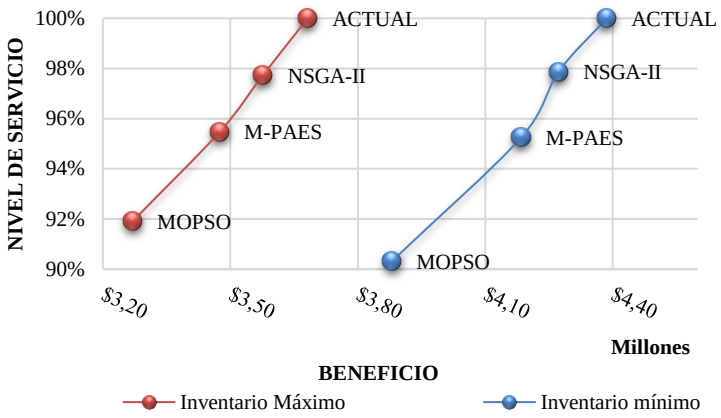

Figura 3. Resultados de los algoritmos. Fuente: Elaboración propia

Por otro lado, al comparar los tres algoritmos con el número de iteraciones, se puede observar que el comportamiento de NSGA-II y M-PAES es exponencial, por lo que se necesita un gran aumento de iteraciones para obtener un mínimo aumento en el nivel de servicio (Figura 4). Y aunque MOPSO a menor número de iteraciones presenta mejores resultados que los otros dos algoritmos, se observa que mientras se incrementa el número de iteraciones, la curva de MOPSO se mueve muy lentamente en el espacio de posibles soluciones e incluso tiende a estancarse en óptimos locales, por lo que los otros dos algoritmos lo superan. Se conoce que M-PAES es un algoritmo híbrido complejo que toma mayor tiempo presentar soluciones, pero éste ha resultado ser mejor en términos de calidad de resultados en comparación con MOPSO; pero, para M-PAES estas soluciones mejoran al incrementar el número de individuos en la población, lo cual difiere de los otros dos algoritmos donde incrementar el número de individuos afecta a la velocidad de convergencia vs. número de iteraciones. Por lo mencionado, 100.000 iteraciones propuestas son suficientes para obtener resultados favorables y para poder comparar los tres algoritmos en términos comunes; además, el costo computacional se puede evaluar bajo las mismas condiciones.

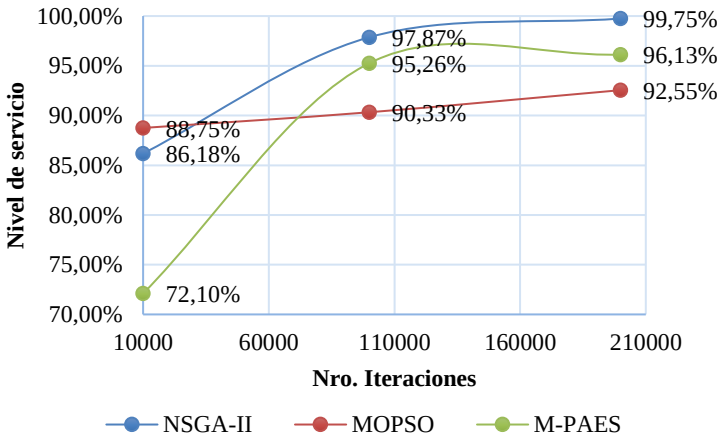

Figura 4. Número de iteraciones por algoritmo. Fuente: Elaboración propia

El problema de optimización en CS ha sido estudiado por varios autores dando como resultados diferentes respuestas a preguntas planteadas u objetivos planteados. Los resultados de esta investigación se comparan con otros encontrados en la bibliografía, por ejemplo, con Babaveisi et al. (2018), Javanshir et al. (2012) y Kannan et al. (2009), quienes coinciden en que la optimización por algoritmo genético (AG) da mejores resultados que PSO en la optimización bi-objetivo de la cadena de suministro; y a su vez están de acuerdo en que el algoritmo MOPSO presenta soluciones en menos tiempo que otros algoritmos, al igual que nuestro problema de optimización. Mientras que el algoritmo memético, M-PAES, en nuestro problema de optimización se toma el mayor tiempo en presentar una respuesta de calidad, en otros problemas de CS como en Jamshidi et al. (2012) y Pishvaee et al. (2010), y otros estudios similares (Karaoglan y Altiparmak, 2015; Moshref-Javadi y Lee, 2016), ha presentado buenos resultados como algoritmo híbrido. En nuestro caso de estudio, pese a no ser un algoritmo muy utilizado en la cadena de suministro resultó presentar mejores resultados que MOPSO, pero menores resultados que el NSGA-II.

