Solución al problema de formación de celdas de manufactura dinámicas Virtuales (Dynamic Virtual Cell Formation Problem, DVCFP) a través de un algoritmo genético híbrido.

Laura Patricia Cortinez Parada
Claudia Alexandra Rodríguez García

Trabajo de grado para optar al título de Ingeniera Industrial

Director:
Edwin Alberto Garavito Hernández
Magíster en Ingeniería Industrial

Codirectora:
Laura Yeraldin Escobar Rodríguez
Ingeniera Industrial

Universidad Industrial de Santander
Escuela de Estudios Industriales y Empresariales
Bucaramanga
2019

Dedicatoria

A mis padres, Juan Carlos y Claudia Margarita
este y todos mis logros son por y para ustedes,
los amo mucho.

Claudia Alexandra Rodríguez García.

A mi familia: mamá, papá y Sandris
a quienes les debo lo que soy y me inspiran día a día.

Laura Patricia Cortinez Parada

Agradecimientos

Al profesor Edwin Garavito, el mejor profesor que tuvimos a lo largo de nuestro pregrado, gracias por ser nuestro guía, por su invaluable conocimiento ayudando a resolver los problemas que se presentaron de la manera más eficaz, a Laura Escobar, nuestra amiga y codirectora, gracias por la paciencia, por el tiempo, por siempre ser nuestro apoyo, y ayudarnos a crecer cada vez más como personas.

Al grupo Ópalo, gracias a todos por siempre estar dispuestos a aconsejar y escuchar, nos llevamos más que compañeros, amigos incondicionales.

A nuestra familia, por toda la paciencia, por ser nuestro soporte, gracias a ustedes somos las profesionales que hoy culminamos esta etapa.

.

Tabla de Contenido

Tabla de Figuras

Lista de Tablas

Lista de apéndices

Resumen

Título: Solución al problema de formación de celdas de manufactura dinámicas Virtuales (Dynamic Virtual Cell Formation Problem, DVCFP) a través de un Algoritmo Genético Híbrido*

Autores: Laura Patricia Cortinez Parada, Claudia Alexandra Rodríguez García**

Palabras clave: Celdas virtuales de manufactura, dinámicas, programación lineal entera mixta, algoritmo genético GA, Optimización por Enjambre de Partículas PSO, Algoritmo Híbrido.

Descripción:

En la presente investigación se aborda el problema de formación de celdas de manufactura virtuales en un ambiente de demanda dinámica (Dynamic Virtual Cell Formation Problem, DVCFP), entre diferentes periodos de tiempo, considerando varios productos con distintas operaciones y diferentes rutas de procesamiento, con el fin de acercarlo a la industria real minimizando los costos de producción. Gracias a la cantidad de variables el DVCFP es considerado de naturaleza NP hard, por lo tanto, para dar solución a este problema se utiliza un modelo matemático de naturaleza lineal entera mixta, y se desarrolla en el software algebraico GAMS/CPLX mediante el algoritmo Branch & Cut, utilizando 10 instancias clasificadas en 2 tamaños (grandes y pequeñas), generadas a partir de la adaptación de la literatura previa, y como alternativa se diseña un Algoritmo Genético Híbrido (GAPSO), combinando las ventajas de los algoritmos evolutivos y los algoritmos poblacionales como la Optimización por Enjambre de Partículas Discreta (DPSO). A partir de un diseño de experimentos se identifican los factores del algoritmo GAPSO que más repercusión presentan en la minimización de los costos asociados a la producción, dando como resultado que el tamaño de la población es el más importante en el desarrollo del algoritmo. Para finalizar se analizan los resultados obtenidos por cada método de solución, y se realiza su respectiva comparación entre el modelo exacto y la metaheurística diseñada, concluyendo que el Algoritmo GAPSO presenta mejores resultados en temas de costos y tiempo computacional que el modelo exacto para las instancias consideradas como pequeñas y grandes.

---

* *Trabajo de grado*

** *Facultad de Ingenierías Físico-Mecánicas. Escuela de Estudios Industriales y Empresariales. Director: Edwin Alberto Garavito Hernández, Magíster en Ingeniería Industrial. Codirector: Laura Yeraldin Escobar Rodríguez. Ingeniera industrial*

Abstract

Title: Solution of the Dynamic Virtual Cell Formation Problem, DVCFP using a hybrid genetic algorithm *

Authors: Laura Patricia Cortinez Parada, Claudia Alexandra Rodríguez García**

Keywords: virtual cell manufacturing, multiperiod, dynamic, GAMS, Mixed integer lineal programation, Genetic Algorithm GA, Particle Swarm Optimization, PSO, hybrid algorithm.

Description:

This research addresses the problem of virtual cell formation in an environment of dynamic demand (Dynamic Virtual Cell Formation problem, DVCFP), between different periods of time, considering several products with different operations and different processing routes, in order to bring it closer to the real industry minimizing production costs.Thanks to the amount of variables the DVCFP is considered to be of a NP hard nature, therefore, to solve this problem a mathematical model of a mixed linear program is used, and is developed in the algebraic software GAMS/CPLX, using 10 instances classified in 2 sizes (high and low), generated from the adaptation of the previous literature, and as an alternative a Hybrid Genetic Algorithm (GAPSO) is designed, combining the advantages of evolutionary algorithms and population algorithms such as Discrete Particle Swarm Optimization (DPSO). From a design of experiments are identified the factors of the algorithm GAPSO that present more impact in the minimization of the costs associated with the production, resulting that population size is the most important factor in algorithm development. To conclude, the results obtained by each solution method are analyzed, and their respective comparison between the exact model and the designed metaheuristic is performed, concluding that the GAPSO algorithm presents better results in cost and computational time issues than the exact model for instances considered as small and high.

---

* *Bachelor Thesis*

** *Facultad de Ingenierías Físico-Mecánicas. Escuela de Estudios Industriales y Empresariales. Director: Edwin Alberto Garavito Hernández, Magíster en Ingeniería Industrial. Codirector: Laura Yeraldin Escobar Rodríguez. Ingeniera Industrial.*

Introducción

Los sistemas de producción actuales se ven amenazados constantemente por las dinámicas de la globalización y las condiciones volátiles del mercado, por lo tanto, asegurar la satisfacción del cliente, ofrecer calidad funcional y atractiva tanto en los procesos como en los productos, entre otros, son factores que se traducen en una ventaja competitiva para la empresa. Esta situación obliga a las organizaciones a adoptar arquitecturas productivas más flexibles y eficientes, que puedan responder de una forma ágil a cambios en los productos y procesos; una de las estrategias utilizadas para optimizar el proceso de producción es el diseño de planta, el cual evalúa la distribución más conveniente de las operaciones de fabricación en cuanto a economía, minimización de tiempos y distancias, seguridad de los trabajadores, satisfacción del cliente, entre otros. Una de las estrategias de distribución de planta es la manufactura celular. Las celdas de manufactura *(Celullar Manufacturing, CM)* surgen como una alternativa de producción por lotes, en donde se busca la subdivisión de sistemas complejos de manufactura mediante la formación de grupos de productos con características en común, de tal manera que cada celda debe ser diseñada para procesar una o más grupos de partes, minimizando la distancia del recorrido de los productos entre máquinas y de esta manera, creando un flujo continuo (Nandurkar & Thomas, 2006).

Ahora bien, el mercado es un medio cambiante, con patrones de demanda impredecibles, y ciclos de vida cada vez más cortos, como consecuencia los sistemas de producción deben tener una respuesta inmediata ante los cambios externos al menor costo posible; por lo tanto, la misma configuración de los sistemas celulares no será óptima en diferentes periodos (Kheirkhah & Ghajari, 2018), surgiendo la necesidad de incorporar el horizonte de planeación en las formulaciones, pero, generando costos innecesarios de reconfiguración y subutilización de máquinas. Como respuesta a estas dificultades, surgen las Celdas de Manufactura Dinámicas

Virtuales (Dynamic Virtual Cells, DVC), las cuales consisten en grupos de máquinas que se encuentran dedicadas a un producto o familia de productos temporalmente al igual que las CM, solo que éstas pueden no estar físicamente agrupadas una con otra (McLean et al. 1982), solucionando los inconvenientes de las metodologías de producción anteriores y ampliando la flexibilidad del sistema, mostrando ventajas en cuanto a la utilización de máquinas, número de piezas producidas, promedio de fabricación, entre otros (Drolet et al. 1990).

Dado lo anterior, en la presente investigación se trabaja el Problema de Formación de Celdas de Manufactura Dinámicas Virtuales (Dynamic Virtual Cell Formation Problem, DVCFP) tomando como base la formulación presentada por Lv, Yuan, & Han, 2013, los cuales plantean el modelo a través de diferentes rutas y tiempos de procesamiento, tomando en cuenta restricciones de balance de carga para las celdas virtuales, bajo parámetros de demanda dinámica. Para solucionar este problema, se propone hacer uso de un algoritmo híbrido, que relaciona las metodologías del algoritmo genético (Genetic Algorithm, GA) y la Optimización por Enjambre de Partículas (Particle Swarm Optimization, PSO), de tal manera que se logre aprovechar las ventajas de cada uno en conjunto.

En este trabajo se comienza con una revisión teórica y de literatura, con el fin de realizar una adecuada contextualización del problema, de manera seguida se presenta la formulación matemática del DVCFP, junto con su correspondiente linealización a modelo MILP, de igual manera se explica el procedimiento realizado de manera exacta en el software GAMS y utilizando metaheurísticas a través de un Algoritmo Genético Híbrido (GAPSO) desarrollado en MATLAB, los resultados son documentados y analizados, presentando las respectivas conclusiones y recomendaciones para futuros investigadores. El cumplimiento de los objetivos planteados para la presente investigación se presenta en la Tabla 1.

Tabla 1. .
Cumplimiento de los objetivos del proyecto

Objetivo específico	***Capítulo***
Identificar la estructura del DVCFP a partir de una revisión de literatura con el propósito de caracterizar las variantes realizadas y los métodos de solución implementados.	Capítulo 3
Diseñar un modelo de programación matemática para el DVCFP, solucionarlo haciendo uso de técnicas exactas y validarlo utilizando instancias encontradas en la literatura.	Capítulo 4 Capítulo 5
Solucionar el modelo matemático propuesto para el DVCFP a través del algoritmo Genético Híbrido a mediante un software de programación (MATLAB).	Capítulo 5
Evaluar el desempeño del algoritmo mediante la comparación con instancias de la literatura o a través de un análisis estadístico de los resultados.	Capítulo 6 Capítulo 7
Realizar un artículo académico de carácter publicable basado en el trabajo de investigación realizado.	Apéndice E

1. Especificaciones del proyecto

1.1 Planteamiento del problema

La competencia internacional y el mercado volátil de la actualidad han impuesto una gran presión sobre los sistemas de fabricación para aumentar su efectividad. En los últimos años, los mercados han experimentado un rápido aumento en la variedad y disminución del ciclo de vida de los productos, reduciendo la habilidad de los sistemas de fabricación clásicos como Flow shop y los sistemas de taller de trabajo para responder a estos "mercados mutantes". Estos cambios rápidos fuerzan a las compañías a aumentar su capacidad de respuesta, flexibilidad, acortar los tiempos de configuración y reducir el inventario de trabajo en proceso, mientras se mantiene una eficiencia aceptable al mínimo costo posible (Drolet, Abdulnour & Rheault, 1996). Como respuesta a estos

requerimientos surge la metodología denominada Celdas de Manufactura Virtuales, en las cuales un grupo de recursos (máquinas, personas, materiales), se asignan a una parte o una parte de la familia de productos como en una celda de manufactura convencional, pero esta agrupación puede no reflejarse en la estructura física del sistema (Kesen, Das, & Gungor, 2010), ahora bien, si a este sistema se le agrega consideraciones en cuanto al horizonte de planeación, resulta en una estructura lógica que evita costos y tiempo perdido de reconfiguración de la planta de producción, permitiendo además, una respuesta rápida a las fluctuaciones del mercado entre periodos (Taylor, Arkat, & Ghahve, 2014).

Al dar solución al Problema de Celdas de Manufactura Dinámicas Virtuales (DVCFP), se busca minimizar los costos asociados al proceso de manufactura, tales como tardanza de los productos, manejo de materiales, costos internos de producción entre otros, tratando de integrar la mayor cantidad de parámetros y atributos de las industrias con el fin de poder implementarlos en sistemas de producción reales. En el presente trabajo de investigación se plantea la consideración de diferentes rutas de procesamiento, capacidad limitada de las máquinas, secuencia de operaciones y configuración de lotes, bajo consideraciones de demanda dinámica, con objeto de minimizar los costos totales de operación, al cual, aprovechándose de la naturaleza NP- hard del modelo matemático, se busca la solución de éste mediante la interrelación de dos de los algoritmos más usados en la resolución de documentos anteriores acerca del DVCFP. El algoritmo genético (AG) y la Optimización por enjambre de partículas (PSO), han ayudado a diferentes autores en la búsqueda de valores ideales, generando resultados válidos en tiempos computacionales aceptables de manera separada, aun así, en la consulta bibliográfica previamente realizada, no se han encontrado autores que utilicen este híbrido en sus formulaciones acerca del problema en cuestión.

Haciendo referencia a la pertinencia teórica y práctica de la investigación, se considera que, como ingenieros industriales o comúnmente llamados ingenieros de producción, uno de los mayores propósitos del ejercicio profesional es la búsqueda de métodos eficaces para los sistemas de manufactura; además, tomando en cuenta investigaciones anteriores enfocadas al Diseño de Sistemas Productivos por parte de la Escuela de Estudios Industriales y Empresariales, el presente trabajo tiene como fin aportar a la literatura de la temática abordada, proporcionando la solución mediante un algoritmo híbrido, queriendo ser punto de referencia para la búsqueda de métodos de solución cada vez más efectivos, junto a la incorporación de diferentes parámetros y/o atributos que acerquen cada vez más el DVCFP a la industria real.

1.2 Justificación del proyecto

El mundo globalizado y cada vez más competitivo al que los ingenieros industriales se enfrentan, exige la adaptación al cambio, agilidad para responder a los clientes en el menor tiempo posible y eficiencia en el uso de los recursos disponibles, por lo tanto, los sistemas de producción deben comportarse de la misma manera.

A lo largo de la historia, se han realizado múltiples estudios e investigaciones acerca del diseño y distribución de planta adecuado para cada tipo de empresa, donde la búsqueda de métodos de producción que disminuyan el tiempo de procesamiento, costos, entre otros ha si un punto de enfoque importante para la industria debido a los beneficios y mejoramientos que se han presentado.

Las celdas de manufactura surgen como una alterativa de producción por lotes aplicando los principios de la tecnología de grupos en donde se hace una agrupación de recursos y maquinas disponibles a partir de familias de productos o partes similares, optimizando el proceso de

manufactura, aun así, el mercado es un medio cambiante, con patrones de demanda impredecibles, y ciclos de vida cada vez más cortos. Como consecuencia los sistemas de producción tienen que ser flexibles y tener una respuesta inmediata ante los cambios externos, es por esto que muchas veces los horizontes de planeación y la programación de celdas para un determinado periodo de tiempo resulta ser la menos adecuada debido a la agrupación física de las maquinas por lo que se genera costos innecesarios, costos de reconfiguración de celdas y poca utilización de recursos en diferentes ventanas de tiempo.

Como una solución a este problema, surgen las Celdas de Manufactura Dinámicas Virtuales *(DVC),* en la cual, las máquinas se encuentran dedicadas a un producto o familia de productos temporalmente al igual que las CM, solo que éstas pueden no estar físicamente agrupadas una con otra *(McLean et al. 1982),* solucionando los inconvenientes de las metodologías anteriores y ampliando la flexibilidad del sistema.

El Problema de Formación de Celdas de Manufactura Dinámicas Virtuales *(Dynamic Virtual Cell Formation Problem, DVCFP),* es considerado una metodología emergente y prometedora (Rabbani, Farrokhi-Asl, Rafiei, & Khaleghi, 2017), aplicable en empresas de mediano y pequeño tamaño, ya que son las impulsadoras de la economía de la ciudad y de Colombia.

Debido a que este problema es de naturaleza NP-Hard, se hace necesario en orden de reducir el consumo de recursos informáticos y de tiempo, utilizar metaheurísticas que desarrollen algoritmos de naturaleza híbrida que conlleven a una respuesta óptima y rápida.

Finalmente se busca ampliar y adaptar a la realidad de la industria moderna una naciente línea de investigación dentro de la Escuela de Estudios Industriales y Empresariales, y como forma de aplicación del Diseño de Sistemas Productivos, en el presente trabajo se busca estudiar el DVCFP

siguiendo las recomendaciones de Lv et al., 2013, quienes formulan el problema considerando parámetros o atributos de los sistemas de manufactura tales como capacidad de los recursos, niveles de inventario, requerimientos de productos con base a una demanda dinámica y secuencias de operaciones junto con rutas diferentes, resolviéndolo mediante un Algoritmo genético híbrido en la búsqueda de obtener valores ideales.

2. Objetivos

2.1 Objetivo General

Desarrollar un modelo matemático con el fin de resolver el problema de Formación de Celdas de Manufactura Dinámicas Virtuales (DVCFP) basado en una formulación de diferentes rutas y tiempos de procesamiento, a través de un algoritmo genético híbrido.

2.2 Objetivos Específicos

- Identificar la estructura del DVCFP a partir de una revisión de literatura con el propósito de caracterizar las variantes realizadas y los métodos de solución implementados.
- Diseñar un modelo de programación matemática para el DVCFP, solucionarlo haciendo uso de técnicas exactas y validarlo utilizando instancias encontradas en la literatura.
- Solucionar el modelo matemático propuesto para el DVCFP a través del algoritmo Genético Híbrido a mediante un software de programación (MATLAB).
- Evaluar el desempeño del algoritmo mediante la comparación con instancias de la literatura o a través de un análisis estadístico de los resultados.
- Realizar un artículo académico de carácter publicable basado en el trabajo de investigación realizado.