La CS estudiada en la investigación se la ha limitado al caso de ensamble y distribución ya que en una empresa de ensamblaje estas dos áreas son críticas, para este estudio los proveedores están en la capacidad de poder surtir de material necesario a la línea de ensamble. Estas limitaciones no lo han tenido otros trabajos que consideran CS de hasta cuatro etapas con cinco niveles (proveedor, productor, distribuidor, mayorista y minorista). Sin embargo, estos estudios de caso como Babaveisi et al. (2018), Guerrero et al. (2016), Kannan et al. (2009), Latha Shankar et al. (2013) y Zapata Cortés (2016), por mencionar algunos, trabajan con pocos productos y pocos periodos de estudio. Lo que diferencia a nuestro caso de optimización de ensamblaje de televisores es que, aunque la CS es "sencilla", el número de variables que maneja el problema hace que incremente su complejidad de resolución; así tenemos un total de 1176 variables, debido a que consideramos 12 periodos de estudio, 1 planta de ensamblaje, 14 productos y 5 centros de distribución. Esta complejidad hace que los algoritmos de optimización muestren su eficiencia llevándonos a encontrar resultados óptimos. Por otro lado, la cercanía de los conjuntos de solución obtenidos por el algoritmo genético NSGA-II, es mucho mayor que la de los conjuntos encontrados por los algoritmos MOPSO y M-PAES, para los dos escenarios de estudio, nivel mínimo y máximo de inventario de seguridad.

6. Conclusiones

En esta investigación, se alcanzó la optimización multi-objetivo de costos en la cadena de suministros de la empresa de ensamble de televisores mediante la aplicación y comparación de tres algoritmos evolutivos, en donde el algoritmo que mejor desempeño tuvo fue NSGA-II. Además, esta

investigación aportó a identificar las variables y criterios de la CS para empresas de ensamblaje que se ven involucradas en una optimización (Tabla 2 – 4). En la CS, las funciones objetivo de maximización de la utilidad y servicio al cliente fueron probadas con los algoritmos NSGA-II, MOPSO y M-PAES, a través de la herramienta PlatEMO, y las cuales se presentaron en las ecuaciones (2) y (4) respectivamente.

Los resultados dan evidencia que el número de iteraciones afecta a algunos algoritmos y a otros no, y la optimización multi-objetivo muestra que NSGA-II tiene mayor efectividad que MOPSO y M-PAES a la hora de buscar en el espacio de soluciones factibles un conjunto de soluciones que presenten vectores objetivo muy cercanos o contenidos en el conjunto real del óptimo de Pareto. Así, un valor de 97.87% de nivel de servicio con un beneficio de $4,272,669.41 fue alcanzado por este algoritmo, este valor representa un incremento de $391,556.66 del peor resultado dado por MOPSO; siendo el mejor escenario de optimización el nivel mínimo de inventario de seguridad.

Una de las limitaciones presentadas en esta investigación, fue tratar a la CS desde proveedores considerando que estos eran capaces de proveer todos los requerimientos de la planta. Sin embargo, en muchas CS la etapa de aprovisionamiento de materias primas es considerara estratégica y presenta varios retos, ya que maneja ventanas de tiempos distintas según la materia prima y la localización del proveedor. Este estudio se puede ampliar considerando el nivel del inventario en proceso de la planta de ensamblaje, ya que estos inventarios conllevan altos costos en algunas partes de la CS. Por otro lado, el estudio de los algoritmos de optimización, se puede complementar con un análisis de sensibilidad de variables usando el problema dual, que permita conocer que variables son más importantes.

Se debe conocer cómo es el orden de ejecución del algoritmo con que se va a trabajar, ya que la herramienta PlatEMO maneja carpetas y subcarpetas donde se almacena información de los distintos algoritmos que administra. Además, para el ingreso del problema de optimización en el programa PlatEMO se recomienda ingresar las funciones objetivo y restricciones en forma lineal, pues de esta manera el programa responde más rápido que si se trabaja con bucles. Especialmente si se tienen funciones muy extensas como es el presente caso de estudio.