3. Marco de referencia

3.1 Revisión de la literatura

El primer autor en referirse al concepto de Celdas Virtuales de Manufactura (*Virtual Cell Manufacturing, VCMs*) fue McLean et al. (1982), definiendo a la celda virtual como un proceso de computador, datos asociados y un conjunto de estaciones de trabajo dinámicas en el taller, que no es identificable como un grupo físico fijo de máquinas. Lo anterior, surge como una respuesta para abordar los problemas de sobrecostos y de control específicos que se evidenciaban en la fase de diseño y puesta en marcha de las CM convencionales; de otra parte, Nomden, Slomp, & Suresh (2006) definen a la VCM como un grupo de recursos los cuales son dedicados a la fabricación de partes de familias de productos, aunque esta agrupación puede no estar reflejada en la estructura física del sistema de producción.

Tomando en cuenta lo anterior, Rheault et al., (1995; 1996) proponen una extensión del concepto de VCMs, en el cual, aprovechando los principios de la tecnología de grupos y las facilidades de reconfiguración de las VCMs, incorporan el horizonte de planeación en sus formulaciones con el fin de responder de manera ágil a los cambios en la mezcla y demanda de los productos. El diseño de las VCMs se conoce como Problema de Formación de Celdas Virtuales de Manufactura Dinámicas, (*Dynamic Virtual Cell Formation Problem, DVCFP*). En otras palabras, cuando llega un nuevo periodo, el sistema evalúa el estado de las máquinas y realiza la respectiva asignación de recursos sin detener la producción.

De esta manera, McLean CR, Bloom HM, Hopp TH, Rheault et al., dan paso a una "estrategia de alto nivel", siendo una decisión de planeación y control de la producción en un sistema de manufactura ya distribuido. Sin embargo, esta organización, puede guiar el diseño de la celda

virtual hacia una mayor eficacia, generando así a una serie de estudios y variantes del DVCFP en el cual, sus investigadores no se han centrado únicamente en la disminución de los diferentes costos en su función objetivo (formulación mono objetivo), sino que también centran su interés, en el análisis de capacidad de recursos disponibles, asignación de trabajadores, tamaño de lotes, programación de tareas y trabajos e integración a la cadena de suministro para un horizonte de planeación establecido, tomando en cuenta diferentes parámetros (tales como, parámetros determinísticos y difusos en cuanto a la demanda y la mezcla de productos, duplicación de máquinas, tiempos de procesamiento, fechas de entrega, limitaciones de herramientas de trabajo críticas, división por lotes, tiempo de comienzo de cada trabajo, rutas alternativas de procesamiento, entre otros), que influyen en la implementación de este sistema en la industria (formulación multi objetivo), finalmente, son identificados los principales métodos de solución utilizados para resolver el DVCFP mediante la revisión de la literatura previamente realizada.

3.1.1 Formulación Mono objetivo. La minimización de costos ha sido el criterio planteado con mayor frecuencia en la literatura del problema en cuestión. Mertins, Friedland, & Rabe (2000) realizan la asignación de capacidades en cada una de las VCMs, a través de la "armonización" del tamaño del lote, con el fin de minimizar los costos de tiempos de preparación de las máquinas.

Asimismo, Jayachitra, Revathy, & Prasad (2010) tienen en cuenta solo la formación de DVCs, a través de parámetros difusos en razón a la demanda y la capacidad de los recursos en los diferentes periodos, junto con la minimización del costo total relacionado con las máquinas y el movimiento secuencial de materiales, Rezazadeh, Mahini, & Zarei (2011), crean un modelo que busca determinar el número óptimo de DVCs para cada periodo, minimizando el costo de manejo de materiales e inventario, subcontratación y otros costos internos de producción, incorporando parámetros determinísticos variantes en cuanto a la demanda y la mezcla de productos. Lv et al.

(2013), realizan una metodología detallada a la realidad de las industria relacionando el tamaño de la celda con los costos internos de producción (costo de alistamiento, trabajo en proceso, inventario, coordinación, manejo de inventario), denotándolos como una función creciente del número de máquinas por celda, incorporando parámetros de la vida real como duplicación de máquinas y tiempos de procesamiento y así, demostrando a través de su modelo que el tamaño de la DVC cambia al igual que el periodo en condiciones dinámicas y diferentes escenarios.

Más adelante, Han, Wang, & Lv (2014) incorporan el costo de configuración de rutas en su función objetivo, integrando el Programa Maestro de Producción (MPS) en su metodología; por otro lado, Wenming Han & Tong (2015), estudian el impacto de la inserción de nuevas tareas en un periodo de producción previamente definido, con el propósito de disminuir o eliminar el conflicto de recursos restrictivos en algunos periodos, y solucionándolo a través de una metodología de red bayesiana; un año más tarde, Tesic, Stevanov, Jovanovic, Tomic, & Kafol (2016), enfocan su propósito hacia el control del tamaño de lote por periodos, incorporando en su función objetivo el aumento del coeficiente de similitud de las piezas, reducción del número de celdas virtuales creadas, disminución de la distancia entre máquinas, aminorando la superposición de celdas, reducción de la duración del periodo y del número de etapas de producción con el fin de acortar el tiempo de entrega de pedido al cliente realizando una serie de pruebas en las cuales simulan diferentes escenarios, para comprobar como la definición previa de un tamaño de lote adecuado tiene una gran influencia en la eficacia de la DVC; en el mismo año, Moradgholi, Paydar, Mahdavi, & Jouzdani (2016), incorporan el costo de los trabajadores bajo condiciones de diferentes habilidades.

3.1.2. Formulación multiobjetivo. Diversos autores han abarcado el problema como un modelo multi objetivo, bajo la condición de formar DVCs óptimas junto con diferentes objetivos adicionales a la función de costos, que se enumerarán a continuación:

3.1.2.1. Programación de las operaciones (schedule problem). Consiste en elaborar un programa que indique las fechas en que debe realizarse cada operación correspondiente a cada pedido, de tal manera que se organice y elija el uso de los recursos que participan en el proceso de producción. Mak & Wang (2002) desarrollaron un algoritmo de programación entera mixta (*MIP*) que minimiza la distancia total en el transporte de materiales y componentes a la que se incurre al fabricar el producto, junto con una metodología de configuración de las DVCs y un método de programación de la producción factible para todas las operaciones de fabricación, generando un uso adecuado de maquinaria, flexibilidad en las rutas de producción y eficiencia en los tiempos de preparación. Más adelante, Mak, Peng, Wang, & Lau (2007) incorporan restricciones a su modelo de programación no lineal entera mixta (*MINLP*) de la producción en la búsqueda de un programa definido y construido, tales como, fechas de entrega, capacidad máxima de los recursos y limitaciones de herramientas de trabajo críticas. Posteriormente, Kai Ling Mak, Ma, & Su (2010), integran al problema de programación de la producción una función objetivo que busca la minimización del costo total de manufactura (transporte de materiales, operación de máquinas, salario de trabajadores y penalizaciones por tardanzas), junto con restricciones en la fuerza de trabajo; Baykasoglu & Gorkemli (2017) proponen un modelo para desarrollar un diseño de distribución de planta para una DVC a través de un modelo basado en edades, tomando en cuenta la formación de cada parte de la familia de productos, dividiendo en fases la programación a través de parámetros de demanda dinámicos.

3.1.2.2. Makespan. Con el fin de determinar el plan adecuado de producción, K. L. Mak, Lau, & Wang (2005) buscan mejorar la metodología previamente establecida de su modelo, adicionando la minimización del makespan (días en total que un pedido demora en producirse), mostrando que las DVCs dan mejores resultados en términos de utilización de la estación de trabajo, tiempos de finalización, throughput y rendimiento, a comparación con los sistemas de CM convencionales al utilizar un caso de la industria real. Kesen, Das, & Güngör (2010) incorpora la reducción del makespan en su función objetivo junto con la distancia total de transporte de materiales y productos en el DVCFP, generando un programa de producción de múltiples trabajos con diferentes rutas de procesamiento, tomando en cuenta la división de lotes y el tiempo de comienzo de cada trabajo en la respectiva máquina asignada.

3.1.2.3. Planeación de la producción (Production Planning). Las decisiones de planeación anticipada en cuanto a recursos limitados, tales como de mano de obra, materia prima, maquinarias, equipo, entre otros, tienen un rol importante en las previas publicaciones acerca del DVCFP, además, algunos autores han propuesto que los requerimientos de fuerza de trabajo deben ser tomados en cuenta en la etapa de diseño de las DVCs un ejemplo de esto se puede encontrar en Mahdavi, Aalaei, Paydar, & Solimanpur (2009) en el cual, basados en decisiones de planeación generan una función que se asegura no se sobrepase la capacidad disponible, minimizando costos de manejo y calculando el costo de tardanza, a través de un modelo que considera flexibilidad en las habilidades de los trabajadores; luego, Iraj Mahdavi, Aalaei, Paydar, & Solimanpur (2011) incorporan a su modelo parámetros difusos y la minimización de costos de pedidos pendientes y la disponibilidad de recursos (maquinas-partes-trabajadores); un año después, Nikoofarid & Aalaei (2012), se basan en el mismo modelo, tomando como punto de inicio la satisfacción total de la demanda mediante parámetros determinísticos.

3.1.2.4. Resource Elements (REs). A pesar de que es una de las metodologías más utilizadas en los estudios de DVCs considerando un solo periodo, también existen trabajos de autores que incorporan la metodología de REs en los estudios dinámicos. Wenmin Han, Chen, & Zhao (2013) definen este como el método para especificar la capacidad única y compartida que presenta cada una de las máquinas a través de vectores (teniendo en cuenta los movimientos entre la pieza y la máquina, niveles tecnológicos, entre otros). Asimismo, Ratchev (2001)describe un método iterativo para diseñar una metodología de diseño de las DVCs a través de cuatro pasos, el primer paso consiste en identificar los requerimientos de componentes mientras se generan las alternativas de procesamiento; entonces, los límites de las capacidades de las DVCs se definen para después seleccionar las herramientas de cada de una de las máquinas elegidas; el paso final es la evaluación del sistema, todo esto utilizando un modelo con parámetros difusos y rutas alternativas, mediante el cual se obtienen resultados acertados para la industrial real, más tarde Liang & Fung (2016) retoman la metodología incorporando la programación de trabajos con el objetivo de conformar las DVCs óptimas junto con la reducción de los costos de procesamiento y manejo de materiales.

3.1.2.5. Cadena de Suministro. El proceso de producción juega un papel crucial en la eficacia de la cadena de suministro, por eso, diferentes autores han mostrado interés en integrar el diseño de ésta junto con la configuración de la DVCs, tal es el caso de Paydar & Saidi-Mehrabad (2015), que dan el primer paso hacia esta integración; a través de la minimización de los costos de compras, producción y distribución de diferentes tipos de productos, junto con la maximización de la eficacia de las partes producidas mediante la agrupación de DVCs. Aalaei & Davoudpour (2016) más adelante, incorporan al diseño de la cadena de suministro y la metodología de formación de DVCs diferentes consideraciones como ubicaciones de múltiples plantas, puntos de venta, sobre demandas de productos y capacidades de recursos inciertas, cabe resaltar que en su función

objetivo, incorporan el costo total de ésta (costos de manejo de materiales, partes y productos, outsourcing, mantenimiento, transporte de materiales, partes y productos dentro y fuera de la planta).

3.1.3. Métodos de solución. Diferentes métodos han sido utilizados con el fin de resolver los modelos matemáticos obtenidos para el DVCFP, algunos autores como Lv et al. (2013) e I. Mahdavi et al. (2009) utilizan métodos exactos como ramificación y acotación (Branch & Bound, B&B), a través de software matemáticos, tales como LINGO 11.0 y 8.0, aplicando diferentes casos de estudio. Debido a que él DVCFP es un problema clasificado como NP-hard, de acuerdo con su complejidad computacional, gran parte de los autores incorporan heurísticas y metaheurísticas a sus documentos.

Uno de los métodos utilizados con mayor frecuencia para abordar el DVCs es el Algoritmo Genético (*Genetic Algorithm, GA*); Moradgholi et al. (2016) concluyen que este algoritmo es capaz de obtener buenas soluciones factibles, en diversos tamaños de instancias y en tiempos computacionales razonables. Otros autores, plantean variaciones al GA, por su parte, K L Mak & Wang (2002), utilizan un Algoritmo Genético de Búsqueda (*Genetic Search Algorithm, GSA*), el cual emplea esquemas de representación para codificar cada solución candidato en forma de cadenas, y un esquema de representación binaria para la programación de trabajos, haciendo que se logre responder rápidamente a cualquier cambio en el horizonte de planeación. Más adelante, K. L. Mak et al. (2005) prueban un Algoritmo Genético Basado en la Edad (*Age Based Genetic Algorithm, ABGA*), donde las poblaciones de soluciones candidatas consiste en individuos de diferentes grupos de edad, proporcionando resultados satisfactorios en términos de velocidad computacional y calidad de la solución final, demostrando que el rendimiento del algoritmo en cuestión supera al AG tradicional. Por otra parte, K. L. Mak et al. (2007), prueba un híbrido entre

el GA y Algoritmo de la Colonia de Hormigas (*Ant Colony Optimization, ACO*), debido a que el resultado de ACO no logra mostrar el programa de producción de manera explícita, es necesario utilizar otro método que asigne operaciones a cada estación de trabajo y así generar el programa de producción final; demostrando que la combinación de estos dos algoritmos junto con los conceptos de VCM, tienen mejor desempeño que las prácticas de manufactura existentes. Luego, Paydar & Saidi-Mehrabad (2017), proponen otro algoritmo híbrido, la Búsqueda de Entorno Variable (*Variable neighborhood search, VNS*), es conocida como una metaheurística de trayectoria efectiva; sin embargo, cuenta con una exploración limitada de la capacidad, por lo tanto, se combina con el GA, obteniéndose buenas soluciones en tiempos computacionales 70 veces menores (aproximadamente) que software como LINGO, demostrando la efectividad y el alcance del GA.

Por otro lado, diversos autores han utilizado las variantes de la Optimización por enjambre de partículas *(Particle Swarm Optimization, PSO*), Kai Ling Mak et al. (2010), utilizan un híbrido entre la Programación con Restricciones (*Constraint Programming, CP*) y PSO, donde CP proporciona las soluciones iniciales, y PSO ayuda a mejorarlas, dando como resultado un algoritmo que entrega resultados mejores en un tiempo computacional considerable, mientras se muestra un mejor rendimiento en la localización del óptimo cuando el tamaño del trabajo es grande. Más adelante, K.L. Mak, Ma, & Cui (2011) muestran la efectividad del algoritmo híbrido propuesto previamente, resolviendo conjuntos de prueba generados al azar, utilizando restricciones complejas. De otra parte, Rezazadeh et al. (2011) proponen un Algoritmo híbrido de Programación Lineal de Optimización por Enjambre de Partículas Incrustadas (*Linear Programming Embedded Particle Swarm Optimization Algorithm, LPEPSO*) y el Algoritmo de Recocido Simulado (*Simulated Annealing, SA*), el cual, genera soluciones de alta calidad en un tiempo insignificante,

aunque, a medida que el tamaño del problema crece, el uso de métodos exactos como B&B, muestra superioridad en cuanto a las soluciones generadas en comparación al algoritmo híbrido.

De otra parte, Ratchev (2001), utiliza el Análisis de Grupos (*Clustering Algorithm, CA*), generando una combinación apropiada de DVCs, para suplir los requerimientos de cada producto en específico. Jayachitra et al. (2010) utiliza tres métodos diferentes para la solución de su modelo, comparándolos entre sí: SA, Lógica Difusa (*Fuzzy Logic, FL*) y el Algoritmo de Agrupamiento por Orden de Rango (*Rank Order clustering Algorithm, ROC*). Asimismo, Iraj Mahdavi et al. (2011) emplean un Enfoque Difuso basado en la Programación por Objetivos *(Fuzzy goal programming-based approach, FGP*), el cual de acuerdo a los resultados obtenidos por los autores, es útil para la toma de decisiones prácticas; sin embargo no es muy conveniente para abordar problemas de la vida real. Murali (2012) centra su objetivo en la asignación de la fuerza de trabajo a cada DVC mediante las Redes Neuronales Artificales (*Artificial neural network, ANN*), utilizando el método de Aprendizaje de Cuantificación Vectorial (*Learning Vector Quantization, LVQ*), los resultados obtenidos en su estudio permiten concluir que el enfoque basado en LVQ, es útil y efectivo bajo diferentes configuraciones de celdas en diferentes periodos de tiempo. Recientemente, Aalaei & Davoudpour (2016) emplean un enfoque de Programación de Metas Múltiples (*Multi- Choice Goal Programming Approach, MGPA*), mostrando soluciones prometedoras en su rendimiento, al generar un modelo que tiene la capacidad de gestionar demandas dinámicas de partes eficientemente.

Teniendo en cuenta la investigación anterior junto con las recomendaciones realizadas por diversos autores, se plantea el balance de carga de trabajo para cada DVC, y la posibilidad de rutas alternativas de procesamiento para cada producto como atributos a tener en cuenta en la investigación. Su importancia se evidencia en la búsqueda de la minimización de los costos de

transferencia de materiales (los cuales se encuentran relacionados con la ruta de cada producto), la distribución equitativa del trabajo entre celdas y, además, el objetivo de acercar los modelos matemáticos a la industria real. Por otra parte, tomando en cuenta los resultados obtenidos a partir de la revisión realizada, se plantea para el presente proyecto el uso del Algoritmo Genético y la Optimización por Enjambre de Partículas como un algoritmo híbrido, considerando su compatibilidad en el manejo de un grupo de individuos, así como la búsqueda de buenas soluciones en tiempos computacionales reducidos.

La información consolidada de los atributos revisados en el presente estudio se presenta en la Tabla **2**, consolidados en las Figura 1 y Figura 2.

Tabla 2. .
Atributos de los sistemas de manufactura estudiados

Atributos de los sistemas de manufactura estudiados	
1. Objetivo de diseño	
1a	*Capacidad*
1b	*Manipulaciòn*
1c	*Tardanza*
1d	*Distancia*
1e	*Makespan*
1f	*Beneficios económicos*
1g	*Elementos excepcionales*
1h	*Diseño de la cadena de suministro*
1i	*Programación de la producción*
1j	*Formación de partes de familias*
1k	*Inserción de nuevas tareas*
1l	*Control de lote*
2. Recursos utilizados	
2a	*Maquinas*
2b	*Inventario*
2c	*Personas*
3. Método de diseño	
3a	*Análisis de grupos*
3b	*Elementos recurso (Res)*
3c	*Programación no lineal entera mixta (MINLP)*
3d	*Programación entera mixta (MIP)*

Continuación Tabla 2. *Atributos de los sistemas de manufactura estudiados*

3e	***Programación lineal (LP)***
3f	*Modelo basado en edades*
3g	*Sistema multiagente*
3h	*Programación difusa*
3i	*Coeficientes de similitud*
3j	*Armonización de tamaño de lote*
3k	*Red bayesiana*
4. Método de solución	
5. Parte de la familia	
Trabajos	
Múltiples partes	
6. Nivel de implementación	
MPS	
MRP	
SFC	
7. Multiobjetivo	

AUTOR	1												2		
	a	b	c	d	e	f	g	h	i	j	k	l	a	b	c
Mertins, Friedland, & Rabe (2000)	x													x	
Ratchev (2001)	x												x	x	
Mak & Wang (2002)		x							x				x	x	
Mak, Lau, & Wang (2005)		x	x	x	x				x				x	x	
Mak, Peng, Wang, & Lau (2007)				x					x				x	x	
Mahdavi, Aalaei, Paydar, & Solimanpur (2009)						x							x		x
Jayachitra et al. (2010)						x							x	x	
(Kesen, Das, & Gungor, 2010)				x	x				x				x	x	
K. Mak, Ma, & Su (2010)						x			x				x		x
Iraj Mahdavi, Aalaei, Paydar, & Solimanpur (2011)						x	x		x				x	x	x
K. L. Mak, Ma, & Cui (2011)						x			x				x	x	
Rezazadeh et al. (2011)						x							x	x	
Murali RV (2012)												x			x
Nikoofarid & Aalaei (2012)						x							x	x	x
Lv et al., (2013)						x							x		
Wenmin Han, Wang, & Lv (2014)						x							x	x	
Han & Tong (2015)											x		x		
Paydar & Saidi-Mehrabad (2015)						x		x					x		
Aalaei & Davoudpour (2016)						x		x					x		
Liang & Fung (2016)									x				x		
Moradgholi et al. (2016)						x							x	x	x
Tesic, Stevanov, Jovanovic, Tomic, & Kafol (2016)												x	x	x	
,Baykasoglu & Gorkemli (2017)									x	x			x		
Paydar & Saidi-Mehrabad (2017)						x		x					x	x	

Figura 1. Atributos de los sistemas estudiados – parte 1.

AUTOR	3											4	5		6			7
	a	b	c	d	e	f	g	h	i	j	k		a	b	a	b	c	
Mertins, Friedland, & Rabe (2000)										x		SIMULACIÓN		x		x		
Ratchev (2001)	x	x										CA		x			x	
Mak & Wang (2002)				x								GSA	x			x	x	x
Mak, Lau, & Wang (2005)			x									ABGA	x			x	x	x
Mak, Peng, Wang, & Lau (2007)			x									ACO / AG		x		x	x	x
Mahdavi, Aalaei, Paydar, & Solimanpur (2009)			x									LINGO 8.0		x	x		x	
Jayachitra et al. (2010)			x					x				SA / FG / ROC		x	x		x	
Kesen, Das, & Gungor (2010)				x								GAMS	x			x	x	x
K. Mak, Ma, & Su (2010)			x									CP / PSO	x			x	x	x
Iraj Mahdavi, Aalaei, Paydar, & Solimanpur (2011)			x					x				FGPBA		x	x		x	x
K. L. Mak, Ma, & Cui (2011)				x								DPSO / CP	x			x	x	x
Rezazadeh et al. (2011)				x								LPEPSO / SA		x		x	x	
Murali RV (2012)					x							LVQ	x		x			
Nikoofarid & Aalaei (2012)				x								LINGO 9.0		x	x		x	x
Lv et al., (2013)			x									LINGO 11.0		x	x		x	
Wenmin Han, Wang, & Lv (2014)				x								CASO ESTUDIO		x		x	x	
Han & Tong (2015)											x	CASO ESTUDIO		x	x		x	
Paydar & Saidi-Mehrabad (2015)			x					x				MCGP		x	x		x	x
Aalaei & Davoudpour (2016)			x									RMCGP		x	x		x	x
Liang & Fung (2016)		x					x					CASO ESTUDIO		x		x	x	
Moradgholi et al. (2016)			x									GA		x		x	x	
Tesic, Stevanov, Jovanovic, Tomic, & Kafol (2016)									x			CASO ESTUDIO		x	x		x	
,Baykasoglu & Gorkemli (2017)		x				x						ANYLOGIC		x				x
Paydar & Saidi-Mehrabad (2017)			x									GA/VNS		x		x	x	x

*Figura 2***.** Atributos de los sistemas estudiados – parte 2.

3.2. Marco Teórico

3.2.1. Tecnología de grupos (Group technology). Es una filosofía que tiene como idea principal agrupar a Partes o piezas de productos Con el fin de formar familias, que tengan en común operaciones similares y máquinas de procesamientos relacionadas a esas operaciones (Elanchezhian, Shanmuga Sundar, & Sundar Selwyn, 2007). Debido a la estructura que plantea esta filosofía se forman las Celdas de Manufactura, que aplica los principios que propone GT (YIN & YASUDA, 2006). La celda de manufactura se plantea como una alternativa de la fabricación convencional por lotes, en donde los productos o partes de productos que no alcanzan un volumen suficiente como para tener su propia línea de fabricación se juntan con partes y productos similares que requieran el mismo proceso y/o recursos (Miltenburg and Zhang, 1991), con el objetivo de

reducir costos y optimizar los recursos ; ya que resulta una manera más económica de producción, las ventajas asociadas a este tipo de sistemas de manufactura son: la simplificación del flujo de piezas y herramientas, la reducción en el tiempo de configuración, el tiempo de rendimiento e inventario en proceso (Elanchezhian et al., 2007).

3.2.2. Celdas de manufactura virtuales. Las celdas de manufactura virtuales son aquellas que tienen como objeto los mismos principios de la celda de manufactura tradicional, con la variante de que el diseño y configuración de la celda no se materializa físicamente, es decir, se asocia a un proceso de computadora, datos virtuales y estaciones de trabajo dinámicas cambiantes en el piso de la tienda (McLean et al. 1982).

A medida de los años el concepto de VCM fue evolucionando, pero la producción por lotes aún sigue vigente, una de las características de VCM es que responde a los cambios que surgen de la demanda debido a que la configuración de celdas no están comandadas por agrupación física de las máquinas, por lo que permite que las partes tengan rutas alternativas de procesamiento como se observa en la *Figura 3* (Kesen, Das, & Gungor, 2010).

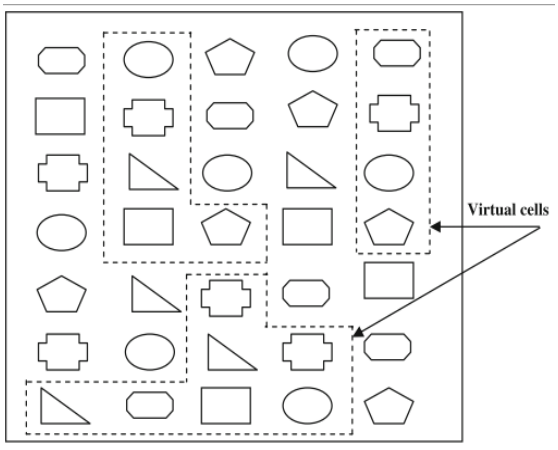


Figura 3. Representación de una celda de manufactura virtual. Adaptado de A mixed integer programming formulation for scheduling of virtual manufacturing cells (VMCs) (Kesen, Das, & Gungor, 2010).

Para un mejor entendimiento del problema, se llevó este estudio a un horizonte de planeación por periodos, en donde surge el DVCFP, en donde la primera operación debe realizarse en el primer periodo y la segunda en el siguiente, aún si las operaciones del primer periodo hayan terminado antes (Kesen, Das, & Gungor, 2010).

3.2.3. Complejidad Computacional. La complejidad computacional hace parte de las 3 áreas de estudio de la Teoría de la Computación. Esta lo que busca es clasificar los problemas de acuerdo con su dificultad y la relación que existe entre esas dificultades, es decir, darle explicación a porque algunos problemas son más sencillos que otros. (Miguel, n.d.)

Los dos focos que utiliza son el tiempo y espacio, haciendo referencia al número de pasos de un algoritmo para resolver un problema y la cantidad de memoria que se utiliza para resolverlo respectivamente.

Los problemas de decisión en la teoría de complejidad de dividen en diferentes clases:

- *P*
- *NP*
- *NP-COMPLETE*
- *NP-HARD*

Los problemas P son aquellos que una máquina de Turing puede resolver en un tiempo polinómico, esto quiere decir que son tratables y a nivel de tiempo computacional son razonables por otro lado los NP son resueltos por maquinas no determinísticas de Turing pero conservando un tiempo polinómico (Miguel, n.d.).

Los problemas NP-complete pertenecen a NP y son más difíciles de solucionar ya que es menos probable encontrar una solución en un tiempo polinómico en cambio los NP-hard son los más

sencillos de este grupo NP sin embargo la complejidad de estos está en que el tiempo de los algoritmos de solución crecen exponencialmente a medida en que el tamaño del problema aumenta (Miguel, n.d.).

3.2.4. Herramienta computacional

GAMS: General Algebraic Modeling System, es un sistema de modelado de alto nivel para programación matemática y optimización. Está diseñado para resolver problemas de optimización lineales, no lineales y problemas de optimización entera mixta. Útil para aplicaciones de modelado complejas y de gran escala, además, permite al usuario construir modelos sostenibles, que se pueden adaptar a diferentes situaciones (Lee & Su, 2014).

MATLAB: Es un lenguaje de alto nivel y un entorno interactivo para el cálculo numérico, la visualización y la programación. Permite analizar datos, desarrollar algoritmos y crear modelos o aplicaciones. El lenguaje, las herramientas y las funciones matemáticas integradas, le permiten explorar múltiples enfoques y llegar a soluciones más rápidas que al trabajar con hojas de cálculo o lenguajes de programación tradicionales, como C/C++ y Java (Moore, 2007).

3.2.5. Optimización. Es la operación mediante la cual se busca mejorar, dar respuesta, o hacer eficiente una tarea o proceso , en los problemas de optimización principalmente se busca maximizar o minimizar la política a seguir , es decir, el conjunto de valores que dan respuesta a la función del problema.(Baquela & Redchuck, 2013).

Estos problemas tienen 3 componentes importantes: un conjunto de restricciones, conjunto de soluciones factibles que hace referencia a todas las posibles combinaciones de variables independientes y por último esta la función objetivo que involucra las variables del sistema con su performance (Baquela & Redchuck, 2013).

3.2.6. Métodos de solución. Gracias a la variedad que existe dentro de los problemas de optimización, existen diferentes métodos de solución en donde estos se clasifican según la continuidad de las variables es decir si toman cualquier valor dentro del conjunto de los reales, ya que existen problemas que poseen variables discretas en donde los valores solo pueden ser números enteros y existen otros en donde se encuentran variables de los dos tipos, estos se conocen como problemas de optimización mixta (Baquela & Redchuck, 2013).

Es por esto por lo que se habla de 3 posibles caminos para dar solución a los problemas de optimización: Mediante cálculos, técnicas de búsquedas y técnicas de convergencias de soluciones. Dada la naturaleza del problema que se está trabajando, la manera de resolverlo es a través de técnicas de búsqueda que consiste en buscar soluciones cercanas o factibles, o buscar la solución óptima, esto se puede hacer por medio de métodos exactos como la programación lineal entera mixta o algoritmos de solución.

3.2.6.1. Algoritmo genético. Está dado para resolver problemas de optimización y búsqueda, se basa en el proceso genético de los seres vivos (Moujahid & Inza, 2008). El algoritmo genético es una técnica de búsqueda basada en la teoría de la evolución de Darwin mediante una población de cromosomas codificados hacia los valores óptimos de la solución del problema (Kelly & Lawrence, 1991).

Esta técnica se basa en los mecanismos de selección que utiliza la naturaleza, de acuerdo a los cuales los individuos más aptos de una población son los que sobreviven, al adaptarse más fácilmente a los cambios que se producen en su entorno. Estos cambios se efectúan en los genes de un individuo (unidad básica de codificación de cada uno de los atributos de un ser vivo), y que sus atributos más deseables (i.e., los que le permiten adaptarse mejor a su entorno) se transmiten a sus descendientes cuando éste se reproduce sexualmente.

Los Algoritmos Genéticos (AGs) son métodos adaptativos que pueden usarse para resolver problemas de búsqueda y optimización. Están basados en el proceso genético de los organismos vivos. A lo largo de las generaciones, las poblaciones evolucionan en la naturaleza de acorde con los principios de la selección natural y la supervivencia de los más fuertes, postulados por Darwin.

Por imitación de este proceso, los Algoritmos Genéticos son capaces de ir creando soluciones para problemas del mundo real. La evolución de dichas soluciones hacia valores óptimos del problema depende en buena medida de una adecuada codificación de estas.

Un algoritmo genético consiste en una función matemática o una rutina de software que toma como entradas a los ejemplares y retorna como salidas cuáles de ellos deben generar descendencia (padres) para la nueva generación (hijos) (Holland, 2012).

- Codificación. Se supone que los individuos (posibles soluciones del problema), pueden representarse como un conjunto de parámetros (genes), los cuales agrupados forman una ristra de valores (a menudo referida como cromosoma). En términos biológicos, el conjunto de parámetros representando un cromosoma particular se denomina fenotipo. El fenotipo contiene la información requerida para construir un organismo, el cual se refiere como genotipo. Los mismos términos se utilizan en el campo de los Algoritmos Genéticos. La adaptación al problema de un individuo depende de la evaluación del genotipo. Esta última puede inferirse a partir del fenotipo, es decir puede ser computada a partir del cromosoma (Ver *Figura 4*), usándola función de evaluación.

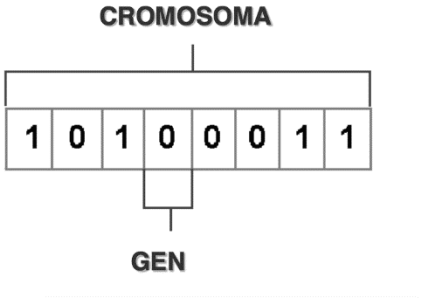


Figura 4. Ejemplo básico de cromosoma**.**

La función de adaptación debe ser diseñada para cada problema de manera específica. Dado un cromosoma particular, la función de adaptación le asigna un número real, que se supone refleja el nivel de adaptación al problema del individuo representado por el cromosoma (Gen, 2012).

- Operadores del algoritmo genético:

  - ***Selección***

La selección determina que individuos son elegidos para la recombinación y cuántos descendientes produce cada individuo. La función de aptitud es una función que mide la idoneidad de un individuo. El primer paso es el cálculo del fitness y por el método con el que se hace:

- Proporcional (proportional fitness assignment)

- Según el rango obtenido (rank-based fitness assignment)

- Según el valor obtenido en combinación de los distintos objetivos (multi-objective ranking)

La actual selección se mejora en el siguiente paso. Los padres son seleccionados según sus aptitudes por uno de los siguientes métodos:

- Selección “ruleta” (roulette-wheel selection)

- Muestreo estocástico universal (stochastic universal sampling)

- Selección local (local selection)

- Selección truncada (truncation selection)

- Torneo para selección (tournament selection)

- ***Cruce o recombinación:***

La recombinación produce nuevos individuos combinando la información contenido en los padres (acoplamiento de la población-padres). Dependiendo de la representación de las variables de los individuos, los siguientes algoritmos pueden ser aplicados:

- Todas representadas (All presentation):

Recombinación discreta

- Recombinación por valor real (Real valued recombination):

Recombinación intermedia

Recombinación lineal

Recombinación extendida lineal

Recombinación de valores binarios

Punto simple/ punto doble/ cruce-multipuntual

- Cruce uniforme

Cruce aleatorio (Shuffle crossover)

Cruce con sustituto reducido

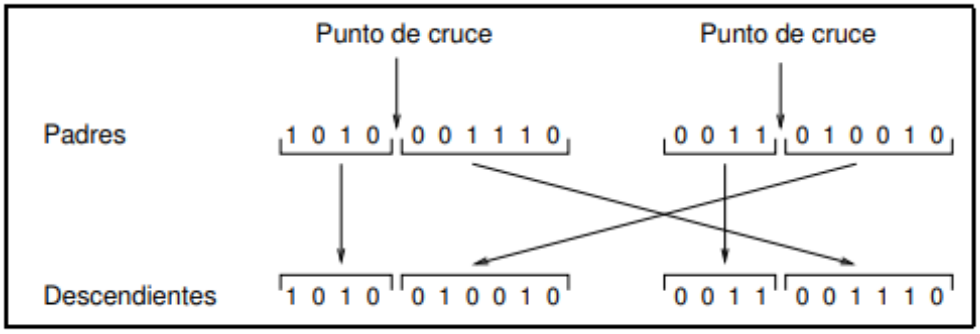


Figura 5. Ejemplo de operador de cruce basado en un punto. Adaptado de Algoritmos genéticos (Moujahid & Inza, 2008).