7. Agradecimiento

Agradezco a Dios por ser mi guía, cuidarme y darme fuerza en momentos de dificultad y bendecirme en cada decisión tomada. A mis padres Jacinto y Dilma, por ser mi ejemplo a seguir, por confiar en mí, por sus consejos y enseñanzas y por su apoyo incondicional en cada momento de mi vida. A mis abuelitos José y Luz, gracias por rezar por mí cada noche, por su preocupación y cariño. A mis hermanos, tías, tíos y primos por tanto cariño y momentos compartidos, por cada palabra de ánimo y momentos en familia.

A los miembros del equipo de investigación IMAGINE de la Dirección de Investigación de la Universidad de Cuenca (DIUC), por la oportunidad de participar en el proyecto “Modelo de optimización de costos en la cadena de suministro en empresas de ensamblaje”; por su aporte invaluable en la elaboración de la tesis y por la gestión realizada para poder terminar este proyecto.

A la empresa caso de estudio, por su colaboración y aporte a la investigación.

A mi Director de tesis, Ing. Juan Llivisaca, por su ayuda, guía y compromiso en el desarrollo de este proyecto. Y al Ing. Mario Peña por sus opiniones y recomendaciones que permitieron resolver y direccionar los distintos aspectos presentados.

8. Referencias

Altiparmak, F., Gen, M., Lin, L., & Paksoy, T. (2006). A genetic algorithm approach for multi-objective optimization of supply chain networks. *Computers & industrial engineering*, *51*(1), 196-215. https://doi.org/10.1016/j.cie.2006.07.011

Amodeo, L., Chen, H., & El Hadji, A. (2008). Supply chain inventory optimisation with multiple objectives: An industrial case study. En A. Fink & F. Rothlauf (Eds.), *Advances in computational intelligence in transport, logistics, and supply chain management* (Vol. 144, pp. 211-230). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-69390-1_11

Babaveisi, V., Paydar, M. M., & Safaei, A. S. (2018). Optimizing a multi-product closed-loop supply chain using NSGA-II, MOSA, and MOPSO meta-heuristic algorithms. *Journal of Industrial Engineering International*, *14*(2), 305-326. https://doi.org/10.1007/s40092-017-0217-7

Coello Coello, C. (Ed.). (2005). Recent trends in evolutionary multiobjective optimization. En *Evolutionary multiobjective optimization. Theoretical advances and applications* (pp. 1-6). Springer-Verlag. https://doi.org/10.1007/1-84628-137-7

Coello Coello, C. (2006). Evolutionary multi-objective optimization: A historical view of the field. *IEEE Computational Intelligence Magazine*, *1*(1), 28-36. https://doi.org/10.1109/MCI.2006.1597059

Coello Coello, C., Lamont, G. B., & Veldhuizen, D. A. V. (Eds.). (2007a). MOEA local search and coevolution. En *Evolutionary algorithms for solving multi-objective problems* (Segunda, pp. 131-174). Springer US. https://doi.org/10.1007/978-0-387-36797-2_3

Coello Coello, C., Lamont, G. B., & Veldhuizen, D. A. V. (Eds.). (2007b). MOP evolutionary algorithm approaches. En *Evolutionary algorithms for solving multi-objective problems* (Segunda, pp. 61-130). Springer US. https://doi.org/10.1007/978-0-387-36797-2_2

Coello Coello, C., Pulido, G. T., & Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, *8*(3), 256-279. https://doi.org/10.1109/TEVC.2004.826067

Corne, D. W., Knowles, J. D., & Oates, M. J. (2000). The Pareto Envelope-Based Selection Algorithm for Multiobjective Optimization. En M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, & H.-P. Schwefel (Eds.), *Parallel problem solving from nature PPSN VI* (pp. 839-848). Springer. https://doi.org/10.1007/3-540-45356-3_82

Cuesta, S., Siguenza-Guzman, L., & Llivisaca, J. (2020). Optimization of assembly processes based on lean manufacturing tools. Case studies: Television and electronic card assemblers. *Communications in computer and information science (CCIS)*. 1st International Conference on Applied Technologies (ICAT), Quito, Ecuador.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182-197. https://doi.org/10.1109/4235.996017

Farahani, R. Z., & Elahipanah, M. (2008). A genetic algorithm to optimize the total cost and service level for just-in-time distribution in a supply chain. *International journal of production economics*, *111*(2), 229-243. https://doi.org/10.1016/j.ijpe.2006.11.028

Fonseca, C. M., & Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. *In proceedings of the fifth international conference on genetic algorithms*, *93*, 416-423.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley Publishing Company.