- ***Mutación:***

Después de la recombinación, cada descendiente sufre una mutación. Las variables de los descendientes son mutadas por pequeñas perturbaciones (tamaño del paso de la mutación), con baja probabilidad. La representación de las variables determina el uso del algoritmo. Dos operadores son relevantes:

- Operador de mutación para los valores reales de las variables

- Mutación para valores de variables binarias

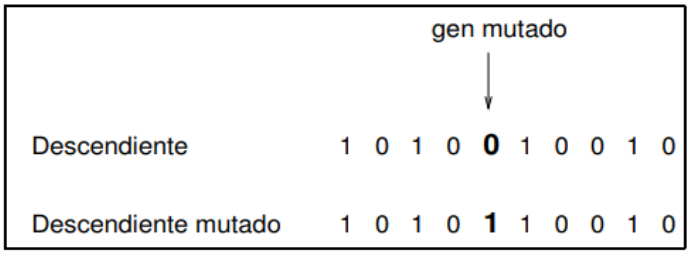


Figura 6. Operador de mutación. Adaptado de Algoritmos genéticos (Moujahid & Inza, 2008).

```
BEGIN /* Algoritmo Genetico Simple */
    Generar una poblacion inicial.
    Computar la funcion de evaluacion de cada individuo.
    WHILE NOT Terminado DO
    BEGIN /* Producir nueva generacion */
        FOR Tamaño poblacion/2 DO
        BEGIN /*Ciclo Reproductivo */
            Seleccionar dos individuos de la anterior generacion,
            para el cruce (probabilidad de seleccion proporcional
            a la funcion de evaluacion del individuo).
            Cruzar con cierta probabilidad los dos
            individuos obteniendo dos descendientes.
            Mutar los dos descendientes con cierta probabilidad.
            Computar la funcion de evaluacion de los dos
            descendientes mutados.
            Insertar los dos descendientes mutados en la nueva generacion.
        END
        IF  la poblacion ha convergido THEN
            Terminado := TRUE
    END
END
```

Figura 7. Pseudocódigo del Algoritmo Genético Simple. Adaptado de Algoritmo Genético Simple (Gen, 2012).

3.2.6.2. Optimización por enjambre de partículas (PSO). La optimización del enjambre de partículas es un método heurístico de búsqueda iterativa y colectiva, con énfasis en la cooperación, presentado originalmente por Kennedy y EBerhart en 1995, es un algoritmo del área de la inteligencia artificial de la rama de inteligencia de enjambres. Está inspirado en el comportamiento social de los seres vivos. Originalmente diseñado para funciones matemáticas no-lineales continuas. Básicamente, PSO trabaja con un conjunto de soluciones candidatas denominado enjambre. Cada miembro del enjambre es una partícula, y esta posee un vector de solución denominado posición. Cada partícula conoce la mejor posición encontrada por el enjambre o mejor global. Si se definen subconjuntos de partículas del enjambre, cada uno de estos se denomina vecindad. Para este caso, cada partícula conoce la mejor posición de su propio vecindario o mejor

local. Una vecindad define como las partículas interactúan entre ellas, y determina como la información se propaga en el enjambre afectando la convergencia del algoritmo. El comportamiento (dirección y velocidad) de cada individuo es el efecto de influencias cognitivas, sociales y estocásticas. El objetivo común de todos los miembros de la población consiste en encontrar la ubicación más favorable dentro de un determinado espacio de búsqueda (Lovay, Romero, & Peretti, 2016).

El método de optimización por enjambre de partículas se ha vuelto popular en los últimos años, dado a su eficiencia y bajo costo computacional (Dodge & Lavalle, 2016).

- **Características del algoritmo**
  - **Cognitiva y social:** La cognitiva contribuye para que la partícula tenga una especie de memoria y pueda saber si anteriormente había adoptado una mejor posición. El factor social, es el responsable de que la partícula sea influenciado por otras partículas en una mejor posición, provocando ser atraída al óptimo encontrado.
  - **Topología de gbest y pbest:** El concepto del modelo gbest es que todas las partículas se comunican entre sí y el mejor punto encontrado, es comunicado a las demás partículas. Tiene la ventaja de ser muy rápido para encontrar un valor óptimo. El pbest indica el mejor valor histórico que ha tenido la partícula a lo largo de las generaciones.
  - **Inercia:** La inercia es un factor que se añadió para dar mayor estabilidad a la fórmula, permite que la partícula siga una trayectoria constante ignorando la influencia de otros, esto con el fin de evitar valores atípicos en los resultados.

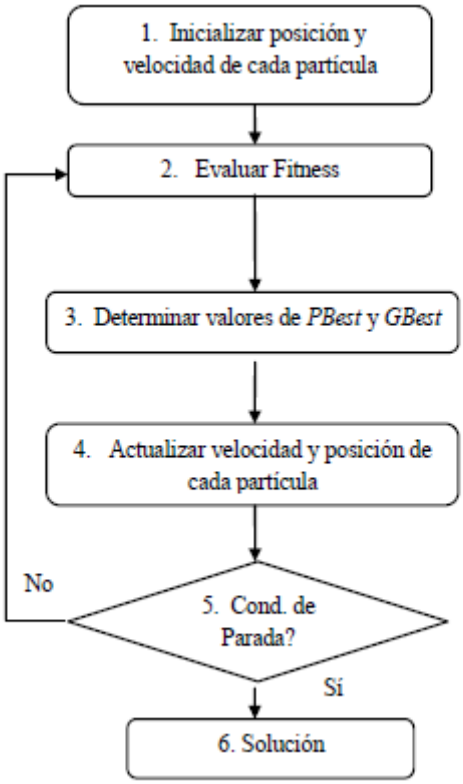


Figura 8. Diagrama de flujo algoritmo PSO básico. Adaptado de Aplicación del algoritmo de Optimización por Enjambre de Partículas en el dimensionamiento óptimo de componentes para Filtros (Lovay, Romero, & Peretti, 2016).

- **Fórmulas del PSO básico*:***

En una iteración t, cada partícula Pi conoce su posición actual $X_i(t) = \{ x_{i1}, \dots, x_{in}\}$. la velocidad con la cual alcanzó su posición actual $V_i(t) = \{ v_{i1}, \dots, v_{in}\}$. y la mejor posición por ella encontrada hasta ese momento $b_1 = \{ b_{i1}, \dots, b_{in}\}$. Todas las partículas del enjambre conocen la mejor posición global g = $\{ g_{i1}, \dots, g_{in} \}$. Si se definieron vecindades, entonces cada partícula conoce también la mejor posición local l = $\{l_{i1}, \dots, l_{in} \}$. Así, a cada iteración t, las partículas se actualizan conforme a:

$$\mathrm{V_i(t)} = \omega * \mathrm{V_i(t)} + \gamma\alpha_1 * \theta_1 \left(\mathrm{b_i} - \mathrm{X_i(t)}\right) + \gamma\alpha_2 * \theta_2 \left(\mathrm{g} - \mathrm{X_i(t)}\right) + \gamma\alpha_3 * \theta_3 \left(\mathrm{l} - \mathrm{X_i(t)}\right)$$

$$X_i(t+1) = X_i(t) + V_i(t+1)$$

Donde ω es conocido como inercia y se utiliza para controlar la influencia del valor anterior de la velocidad. Por su parte $\gamma\alpha_1, \gamma\alpha_2$ y $\gamma\alpha_3$ son números aleatorios uniformemente distribuidos en el intervalo [0,1], mientras que θ_1, θ_2 y θ_3 son parámetros conocidos como constantes de aceleración (Chamorro, Villagra, & Baran, 2014).

Estas ecuaciones representan que cada partícula tendrá una posición que cambiara de acuerdo con su velocidad, la cual está condicionada a su histórico, su mejor posición y la mejor posición del grupo, cabe resaltar que estas ecuaciones solo son válidas en un espacio continuo, por lo que, investigadores han adaptado este algoritmo a sus problemas, creando variantes del PSO, tales como: MOPSO (Optimización por enjambre de partículas multiobjetivo) , DPSO (Optimización por Enjambre de partículas Discreto), entre otros.

4. Modelo matemático para el DVCFP

La formulación matemática propuesta para abordar el DVCFP consiste en la adaptación del modelo matemático presentado por Lv, Yuan, & Han (2013), el cual fue modificado de acuerdo a consideraciones de costos. En este modelo de naturaleza lineal entera mixta se busca minimizar los costos asociados al proceso de manufactura bajo parámetros de variación de demanda, diferentes rutas y tiempos de procesamiento asociados, junto con restricciones de balance de carga para celdas virtuales.

Supuestos:

- La demanda para cada tipo de parte es determinística, conocida para cada periodo y debe ser cubierta en su totalidad, no se admite pedido pendiente ni inventario.
- El costo de operación es conocido, el cual es dependiente de la carga de trabajo de cada máquina.
- El tiempo disponible en cada tipo de máquina es conocido y constante para todos los periodos del horizonte de planeación.
- Se asume que las máquinas son multipropósito. No se considera averías ni compra de maquinaria.
- Las máquinas se encuentran ya distribuidas previamente y disponibles desde el comienzo del horizonte de planeación.
- El transporte de los productos se realiza por lotes, los cuales corresponden a la cantidad demandada.
- El costo unitario de transferencia de materiales entre celdas y entre máquinas es conocido y constante para todos los periodos. Este es independiente de la distancia, debido a que se considera que las partes son transportadas por máquinas automatizadas (robots, bandas transportadoras, grúas, camiones o vehículos AGV).
- No se considera el valor del dinero a lo largo del tiempo entre periodos.

Notación:

- **Índices**

  h = Periodos (h=1,…, H).

  c= Celdas de manufactura (c=1,…,C).

  m = Tipos de máquina (m=1,…,M).

p = Tipos de partes (p=1,…,P).

j = Operación que pertenece a cada parte p (j=1,…,Op).

s= Índice para el tamaño de la celda (s=1,…,S).

- **Parámetros**

Dph: Demanda de la parte p para el periodo h

β_m: Costo de operación para cada máquina m

K_s: Costo interno de producción cuando la celda es de tamaño s (el número de máquinas es igual a s), este se supone como una función no decreciente de s, es decir, cuantas más maquinas estén asociadas en una celda mayor serán los costos de alistamiento, inventario de producto en proceso, coordinación y manejo.

Z: Costo unitario de transporte entre celdas.

U: Costo unitario de transporte entre máquinas.

t_{jom}: Tiempo de procesamiento de la operación j en la máquina m para producir la parte p.

a_{jpm}: igual a 1, si la operación j de la parte p puede ser realizada en la máquina tipo m; de lo contrario, 0.

CAP_m: Capacidad en tiempo de la maquina tipo m en cada periodo

LI: Mínimo número de máquinas m en cada celda

LS: Número máximo de máquinas m en cada celda

Q_m: Número de máquinas tipo m disponibles

F: Factor de balance de carga entre celdas virtuales, este valor se mide a criterio del investigador, teniendo en cuenta el valor (porcentaje) mínimo de producción por cada una de las celdas.

- **Variables de decisión**

  N_{mch}: Número de máquinas tipo m en la celda c para el periodo h

  X_{jpmch}: 1 Si la operación j de la parte p es procesada en la celda c para el periodo h, 0 de lo contrario

  Y_{chs}:1 si la celda c es de tamaño s en el periodo h, o de lo contrario

Formulación matemática

$$\sum_{h}^{H}\sum_{c}^{C}\sum_{m}^{M}\sum_{p}^{P}\sum_{j}^{Op}\beta_m * t_{jpm} * X_{jpmch} * D_{ph} + \sum_{h}^{H}\sum_{c}^{C}\sum_{s}^{S}\kappa_s * Y_{chs} + \sum_{h}^{H}\sum_{p}^{P}\sum_{j}^{Op}\sum_{c}^{C}\left| z * D_{ph}\left(\sum_{m}^{M} X_{(j+1)pmch} - \sum_{m}^{M} X_{jpmch}\right)\right| + \sum_{h}^{H}\sum_{p}^{P}\sum_{j}^{Op}\sum_{m}^{M}\sum_{c}^{C} U * D_{ph} * \left(X_{(j+1)pmch} * \sum_{v\,(v=m)}^{M} X_{jpvch}\right) \quad (1)$$

Sujeto a:

$$\sum_{c}\sum_{m} a_{jpm} * X_{jpmch} = 1 \;\; \forall j, p, h \quad (2)$$

$$X_{jpmch} \leq a_{jpm} \;\; \forall j, p, m, c, h \quad (3)$$

$$\sum_{p}\sum_{j} D_{ph} * t_{jmp} * X_{jpmch} \leq N_{mch} * CAP_m \;\; \forall m, c, h \quad (4)$$

$$\sum_{c} N_{mch} \leq q_m \quad \forall m, h \tag{5}$$

$$\sum_{m} N_{mch} \leq LS \quad \forall c, h \tag{6}$$

$$\sum_{m} N_{mch} \geq LI \quad \forall c, h \tag{7}$$

$$\sum_{m}^{M} N_{(m,c,h)} - \sum_{s}^{S} COEF_s * Y_{(c,h,s)} = 0 \ \forall c, h \tag{8}$$

$$\sum_{s} Y_{chs} = 1 \quad \forall h, c \tag{9}$$

$$\sum_{j}\sum_{m}\sum_{p}(t_{jpm} * X_{jpmch} * D_{ph}) \geq \frac{q}{c}\sum_{j}\sum_{m}\sum_{p}\sum_{w(w=c)} t_{jpm} * X_{jpmch} * D_{ph} \quad \forall c, h \tag{10}$$

$$X_{jpmch}, Y_{chs} \in [0,1] \ \textit{binario} \ \forall j, p, m, c, h, s \tag{11}$$

$$N_{mch} \in \mathbb{Z} \quad \forall m, \forall c, \forall h \tag{12}$$

El modelo no lineal propuesto, busca minimizar el costo total asociado a la producción, compuesto por cuatro términos, el primero calcula el costo total de operación de todas las maquinas requeridas en todas las celdas durante el horizonte de planeación, es decir, la suma de la carga de trabajo por producto asignado a cada tipo de máquina en cada celda y su costo asociado, el segundo término representa el costo interno de producción en cada celda, esto es el costo de alistamiento de las máquinas, el mantenimiento de inventario en proceso y coordinación, es decir, cuánto más

máquinas se encuentren formando una celda, mayor será el valor de los costos asociados; el tercer término calcula el costo de transportar material entre dos operaciones consecutivas que se encuentran en diferentes celdas y el cuarto término evalúa el costo de transportar productos en proceso entre dos máquinas diferentes cuyas operaciones son consecutivas.

Por otro lado, el conjunto de restricciones de la ecuación 2 y 3 limita la asignación de cada operación de cada tipo de producto a una sola celda y máquina dentro del sistema. El conjunto de restricciones de la ecuación 4 garantiza que las capacidades de cada tipo de máquina en cada celda del sistema no sean excedidas y a su vez, puedan satisfacer los requerimientos de demanda. La ecuación 5 limita el número de tipos de máquinas disponibles en cada periodo, de tal manera que no se asignen más máquinas de las que se encuentran en el sistema. El conjunto de restricciones de la ecuación 6 y 7 determinan que el número de máquinas en cada celda debe ser un valor entre el mínimo y máximo establecido. El conjunto de restricciones 8, consisten en ecuaciones para la programación correcta del modelo matemático, debido a que representa la activación de la variable binaria $Y_{(c,h,s)}$ ligada al valor de la variable entera $N_{(m,c,h)}$. La ecuación 9 asigna que cada celda solo puede tener un valor constante en el periodo de planeación, es decir, solo una cantidad de máquinas correspondiente por periodo. La ecuación 10 representa el conjunto de restricciones que busca el balance de carga en las celdas, es decir, limita a que cada celda debe producir al menos un porcentaje (q) de la carga total, esto con el fin de evitar que algunas celdas queden sobrecargadas, mientras otras se encuentran sin producir.

Finalmente, las ecuaciones 11 y 12, destacan la naturaleza de las variables de decisión dentro del sistema.

- ***Linealización del modelo:***

Tal como se observa en la ecuación 13, la formulación de la función objetivo propuesta es de naturaleza no lineal entera mixta, debido a las expresiones en valor absoluto (costos de intercelulares y costos intracelulares) que allí se encuentran.