Gómez, H. B. D., Cáceres, R. G. G., & Mancilla, N. P. (2008). Las Pymes: Costos en la cadena de abastecimiento. *Revista Escuela de Administración de Negocios*, *63*, 5-22. https://doi.org/10.21158/01208160.n63.2008.438

Guamán Ochoa, M. M., Cárdenas Arias, B. E., Siguenza-Guzman, L., & Segarra, L. (2020). Integración de información de costos para la toma de decisiones en industrias de ensamblaje. *Revista Economía y Política*, 100-117.

Guerrero, M., Gómez, D., Zapata, D., & Cárdenas, M. V. (2016). Comparación de tres metaheurísticas para la optimización de inventarios con estimación de demanda. *Revista Ingeniería Industrial*, *15*(1), 51-68.

Guerrero, P. (2018). *Tiempos estándar y modelización de procesos de ensamblaje: Televisores y tarjetas electrónicas usando programación no lineal y BPMN* (Tesis de Pregrado) [Universidad de Cuenca]. http://dspace.ucuenca.edu.ec/handle/123456789/31279

Hernández Romero, N., Medina Marín, J., & Seck Tuoh Mora, J. (2012). *Introducción a matlab para resolver problemas de ingeniería. Aplicando algoritmos genéticos*.

Horn, J., Nafpliotis, N., & Goldberg, D. E. (1994). A niched Pareto genetic algorithm for multiobjective optimization. *Proceedings of the First IEEE Conference on Evolutionary Computation*, *1*, 82-87. https://doi.org/10.1109/ICEC.1994.350037

Jamshidi, R., Fatemi Ghomi, S. M. T., & Karimi, B. (2012). Multi-objective green supply chain optimization with a new hybrid memetic algorithm using the taguchi method. *Scientia Iranica*, *19*(6), 1876-1886. https://doi.org/10.1016/j.scient.2012.07.002

Javanshir, H., Ebrahimnejad, S., & Nouri, S. (2012). Bi-objective supply chain problem using MOPSO and NSGA-II. *International journal of industrial engineering computations*, *3*(4), 681-694. https://doi.org/10.5267/j.ijiec.2012.02.003

Kannan, G., Haq, A. N., & Devika, M. (2009). Analysis of closed loop supply chain using genetic algorithm and particle swarm optimisation. *International journal of production research*, *47*(5), 1175-1200. https://doi.org/10.1080/00207540701543585

Kao, Y.-T., & Zahara, E. (2008). A hybrid genetic algorithm and particle swarm optimization for multimodal functions. *Applied Soft Computing*, *8*(2), 849-857. https://doi.org/10.1016/j.asoc.2007.07.002

Kaplan, R. S., & Norton, D. P. (1992). The balanced scorecard: Measures that drive performance. *Harvard Business Review*, *January-February 1992*, 71-79.

Karaoglan, I., & Altiparmak, F. (2015). A memetic algorithm for the capacitated location-routing problem with mixed backhauls. *Computers & Operations Research*, *55*, 200-216. https://doi.org/10.1016/j.cor.2014.06.009

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *In proceedings of the 1995 IEEE international conference on neural networks*, *4*, 1942–1948.

Knowles, J. D., & Corne, D. W. (2000). M-PAES: A memetic algorithm for multiobjective optimization. *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No.00TH8512)*, *1*, 325-332 vol.1. https://doi.org/10.1109/CEC.2000.870313

Konak, A., Coit, D. W., & Smith, A. E. (2006). Multi-objective optimization using genetic algorithms: A tutorial. *Reliability engineering & system safety*, *91*(9), 992-1007. https://doi.org/10.1016/j.ress.2005.11.018

Latha Shankar, B., Basavarajappa, S., Kadadevaramath, R. S., & Chen, J. C. H. (2013). A bi-objective optimization of supply chain design and distribution operations using non-dominated sorting algorithm: A case study. *Expert systems with applications*, *40*(14), 5730-5739. https://doi.org/10.1016/j.eswa.2013.03.047

Mendoza, A., Fontalvo, T., & Visbal, D. (2015). Optimización multiobjetivo en una cadena de suministro. *Revista Ciencias Estratégicas*, *22*(32), 295-308.