Para el presente trabajo, se toma como base las investigaciones realizadas por Ahkioon, Bulgak, & Bektas (2009) con el fin de transformar las expresiones de la función objetivo para trabajar el problema como un modelo de programación lineal entera mixta. Para ello, se propone la introducción de las variables auxiliares como se muestra a continuación:

- ***Linealización de costos intercelulares:***

$$\sum_{h}^{H}\sum_{p}^{P}\sum_{j}^{Op}\sum_{c}^{C}\left|\alpha_p * \left(\sum_{m}^{M} X_{(j+1)pmch} - \sum_{m}^{M} X_{jpmch}\right)\right|$$

$$\left|X_{(j+1)pmch} - X_{jpmch}\right| = YP_{jpmch} + YM_{jpmch} \quad (13)$$

Sujeto a:

$$X_{(j+1)pmch} - X_{jpmch} = YP_{jpmch} + YM_{jpmch}\ \forall j, p, m, c, h \quad (14)$$

En este caso se introducen las variables auxiliares YP_{jpmch} y YM_{jpmch}, estas, representan las transferencias intercelulares del producto tipo p desde la maquina m de la celda c entre las operaciones j y (j+1) para el periodo h, incorporando las restricciones asociadas (ecuaciones 13 y 14), para así, reformular el tercer rubro de la función objetivo como se muestra a continuación:

$$\sum_{h}^{H}\sum_{p}^{P}\sum_{j}^{Op}\sum_{c}^{C}\sum_{m}^{M} Z_p\ (YP_{jpmch} + YM_{jpmch}) \quad (15)$$

Hillier & Lieberman (2002, pág. 334) lo definen de la siguiente manera,

$$YP_{jpmch} = \begin{cases} 1, & \text{si } X_{(j+1)pmch} - X_{jpmch} = 1 \\ 0; & \text{De lo contraario} \end{cases}$$

$$YM_{jpmch} = \begin{cases} 1, & \text{si } X_{(j+1)pmch} - X_{jpmch} = -1 \\ 0; & \text{De lo contrario} \end{cases}$$

Por lo tanto, la expresión siempre será estrictamente positiva, en cualquiera de los dos casos.

Aun así, este enunciado presenta un inconveniente, la doble contabilización de los costos, debido a que las variables toman en cuenta el costo de realizar la operación j y (j+1) como dos costos diferentes, aunque el transporte realizado es solo hacia una dirección, por lo tanto, para evitar este problema, se toma solo una de las dos variables para el desarrollo del modelo como se muestra a continuación, utilizando las ecuaciones 12 y 14:

$$\sum_{h}^{H}\sum_{p}^{P}\sum_{j}^{Op}\sum_{c}^{C}\sum_{m}^{M} z_p * YP_{jpmch} \quad (16)$$

$$X_{(j+1)pmch} - X_{jpmch} = YP_{jpmch} + YM_{jpmch}\ \forall j, p, m, c, h$$

$$YP_{jpmch}, YM_{jpmch} \in [0,1]\ \textit{binario}\ \forall j, p, m, c, h$$

- ***Linealización de los costos intracelulares:***

El quinto término de la función objetivo denota una multiplicación entre dos variables, lo cual se convierte en un término no lineal. Para los costos de transporte entre dos máquinas de la misma

celda, cuyas operaciones son consecutivas, Ahkioon, Bulgak, & Bektas (2009)**,** plantean la utilización de la variable lineal auxiliar O_{jpmvch}, sujeto a las restricciones 18 y 19. El índice v utilizado en la ecuación 17, cumple la misma función que el índice m (máquinas), pero, para calcular los costos intracelulares es necesario que funcionen de manera independiente, así es posible comparar si para un mismo producto se realizan dos operaciones consecutivas en la misma máquina o en otra diferente.

$$\sum_{h}^{H}\sum_{p}^{P}\sum_{j}^{Op}\sum_{m}^{M}\sum_{c}^{C}\gamma_p\ (X_{(j+1)pmch} * \sum_{v\ (v=m)}^{M} X_{jpvch}) \tag{17}$$

$$O_{jpmvch} = X_{(j+1)pmch*} X_{jpvch} \tag{18}$$

Sujeto a:

$$O_{jpmvch} \geq X_{(j+1)pmch} - X_{jpmch} + X_{jpvch} - 1\ \forall j, p, m, v, c, h \tag{19}$$

Para explicar la formulación matemática propuesta, Ahkioon et al. (2009), consideran dos casos:

(i). Si $X_{(j+1)pmch} * X_{jpvch} = 0$, bajo esta situación se pueden considerar 3 sub-casos:

(a). $X_{(j+1)pmch} = 1$, $X_{jpvch} = 0$

(b) $X_{(j+1)pmch} = 0$, $X_{jpvch} = 1$

(c) $X_{(j+1)pmch} = 0$, $X_{jpvch} = 0$

Según la ecuación (15), $O_{jpmvch} \geq -1$, y debido a que la variable está restringida a valores positivos, al minimizar la función objetivo se asegura que $O_{jpmvch} = 0$.

(ii) Si $X_{(j+1)pmch} * X_{jpvch} = 1$, *indica que* $X_{(j+1)pmch} = X_{jpvch} = 1$, *por lo tanto,* $O_{jpmnch} \geq 1$.

De tal manera que, la formulación propuesta es equivalente al rubro no lineal de la función objetivo, sin embargo, por consideraciones en cuanto a los paquetes de instancias utilizados para la validación del modelo en los softwares informáticos utilizados (GAMS CLPX), es posible que dos operaciones consecutivas puedan ser elaboradas en una misma máquina, para solucionar esto, se propuso agregarle a la ecuación (15) la variable X_{jpmch}, de tal manera que no se considere el costo intracelular cuando la operación consecutiva (j+1) de la parte p se realice en la misma máquina m donde se realizó la operación j, en la celda c del periodo h.

De tal manera que el cuarto rubro de la función objetivo será lineal bajo la siguiente formulación, junto con las ecuaciones 18 y 19:

$$\sum_{h}^{H}\sum_{p}^{P}\sum_{j}^{Op}\sum_{m}^{M}\sum_{c}^{C}\sum_{(v=m)}^{M} \upsilon_p * O_{jpmvch} \qquad (20)$$

$$O_{jpmvch} = X_{(j+1)pmch*} X_{jpvch} \ \forall j, p, m, v, c, h$$

$$O_{jpmvch} \geq X_{(j+1)pmch} - X_{jpmch} + X_{jpvch} - 1 \ \forall j, p, m, v, c, h$$

$$O_{jpmvch} \in [0,1] \textit{ binario } \forall j, p, m, v, c, h, s$$

5. Desarrollo de los algoritmos

5.1. Instancias generadas para la validación del modelo matemático

Para la validación del modelo propuesto se utilizan 2 diferentes conjuntos de tamaños de instancias clasificados de acuerdo a la cantidad de partes y máquinas a tener en cuenta (grandes y pequeñas), cada uno de estos conjuntos se constituye de 5 conjuntos de datos (para un total de 10 instancias evaluadas), generados cada uno aleatoriamente según la adaptación de los parámetros estipulados por Mungwattana, Ellis, & Sturges (2000) como se observa en la Tabla 3. Estos parámetros son programados en el software MATLAB.

Tabla 3.
Parámetros aleatorios para las instancias.

Parámetro	***Valor***
Número de operaciones	U (2,6)
Demanda/periodo	U (500,1000)
Tiempo de procesamiento	U (0.1,1.0)
Costo de operación	N ($0,83, $0,23)

Adicionalmente, para todos los conjuntos de instancias se establecen los mismos parámetros correspondientes al tamaño máximo y mínimo de celda, capacidad en tiempo de las máquinas y número de celdas permitido. Los valores de los costos inter e intracelulares son tomados de Nsakanda, Diaby, & Price (2006), el factor de balance de carga de las celdas virtuales, es tomado de las instancias presentadas por Lv, Yuan, & Han (2013) (Ver Tabla 4).

Tabla 4.
Parámetros constantes para los paquetes de instancias

Parámetro	**Valor**
Capacidad	7.000
Número de celdas permitido	3

Continuación Tabla 4. *Parámetros constantes para los paquetes de instancias*

Número mínimo de máquinas por celda	**2**
Número máximo de máquinas por celda	10
Costo de transporte intracelular	$1
Costo de transporte intercelular	$3
Factor de balance de carga entre celdas	0,9

En la Tabla 5 se observan los tamaños de instancias utilizados para la generación de los datos aleatorios.

Tabla 5.
Caracterización de los tamaños de paquetes de instancias

Tamaño	Partes	Máquinas	Operaciones
Alto	20	13	2-6
Bajo	14	9	2-6

Los 2 conjuntos de instancias son utilizados por el modelo matemático para la implementación en GAMS/CLPX y el algoritmo GAPSO del mismo, con el fin de obtener resultados en tema de costos y evaluar el tiempo computacional utilizado por cada uno de ellos.

5.2 Desarrollo del Modelo MILP en el software GAMS

La formulación matemática propuesta se desarrolla como un modelo de programación lineal entera mixta (MILP) utilizando el software GAMS/CPLX.

Todo programa en GAMS está constituido por un conjunto de bloques que especifican diferentes características del modelo, los cuales son: conjuntos, parámetros, variables, ecuaciones y sentencias del modelo y del “solve”.

Conjuntos:

Los conjuntos se introducen en GAMS dentro de sentencias que empiezan con la palabra clave "sets" y especifican los índices relacionados a la formulación matemática del problema, los subconjuntos pj y pja se definen en función de los índices primarios e indican la dimensión de la secuencia de fabricación de cada uno de los productos.

Adicionalmente se utiliza la función "alias" de GAMS, el cual es un comando ligado a la definición de conjuntos y permite dar varios nombres equivalentes al mismo conjunto, es decir, el índice w da un segundo nombre al índice c (índice para las celdas de manufactura), debido a que en el modelo los dos conjuntos se encuentran implicados en diferentes sumas y/o productos, pero necesitan variar independientemente. En la figura 9. Se muestran los índices utilizados para el DVCFP.

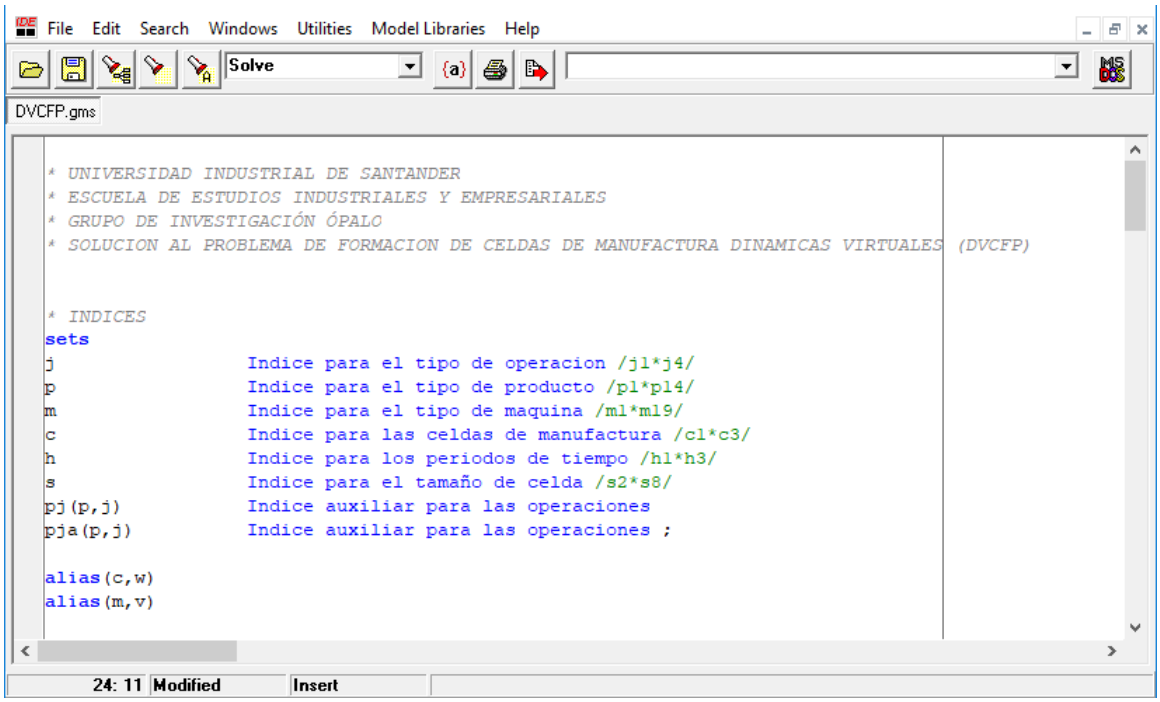


Figura 9. Declaración de índices para el DVCFP. Adaptado del software GAMS 23.9.5

Parámetros:

Los parámetros de la programación son los datos de entrada necesarios para la solución de este y se definen mediante el comando “parameters”, como se observa en la Figura 10.

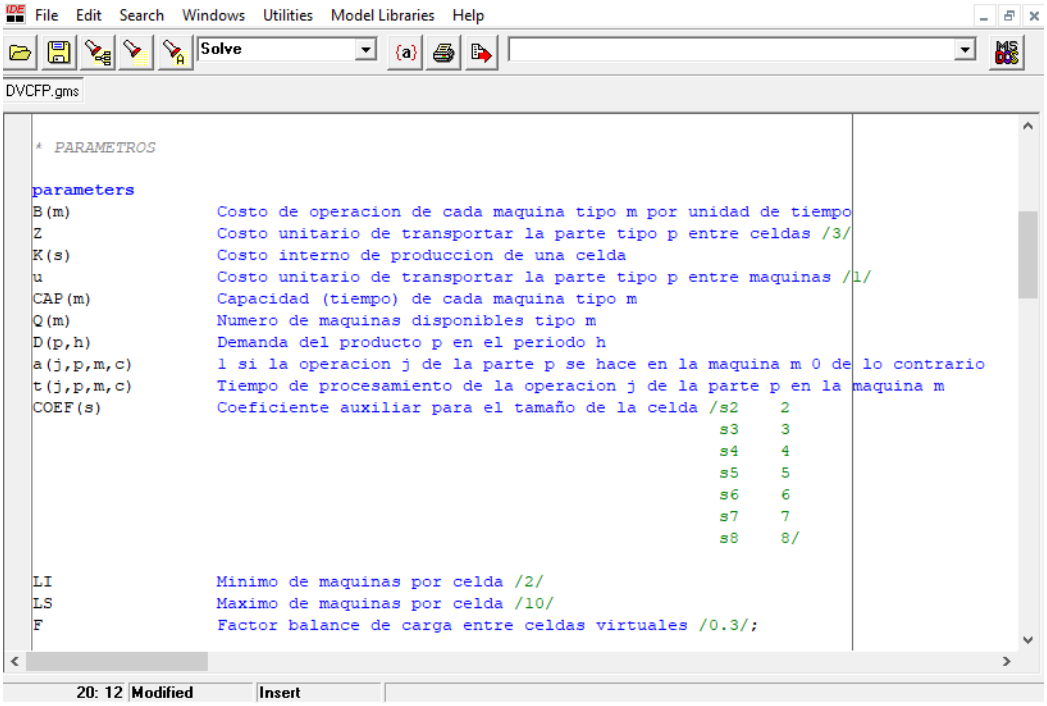


Figura 10. Declaración de parámetros para el DVCFP. Adaptado del software GAMS 23.9.5

Cabe resaltar que bajo esta declaración solo es posible entregar los datos en los que interviene un índice, cuando intervienen más, es posible utilizar el comando “table” (para dos índices) o cargarlos utilizando hojas de cálculo de Excel (donde pueden intervenir parámetros de dos o más índices) mediante la orden “call GDXXRW” como se muestra en la figura 11.

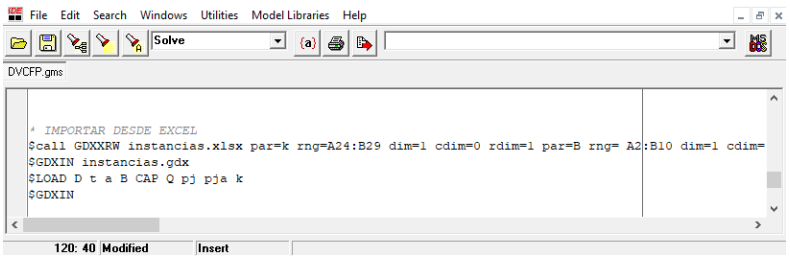


Figura 11. Declaración de parámetros cargados desde Microsoft Excel. Adaptado del software GAMS 23.9.5

Variables:

En este bloque se definen las variables que va a calcular GAMS con base en los parámetros y las ecuaciones especificadas, para resolver el DVCFP se utilizan tres tipos de variables:

- Variables enteras: Las cuales pueden tener valores solo en el intervalo (0, ∞), siempre y cuando sean valores discretos, en este caso para el número de máquinas por celda ($N_{,m,c,h}$).
- Variables positivas: Variables las cuales su valor puede ser cualquier número dentro del intervalo de (0, ∞), para el DVCFP se caracterizan como positivas las variables asociadas a los costos totales de operación (coop), costos de movimientos intercelulares (cointer), costos de movimientos intracelulares (cointra) y costos por tamaño de celda (codent).
- Variables binarias: Aquellas que toman valores de cero o uno, se nombran como binarias las variables de asignación de operaciones a celdas $X_{j,p,m,c,h}$ y de tamaño de celda $Y_{c,h,s}$, además de las variables auxiliares utilizadas para cuantificar los movimientos intra e intercelulares $YP_{j,p,m,c,h}$, $YM_{j,p,m,c,h}$ y $O_{j,p,m,v,c,h}$.

En la figura 12. Se muestra la definición de variables utilizada en el GAMS para el DVCFP.

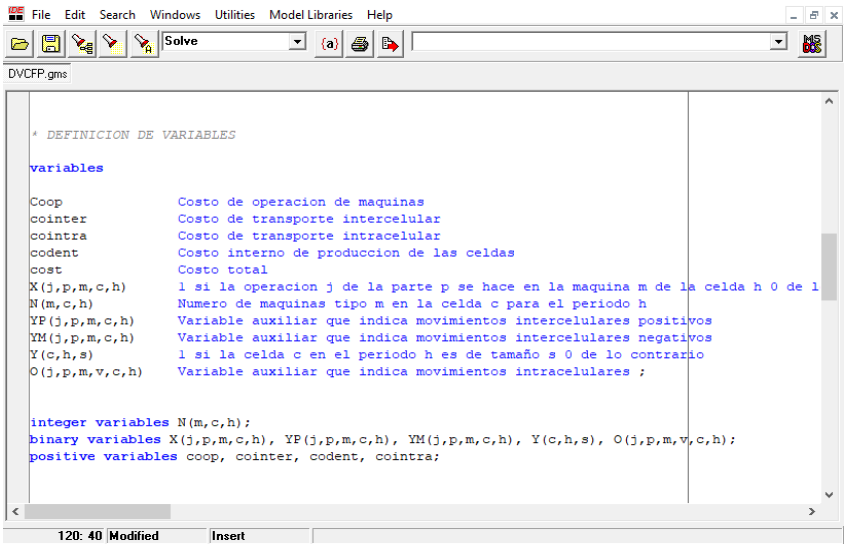


Figura 12. Declaración de variables para el DVCFP. Adaptado del software GAMS 23.9.5

Ecuaciones:

La función "equations" identifica el comando para definir las restricciones y la función objetivo en GAMS. En la figura 13 se observa las ecuaciones utilizadas para el modelo de DVCFP.

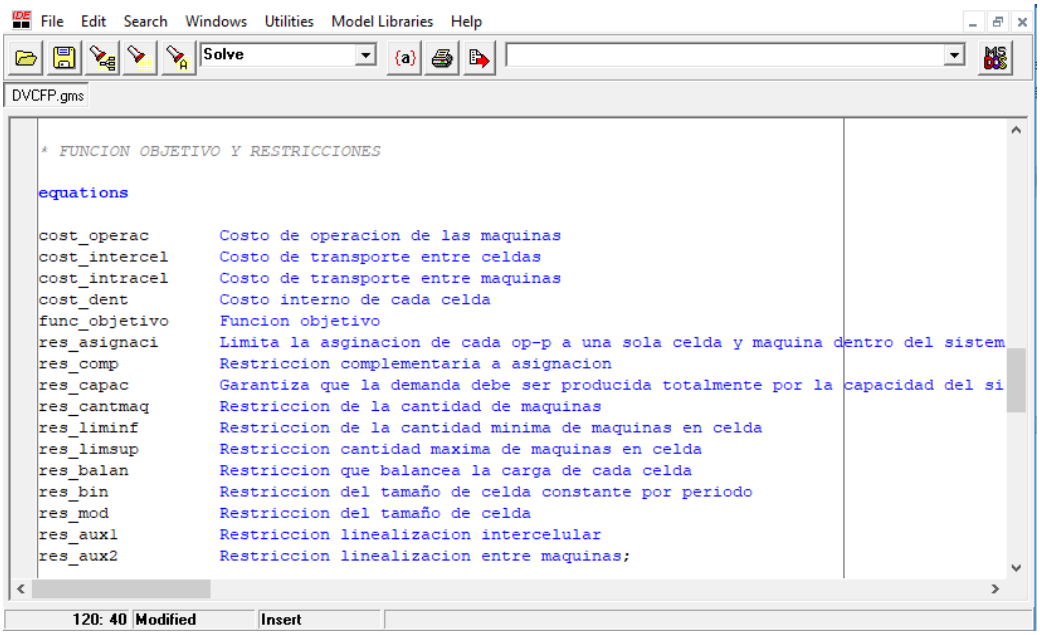


Figura 13. Declaración de ecuaciones. Adaptado del software GAMS 23.9.5

Sentencias del modelo y del solve:

Debido al gran tiempo computacional se utilizan los comandos iterlim y reslim limitando el tiempo de iteraciones y solución del GAMS a una hora (60 minutos) para todos los paquetes de instancias. Y finalmente se declara el modelo y el "solve" para dar solución al problema. En la figura 14, se observan las sentencias utilizadas para la implementación del modelo exacto en GAMS.

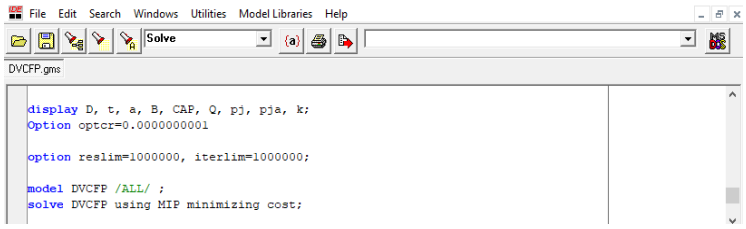


Figura 14. Declaración de sentencias del modelo y del solve. Adaptado del software GAMS 23.9.5

En el apéndice A se muestra el código completo utilizado para el uno de los conjuntos de instancias.

5.3 Algoritmo Genético Híbrido

El algoritmo GAPSO propuesto consiste en la utilización de la Optimización por Enjambre de Partículas Discreta (DPSO) en una de las etapas del Algoritmo Genético (GA), con el fin de refinar los resultados que la metaheurística se encuentra evaluando.

5.3.1. Parámetros de entrada. Los parámetros requeridos para el funcionamiento de los algoritmos son tomados de Paydar & Saidi-Mehrabad (2017) para el AG y Rezazadeh, Mahini, & Zarei (2011) para el DPSO, los cuales proponen estos valores basados en los resultados experimentales de su documentos acerca del DVCFP. Para el algoritmo genético se establece la probabilidad de cruce y mutación, las cuales indican que el 90% de la población es cruzado y solo el 5% es mutado. Para el DPSO, se instauran 3 parámetros iniciales, la constante de inercia, que asegura que las partículas no se dispersan de manera abrupta en el espacio de búsqueda, y las constantes de aceleración cognitiva y social que indican la influencia de sí mismo y del enjambre en la evolución de la partícula a lo largo de las iteraciones en el algoritmo. En la Tabla 6 se muestran los valores utilizados en la implementación del GAPSO en MATLAB.

Tabla 6.
Parámetros utilizados para los algoritmos.

AG	***DPSO***
Probabilidad de cruce: 0,9	Coeficiente de inercia de la partícula (w): 1
Probabilidad de mutación: 0,05	Constante de aceleración cognitiva (c1): 1.5
	Constante de aceleración social(c2): 1.5

5.3.2. Parámetros de instancias. Para la solución del DVCFP mediante la metaheurística propuesta se realiza la adaptación de los 10 conjuntos de instancias previamente utilizados en el modelo exacto.

5.3.3. Representación de la solución. En el algoritmo GAPSO, cada solución o partícula en el espacio de búsqueda está representada por un cromosoma, para este trabajo se utiliza un vector de tamaño (H x$\sum_p Op$), donde H son los periodos y $\sum_p Op$ son la suma de todas las operaciones necesarias para la fabricación de todos los productos en cada uno de los periodos.

En la Figura 15, se presenta un ejemplo de vector solución propuesto para un periodo del horizonte de planeación, 3 productos, 7 máquinas, 2 celdas y la cantidad de operaciones necesarias para cada una de las partes. Cabe resaltar que la solución se encuentra en las filas 3 y 4, indicando qué máquina hará cada operación de cada parte en cada una de las celdas correspondientes.

El número de columnas (longitud) del vector está determinado por la cantidad de operaciones necesarias para producir cada tipo de producto, en los distintos periodos a considerar. Debido a que la generación de operaciones para cada parte es aleatoria (ver en la Tabla 3), a cada producto le corresponde diferente número de columnas; como se observa en la figura 15, en la fila de operaciones (fila 3), cada casilla está representada por el número de parte (tipo de producto) a la cual pertenece cada operación, es decir, para el producto número 1, el cual tiene 2 operaciones, le corresponden 2 columnas encabezadas por el número 1, para el producto número 2 con 3 operaciones, le pertenecen 3 columnas, por lo tanto, se ubica el número 2 repetido 3 veces, y así para cada una de las partes.

Periodo	h1								
Partes	p1	p1	p2	p2	p2	p3	p3	p3	p3
Operaciones	1	1	2	2	2	3	3	3	3
Máquinas	1	5	3	1	4	2	8	6	7
Celdas	1	2	1	1	2	1	2	1	2

Figura 15. Representación de vector solución para un periodo.

En la fila 4 (máquinas), se ubica aleatoriamente la máquina escogida para realizar la operación representada por la columna. Adicionalmente, se hace necesario generar una matriz de partes-operaciones-máquinas, la cual se utiliza para la generación de vectores factibles en la asignación, es decir, que en el momento en que se escoja la máquina que haga la operación designada, sea factible su relación. Esta matriz surge debido a la gran cantidad de infactibilidades resultantes que se generan cuando se aleatoriza la asignación de máquinas a las operaciones, por lo tanto, se hace necesario parametrizar la asignación de máquinas, de tal manera que el algoritmo garantice que la máquina que se va a seleccionar corresponda con las factibles y escoja solo las máquinas habilitadas para realizar cada operación.

Continuando con el ejemplo anterior, en la Figura 16 se puede observar un ejemplo de la matriz de partes-operaciones-máquinas, de tal manera en que para la primera operación de la parte 1, es posible escoger entre las máquinas 1,2 y 3 solamente, ya que en las demás no es posible realizar esta operación. Teniendo en cuenta lo anterior, el vector no asignará la máquina 5, para realizar esta operación (primera operación de la parte 1), y así sucesivamente con las demás operaciones partes.

	Operaciones								
Máquinas	1	1	2	2	2	3	3	3	3
	1	3	1	1	1	2	7	3	7
	2	4	2	4	2	3	8	4	0
	3	5	3	6	3	0	0	6	0
	0	0	0	7	4	0	0	0	0
	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0

Figura 16. Ejemplo corto de matriz partes-operaciones-máquinas.

Ahora bien, en la última fila (celdas) del vector, se ubican aleatoriamente números enteros entre la cantidad de celdas permitidas por periodo. Para el caso de la figura 15, se ubica aleatoriamente números entre 1 y 2, indicando que se permite la formación de 2 celdas por periodo, éste número determina en que celda se ubica la máquina que realiza la operación designada.

5.3.4. Función objetivo

La función fitness del algoritmo consiste en la adaptación del modelo matemático propuesto en la sección anterior a través del manejo de matrices que representan los costos y asignaciones del problema. Para las restricciones de capacidad, cantidad de máquinas por celda y balance de carga, se establecen penalizaciones de costos cuando se utiliza la máquina más tiempo del que es establecido como posible, si se asignan más o menos máquinas de las estipuladas en los parámetros y, cuando alguna celda tenga menos del porcentaje establecido como mínimo para la carga de trabajo del periodo, generando un costo alto, para que así el algoritmo descarte soluciones que generen altos costos y haga uso únicamente de soluciones factibles.

5.3.5. Desarrollo del algoritmo

El algoritmo genético híbrido se divide en 5 etapas: Generación de la población inicial, en la cual se generan los grupos de vectores solución de acuerdo a la cantidad especificada de población

inicial, continuando a la etapa de selección, donde los individuos destinados para la reproducción se eligen entre un número aleatorio de individuos o cromosomas, posteriormente se realiza el cruce, en el cual se seleccionan los genes de los cromosomas progenitores creando una nueva descendencia; debido a que para el funcionamiento de ambos algoritmos en conjunto, es necesario generar grupos de poblaciones iniciales que consumen memoria computacional, se escoge operar el DPSO luego que se obtenga la primera descendencia, entregando una población de líderes o mejores soluciones en el espacio de búsqueda de cada enjambre, y finalmente se realiza la mutación, la cual a través de pequeños cambios aleatorios busca mejorar las soluciones, el algoritmo opera hasta el cumplimiento del criterio de parada, para este caso se utiliza un número de generaciones definido y calibrado. Cabe resaltar que para el algoritmo GAPSO propuesto se utiliza la optimización por enjambre de partículas discreta (DPSO), debido a la naturalidad discreta del DVCFP, ya que el PSO común es un algoritmo utilizado para optimizar funciones continuas. A continuación, se observa en la figura 17, el diagrama de flujo, el cual, resume cada una de las etapas utilizadas en el algoritmo.

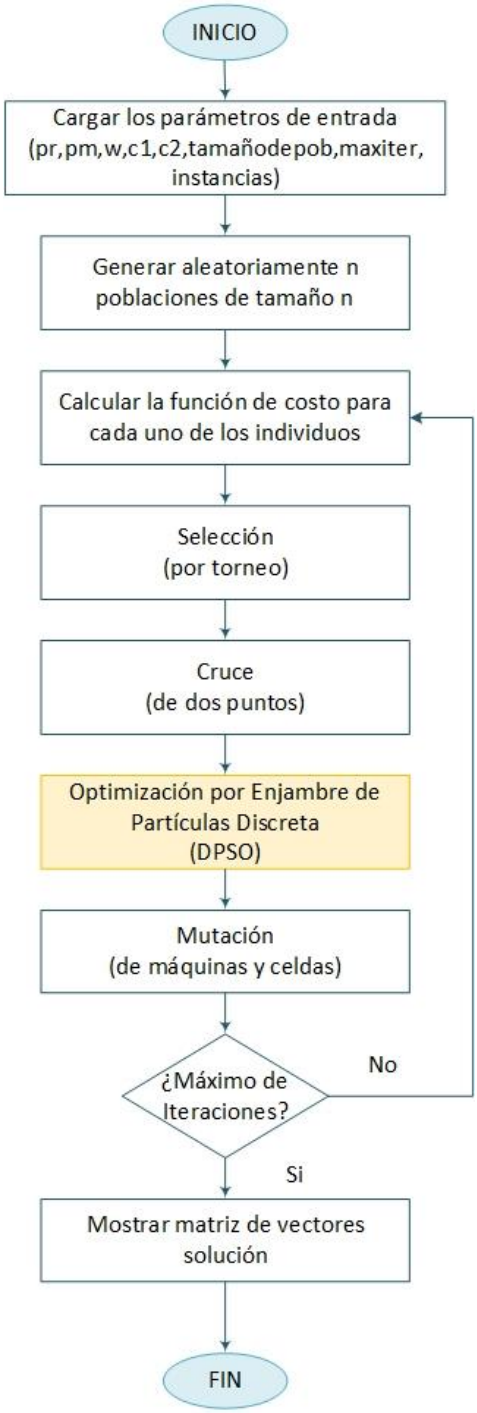


Figura 17. Diagrama de Flujo Algoritmo Híbrido GAPSO.

5.3.5.1. Generación de la población inicial. En el algoritmo genético, el primer paso es la generación de un grupo de soluciones iniciales llamadas "población inicial", el número de soluciones iniciales incluidas en la población es llamado "tamaño de población". Para el caso del

DVCFP, la generación de población se realiza de manera aleatoria bajo los parámetros descritos anteriormente de los vectores solución, en los cuales se evalúa la función objetivo para cada uno de estos resultados.

Debido a que el Algoritmo Genético entrega como resultado un grupo de vectores y la Optimización por enjambre de Partículas genera un solo vector (partícula líder), se propone la generación de diferentes grupos de poblaciones iniciales del mismo tamaño n, de tal manera que el DPSO logre operar conjuntamente con el AG sin perder información genética, es decir, si se escoge un tamaño de población de 100 individuos, se generarán 100 poblaciones iniciales de tamaño 100 (100x100), en este caso el algoritmo genético puede trabajar conjuntamente manejando poblaciones de vectores hasta el DPSO, donde se entrega una matriz de 100 vectores líderes de cada grupo, manteniendo la compatibilidad con las etapas faltantes del AG.

5.3.5.2. Selección. El proceso de selección sirve para escoger a los individuos de la población mejor adaptados, para que actúen de progenitores de la siguiente generación. Para el DVCFP se utiliza la selección por torneo, la cual consiste en realizar la selección con base en comparaciones directas entre individuos. Primero se selecciona al azar un número k de individuos, a estos se les evalúa su función fitness (función de costo), se selecciona al mejor (en este caso al de menor función de costo) y se inserta en una matriz en la cual tendrá más probabilidades de reproducirse. El proceso se realiza hasta que el número de descendientes sea igual al tamaño de la población.

Adicionalmente, la selección se realiza para cada uno de los n grupos de vectores generados, como se observa en la Figura 18 para un ejemplo pequeño de tamaño de población igual a 4, por lo tanto, se generan 4 poblaciones (n) de 4 individuos cada una.

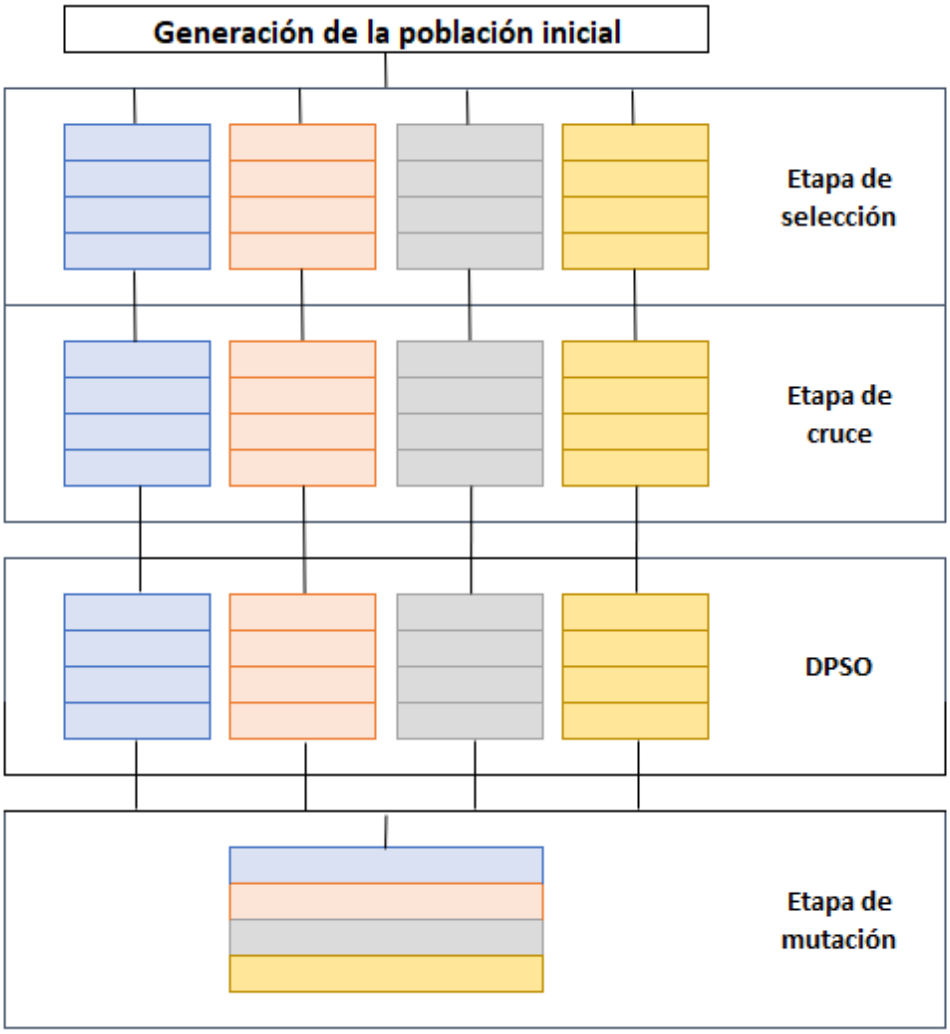


Figura 18. Manejo de las poblaciones del algoritmo híbrido.