Mesa Alvarez, J. J. (2017). *Algoritmo de optimización multiobjetivo basado en comportamiento emergentes de enjambres*. Universidad Distrital Francisco José de Caldas.

Moscato, P., & Cotta, C. (2003). Una introducción a los algoritmos memeticos. *Inteligencia artificial: Revista Iberoamericana de Inteligencia Artificial*, *19*, 131-148.

Moshref-Javadi, M., & Lee, S. (2016). The latency location-routing problem. *European Journal of Operational Research*, *255*(2), 604-619. https://doi.org/10.1016/j.ejor.2016.05.048

Movahedipour, M., Yang, M., Zeng, J., Wu, X., & Salam, S. (2016). Optimization in supply chain management, the current state and future directions: A systematic review and bibliometric analysis. *Journal of Industrial Engineering and Management*, *9*(4), 933-963. https://doi.org/10.3926/jiem.2035

Pinto, E. G. (2004). *Supply chain optimization using multi-objective evolutionary algorithms*. *15*.

Pishvaee, M. S., Farahani, R. Z., & Dullaert, W. (2010). A memetic algorithm for bi-objective integrated forward/reverse logistics network design. *Computers & Operations Research*, *37*(6), 1100-1112. https://doi.org/10.1016/j.cor.2009.09.018

Reed, P., Minsker, B., & Goldberg, D. E. (2000). Designing a competent simple genetic algorithm for search and optimization. *Water Resources Research*, *36*(12), 3757-3761. https://doi.org/10.1029/2000WR900231

Reyes-Sierra, M., & Coello Coello, C. (2006). Multi-objective particle swarm optimizers: A survey of the state of the art. *International Journal of Computational Intelligence Research*, *2*(3), 287-308.

Ruiz Lizama, E. (2014). Optimización multi-objetivo al problema de distribución de planta usando algoritmos genéticos: Cuestiones previas para una propuesta de solución. *Industrial Data*, *17*(2).

Sánchez Gómez, M. (2008). Generación de valor en la cadena de suministro. Best practices. En *Cuantificación y generación de valor en la cadena de suministro extendida* (pp. 91-120). Del Blanco.

Schaffer, J. D. (1984). *Some experiments in machine learning using vector evaluated genetic algorithms*. Vanderbilt University.

Srinivas, N., & Deb, K. (1994). Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, *2*(3), 221-248. https://doi.org/10.1162/evco.1994.2.3.221

Stephens, S. (2001). Supply chain operations reference model version 5.0: A new tool to improve supply chain efficiency and achieve best practice. *Information Systems Frontiers*, *3*(4), 471-476. https://doi.org/10.1023/A:1012881006783

Talavera, F., Prieto, J., Crichigno, J., & Barán, B. (2004). *Comparación de algoritmos evolutivos multi-objetivos en un ambiente multicast*. X Congreso Argentino de ciencias de la computación. http://hdl.handle.net/10915/22550

Tian, Y., Cheng, R., Zhang, X., & Jin, Y. (2017). PlatEMO: A MATLAB Platform for Evolutionary Multi-Objective Optimization [Educational Forum]. *IEEE Computational Intelligence Magazine*, *12*(4), 73-87. https://doi.org/10.1109/MCI.2017.2742868

Velasco Bermeo, N. (2009). *Algoritmo genético multiobjetivo autoevolutivo para resolver el problema de secuenciación de trabajos en una línea de ensamble con sistema de producción justo a tiempo*. Instituto Tecnológico y de Estudios Superiores de Monterrey.

Wu, T., & Blackhurst, J. (Eds.). (2009). *Managing supply chain risk and vulnerability: Tools and methods for supply chain decision makers*. Springer.

Xiong, F., Gong, P., Jin, P., & Fan, J. F. (2018). Supply chain scheduling optimization based on genetic particle swarm optimization algorithm. *Cluster Computing*. https://doi.org/10.1007/s10586-018-2400-z

Zapata Cortés, J. A. (2016). *Optimización de la distribución de mercancías utilizando un modelo genético multiobjetivo de inventario colaborativo de m proveedores con n clientes*. Universidad Nacional de Colombia - Sede Medellín.