5.3.5.3. Operadores de Cruce. Por cada par de padres escogidos al azar, una pequeña proporción de genes seleccionados aleatoriamente se intercambia para producir descendientes con buena información genética, para el presente trabajo se utiliza el cruce de dos puntos, debido a que en la revisión de literatura realizada, se encuentra que es el mayor operador utilizado para los problemas del DVCFP y genera resultados considerablemente buenos; además, con esta técnica se consigue que el espacio de búsqueda del problema sea explorado más a fondo, reduciendo sesgos posicionales. Este cruce consiste en tomar una parte del vector de uno de los padres, comprendida entre dos límites, a y b, para reemplazarlos en la misma posición en el segundo padre, el proceso

se puede observar para un pequeño ejemplo (vector de 3 productos, 5 máquinas y 2 celdas) en la figura 19. Cabe resaltar que este proceso se realiza n veces para las n poblaciones generadas.

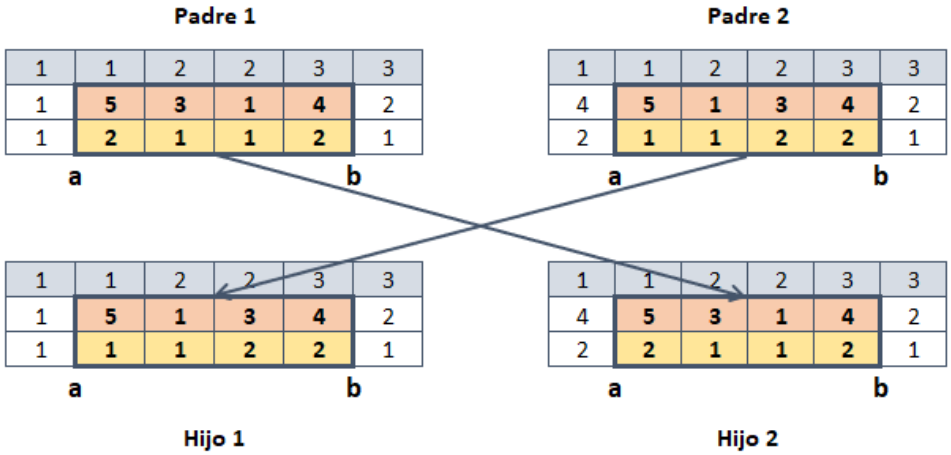


Figura 19. Ejemplo del operador de cruce del algoritmo genético.

5.3.5.4. Etapa de la Optimización por Enjambre de Partículas Discreta (DPSO). En el desarrollo del algoritmo, el DPSO recibe las n poblaciones de n individuos para trabajarlas cada una por separado, entregando las mejores soluciones o líderes de cada grupo en una matriz de tamaño n (como se observa en la figura 18). El PSO, se ha utilizado desde sus inicios para optimizar funciones continuas no lineales, pero en vista de que la mayoría de problemas industriales son discontinuos en el espacio de búsqueda, Rameshkumar, Suresh, & Mohanasundaram (2005) proponen una variante discreta de este algoritmo, el DPSO, para problemas de programación de operaciones utilizando vectores. Debido a que el problema tratado en este trabajo es de carácter discreto, se utiliza la adaptación propuesta por Huang, Guan, & Yang (2018) del PSO.

Con el fin de aprovechar las ventajas de este algoritmo, se redefinen las ecuaciones de posición y de velocidad como se observa a continuación:

$$X_i^{t+1} = c_2 \otimes f_3 \_(c_1 \otimes f_2 \ (w \otimes f_1 (X_i^t), pbest_i^t), gbest_i^t) \quad (1)$$

Donde X_i^{t+1} es la posición de la partícula en la generación t, w, c_1, c_2, representan las constantes de inercia, aprendizaje propio y aprendizaje social respectivamente, $pbest_i^t$ y $gbest_i^t$ indican la mejor posición de la partícula i en la generación t y la mejor posición de la partícula en el enjambre en la generación t respectivamente.

Esta ecuación de posición se descompone en 3 fórmulas subsecuentes de acuerdo con las ecuaciones utilizadas en el PSO.

- **Componente de velocidad inicial.**

En el algoritmo DPSO, cada individuo o cromosoma, es considerado como una partícula sin volumen; a cada una de estas partículas se le asocia una velocidad en el espacio de búsqueda multidimensional; representando la probabilidad de que la posición de la partícula cambie.

$$A = w \otimes f_1 (X_i^t) = \begin{cases} f_1 (X_i^t), & r < w \\ X_i^t, & \text{de lo contrario} \end{cases} \quad (2)$$

$f_1 (X_i^t)$ denota el operador de mutación y r es un número aleatorio uniforme entre (0,1) que asegura que se mantiene la inercia entre las partículas (evitando convergencia prematura y que las partículas no se dispersan de manera abrupta) a pesar de los cambios. Este componente indica que la posición de la partícula es influenciada por su posición anterior y el operador de mutación se utiliza para implementar una búsqueda aleatoria, ayudar a las partículas a evitar óptimos locales y diversificar la población. Este operador trabaja al igual que la etapa de mutación para el desarrollo del algoritmo genético del híbrido.

En la Figura 20 se muestra un ejemplo (3 productos, 5 máquinas y 2 celdas) de mutación de la partícula. Para el caso en que $\boldsymbol{r < w}$, se escoge una posición aleatoria en la segunda fila (máquinas), el algoritmo enlaza la matriz de partes-operaciones-máquinas para encontrar la máquina factible (estrictamente diferente a la que ya se encuentra en el vector), la cual será reemplazada en la partícula. De lo contrario, $\boldsymbol{r > w}$, no se aplica ningún operador a la partícula y continúa con los siguientes componentes.

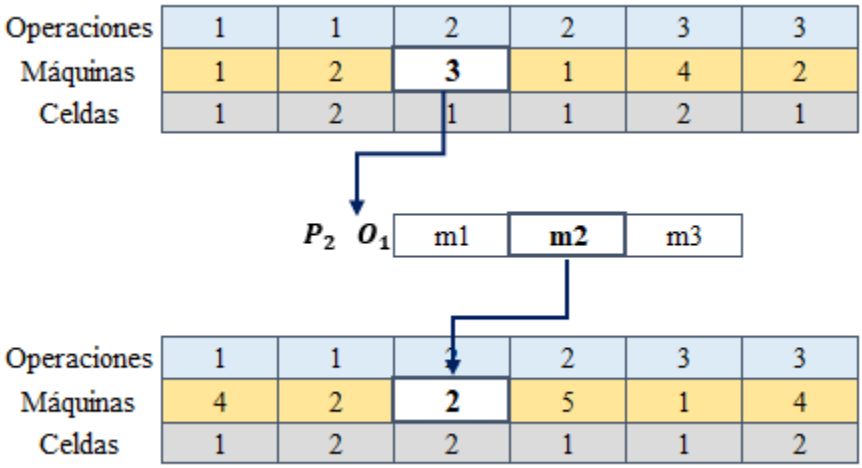


Figura 20. Ejemplo de operador de mutación utilizado en el DPSO.

- **Componente de aprendizaje personal.**

Cada partícula cuenta con una "función de memoria", y ajusta su trayectoria de acuerdo con dos tipos de componentes, la mejor posición o valor en el que se ha encontrado su función fitness y la mejor posición que se ha encontrado en todo el enjambre en cada generación. El componente de aprendizaje personal puede definirse como el resultado de las posiciones en los que la partícula ha estado a través del tiempo, y la mejor posición o valor se almacena como pbest, el cual cambia a medida que la partícula encuentra mejores resultados.

$$B= c_1 \otimes f_2(A, pbest_i^t) = \begin{cases} f_2(A, pbest_i^t), & r < c_1 \\ A, & \text{de lo contrario} \end{cases} \quad (3)$$

En el cual $f_2(A, pbest_i^t)$ indica el operador de cruce entre la partícula y el pbest o mejor personal, bajo la condición de que se produce el cruce sólo si el numero aleatorio r es menor a la constante de aceleración cognitiva (ver Tabla 6), determinando el balance entre la influencia del conocimiento del individuo en su cambio de posición (o cambio en el vector), es decir, la ecuación describe que la partícula puede ajustar su posición (valor) de acuerdo a las mejores posiciones (valores) de sí misma encontrados a través de las generaciones en su memoria. El operador de cruce perteneciente a esta etapa es el cruce de dos puntos con el cual se realiza el módulo de cruce del Algoritmo Genético del híbrido. En la figura 21, se presenta un ejemplo de cruce (3 productos, 5 máquinas y 2 celdas) cuando $r < c_1$ entre la partícula y el pbest asociado a ella, en el caso en que $r > c_1$, no se realiza ningún cambio en la partícula, dejando su memoria personal intacta.

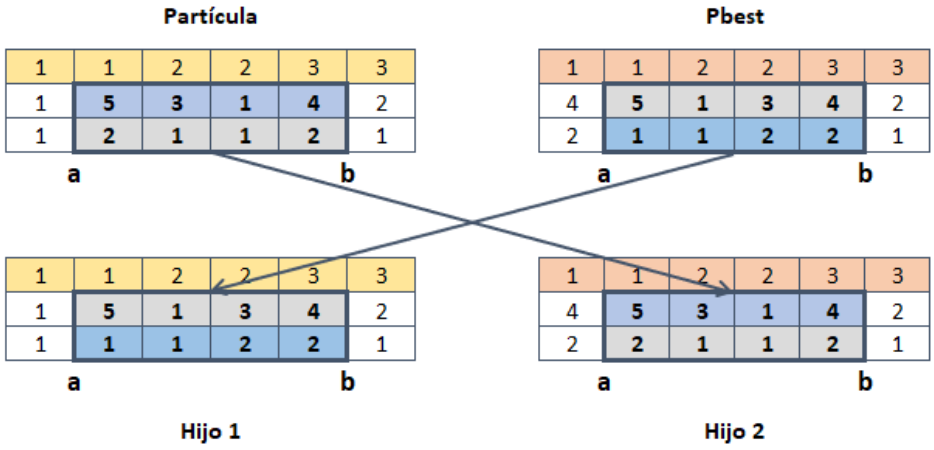


Figura 21. Ejemplo de cruce de aprendizaje personal de la partícula en el DPSO.

- **Componente de aprendizaje social.**

El movimiento de cada partícula depende de su mejor posición obtenida (aprendizaje personal), así como de la mejor posición global hallada de todas las partículas en el espacio de búsqueda, esto se conoce como aprendizaje social, la mejor posición encontrada en todo el enjambre se conoce como gbest. A medida que se descubren nuevas y mejores posiciones, éstas comienzan a orientar los movimientos de las partículas influenciadas por sí misma y por todo el grupo.

$$C = c_2 \otimes f_3\ (B, gbest_i^t) = \begin{cases} f_3\ (B, gbest_i^t)\ , & r < c_2 \\ B\ , & \text{de lo contrario} \end{cases} \quad (4)$$

$f_3\ (B, gbest_i^t)$ denota el operador de cruce entre la partícula y el gbest (mejor grupal). La ecuación 4 describe que las partículas ajustan su posición según la mejor partícula del grupo, lo que significa que las partículas adquieren conocimiento del enjambre bajo la condición de influencia social. Este cruce se realiza de la misma manera que en el componente de aprendizaje personal.

En la figura 22, se muestra un ejemplo de cruce (3 productos, 5 máquinas y 2 celdas) cuando $r < c_2$ entre la partícula y el gbest de la población, en el caso de que $r > c_2$, no se realiza ningún cambio.

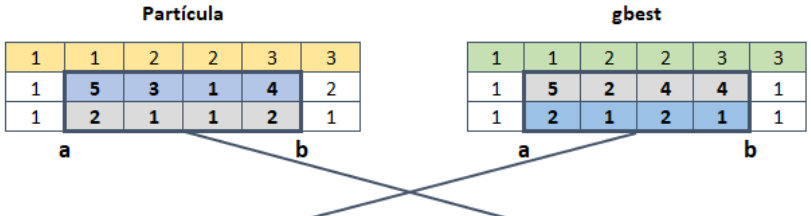


Figura 22. Ejemplo de cruce de aprendizaje social de la partícula en el DPSO.

El DPSO opera hasta que se cumple el criterio de parada, en este caso el número máximo de generaciones con el que trabaja el GAPSO general, parámetro de entrada establecido en el algoritmo. El DPSO entrega una matriz de individuos (líderes de cada uno de los n grupos) como se observa en la figura 18, el cual es parámetro inicial de la función de mutación. En la figura 23, se muestra el diagrama de flujo del DPSO.

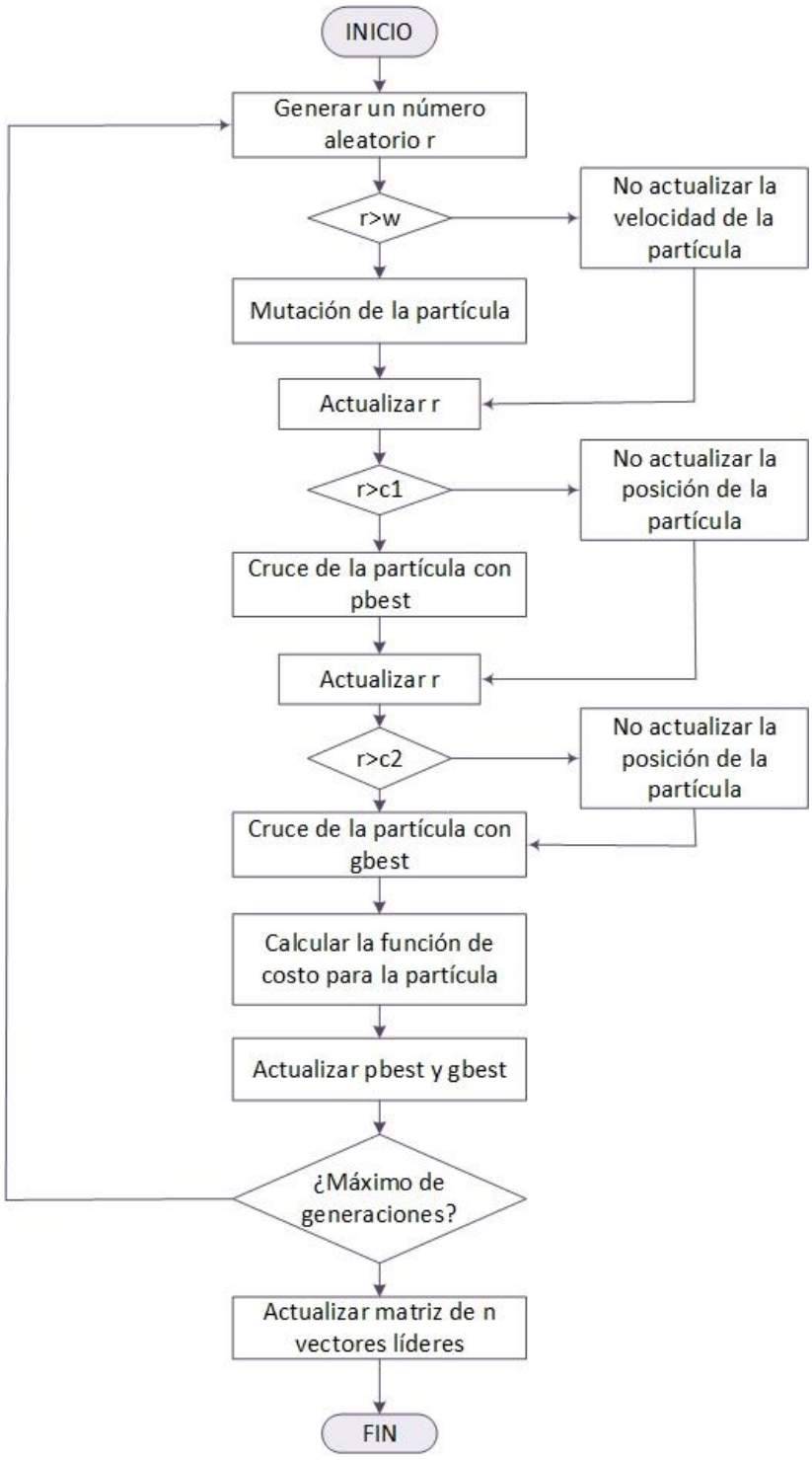


Figura 23. Diagrama de flujo DPSO.

5.3.5.5 Operador de mutación. Las funciones de mutación realizan pequeños cambios aleatorios en los individuos de la población, generando diversidad genética y habilita al algoritmo

realizar la búsqueda en un espacio de soluciones más amplio. En este trabajo se utilizan dos tipos de operadores de mutación: mutación de máquinas y mutación de celdas.

La mutación de máquinas selecciona aleatoriamente una posición de la cuarta fila del vector y remplaza el tipo de máquina seleccionada para la operación con base en la matriz de operaciones-partes-máquinas (como se observa en la figura 24), al igual que la mutación de celdas selecciona aleatoriamente una posición y lo cambia entre los números posibles de cantidad de celdas, teniendo en cuenta que el número no puede ser el mismo que se encontraba antes en esa posición.

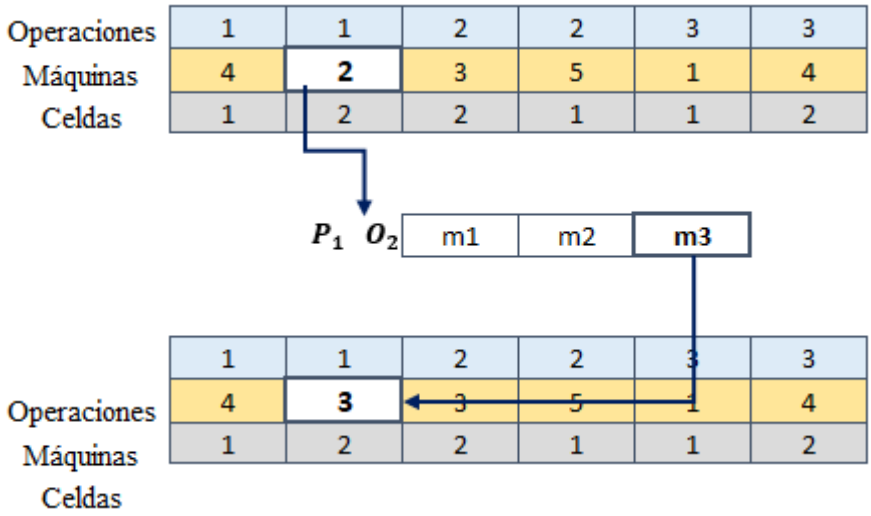


Figura 24. Ejemplo de mutación de máquinas para el algoritmo genético.

El proceso se repite hasta que el algoritmo alcanza o establece su criterio de parada, para el caso del GAPSO aplicado al DVCFP, se escoge un número máximo de generaciones, el cuál es calibrado y definido como parámetro de entrada del algoritmo. Debido a la forma de programación del algoritmo, cuando se cumple este criterio, entrega el vector asociado a la menor función objetivo encontrada para los periodos del horizonte de planeación definido.

6. Calibración del algoritmo

Para la calibración del GAPSO propuesto se realiza un diseño de experimentos tipo 2^k implementado en el software Minitab 2018. El diseño experimental tipo 2^2 se desarrolla con 5 réplicas por cada tamaño de conjunto de instancias. Los factores junto con sus respectivos niveles se muestran en la Tabla 7 Cabe resaltar que estos valores son calibrados debido a que son compartidos por los dos algoritmos y no se cuenta con estos parámetros en la literatura encontrada del DVCFP para un algoritmo genético híbrido con el DPSO.

Tabla 7.
Factores y niveles del diseño de experimentos

Factor	***Nivel alto***	***Nivel bajo***
Tamaño de población	200	50
Máximo de generaciones	100	25

A continuación, se presenta el análisis realizado para cada uno de los diferentes tamaños de instancias:

- ***Instancias nivel bajo:***

El análisis de varianza presentado en la Tabla 8, permite notar que el efecto principal para las instancias nivel bajo es el tamaño de población, con un valor p cercano a 0 (0,026), el cual está acorde a los principios del Algoritmo Genético. El diagrama de Pareto de la figura 24, muestra la incidencia significativa de este factor en cuestión, respecto al número máximo de iteraciones y la interacción entre ellos, además, con base en la Figura 25, se puede concluir que los niveles de factores que minimizan el costo son: tamaño de población 200 y un máximo número de iteraciones de 50, valores con los cuales se corren los paquetes de instancias bajas o pequeñas en el software MATLAB 2018 para analizar sus resultados.

Tabla 8.
Análisis de varianza para las instancias nivel bajo

Fuente	*GL*	*SC Ajust.*	*MC Ajust.*	*Valor F*	*Valor p*
Modelo	3	23914829	7971610	2,51	0,096
Lineal	2	21524660	10762330	3,39	0,059
Tpob	1	19078858	18078858	6,00	0,026
MaxIter	1	2445802	2445802	0,77	0,393
Interacciones de 2 términos	1	2390170	2390170	0,75	0,399
Tpob*MaxIter	1	2390170	2390170	0,75	0,399
Error	16	50854817	3178426		
Total	19	74769646			

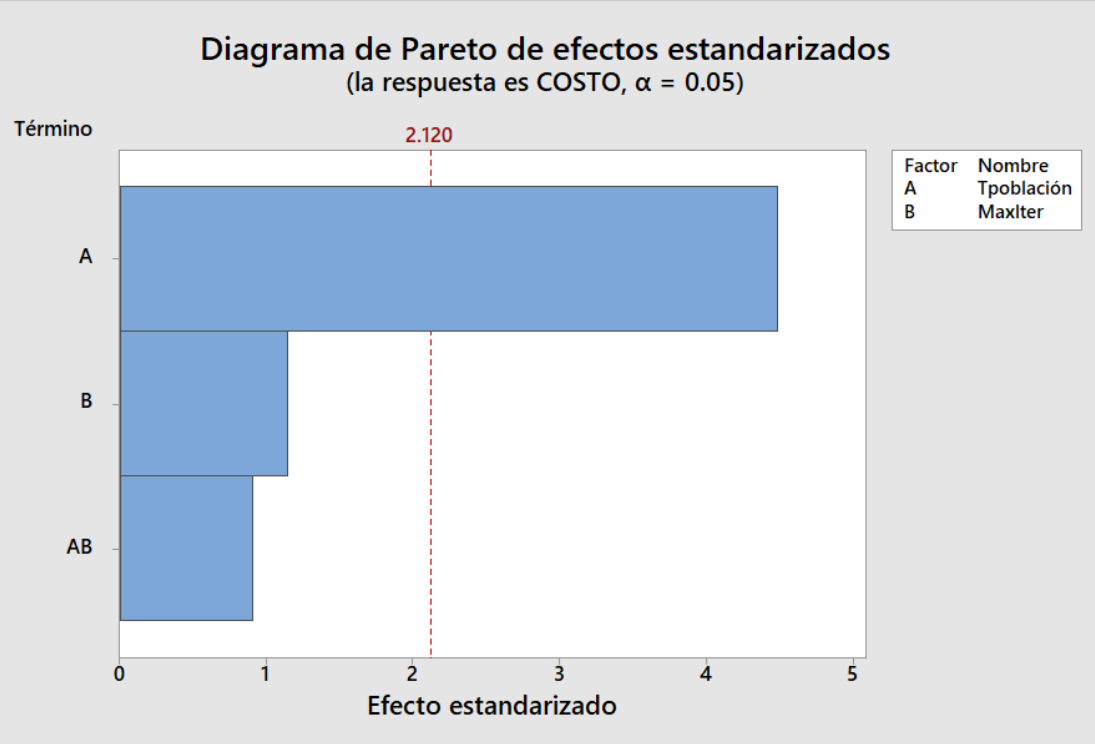


Figura 25. Diagrama de Pareto de efectos estandarizados para las instancias pequeñas. Adaptado de software Minitab 2018.

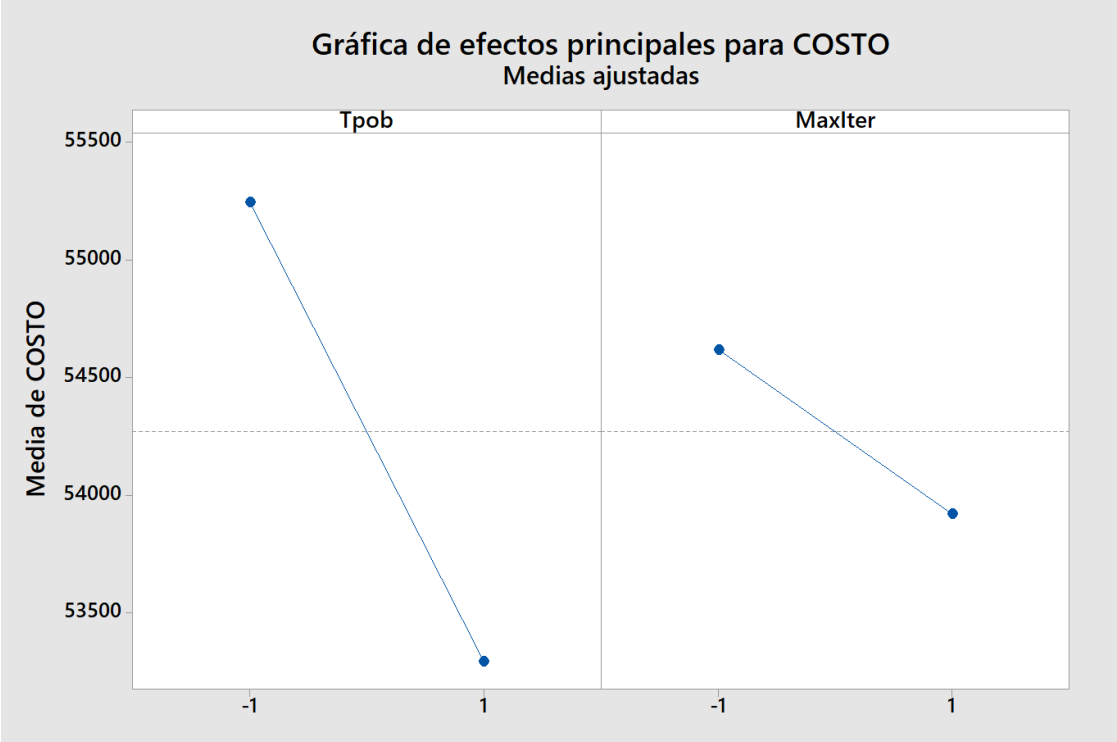


Figura 26. Gráfica de efectos principales para las instancias pequeñas. Adaptado de software Minitab 2018.

- ***Instancias nivel alto:***

Tomando como referencia el análisis de varianza de la Tabla 9, se logra observar que el efecto principal para las instancias grandes es el tamaño de población con un valor de 0,001. La figura 27, reafirma los resultados del análisis de varianza para las instancias en cuestión, mientras que el número máximo de iteraciones no presenta una contribución considerable.

Finalmente, la figura 28 muestra que el nivel del factor tamaño de población que minimiza la función de costo es el nivel alto (200) mientras el número de iteraciones no presenta mayor influencia en la función de costo, por lo tanto, se toma el nivel bajo (25), y el tamaño de población nivel alto (200), como parámetros iniciales para el desarrollo del algoritmo GAPSO, el cual permite dar resultados en menores tiempos computacionales.

Tabla 9.
Análisis de varianza para las instancias nivel alto

Fuente	*GL*	*SC Ajust.*	*MC Ajust.*	*Valor F*	*Valor p*
Modelo	*3*	*71526019*	*23842006*	*6,78*	*0,004*
Lineal	*2*	*59657237*	*29828618*	*8,48*	*0,003*
Tpob	*1*	*59350460*	*59350460*	*16,88*	*0,001*
MaxIter	*1*	*306776*	*306776*	*0,09*	*0,772*
Interacciones de 2 términos	*1*	*11868782*	*11868782*	*3,38*	*0,085*
Tpob*MaxIter	*1*	*11868782*	*11868782*	*3,38*	*0,085*
Error	*16*	*56256790*	*3516049*		
Total	*19*	*127782809*			

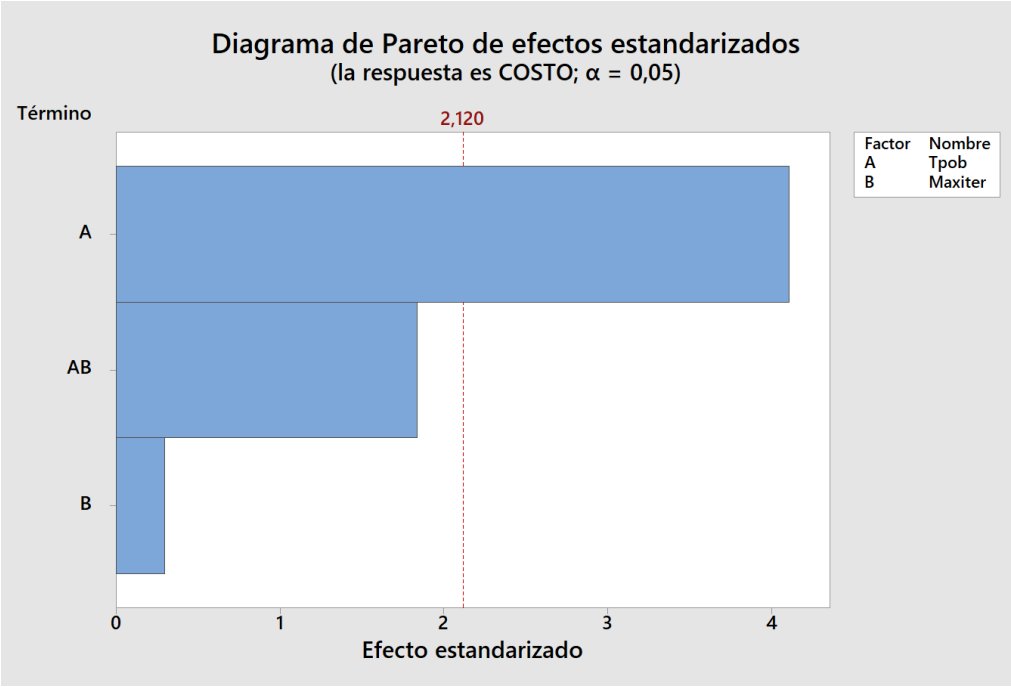


Figura 27. Diagrama de Pareto de efectos estandarizados para las instancias grandes. Adaptado de software Minitab 2018.

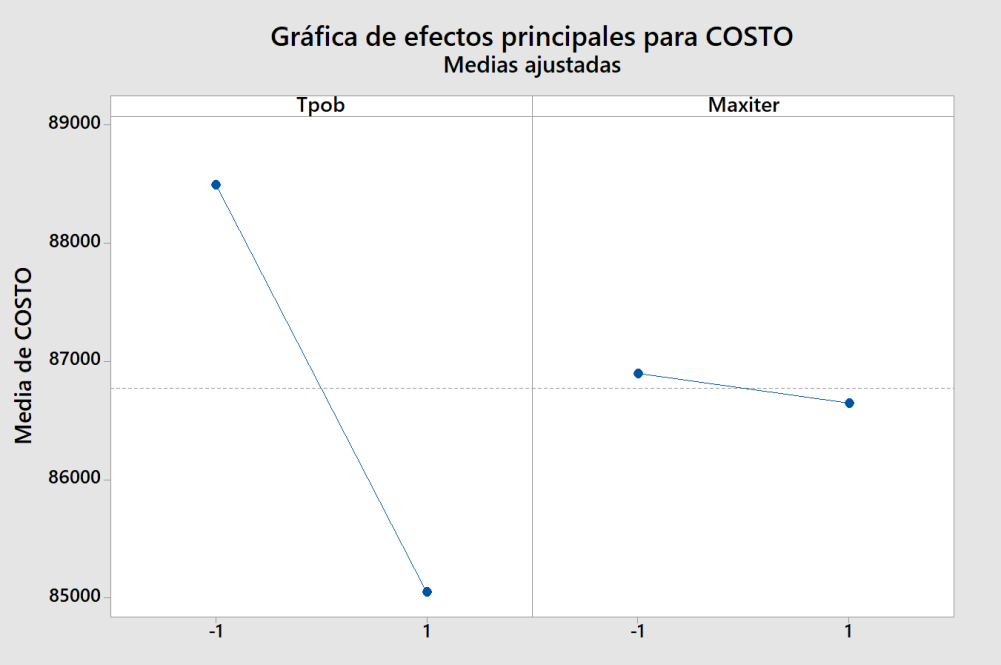


Figura 28. Gráfica de efectos principales para las instancias grandes. Adaptado de software Minitab 2018.

7. Resultados

El modelo propuesto para solucionar el DVCFP se ejecuta en un computador con procesador Intel® Core™ i5-4570 de 3,2 GHz, RAM de 8 GB y sistema operativo Windows 7 Home premium de 64 bits.

7.1 Desarrollo del modelo MILP a través de GAMS

Para el desarrollo del modelo MILP se utiliza el software GAMS/CPLX versión 23.9.5. Debido a la complejidad computacional del problema se estableció el límite de una hora (3.600 segundos) para la solución de cada uno de los paquetes de datos.

- ***Instancias nivel bajo:***

En la Tabla 10 se presentan los resultados en términos de costos y el GAP relativo relacionado, para los 5 paquetes de instancias nivel bajo se presentan soluciones factibles.

El algoritmo Branch & Cut en GAMS, mantiene dos números relevantes: "mejor estimación" y "mejor número entero". El "mejor entero" es la mejor solución que satisface todas las restricciones encontrada hasta ahora. La "mejor estimación" proporciona un límite para la solución entera óptima. La calidad de una solución se puede medir como la distancia de la solución óptima, a esta distancia en términos porcentuales se le conoce como GAP relativo, cuando el GAP relativo es cero (como fue establecido en el código) el algoritmo termina.

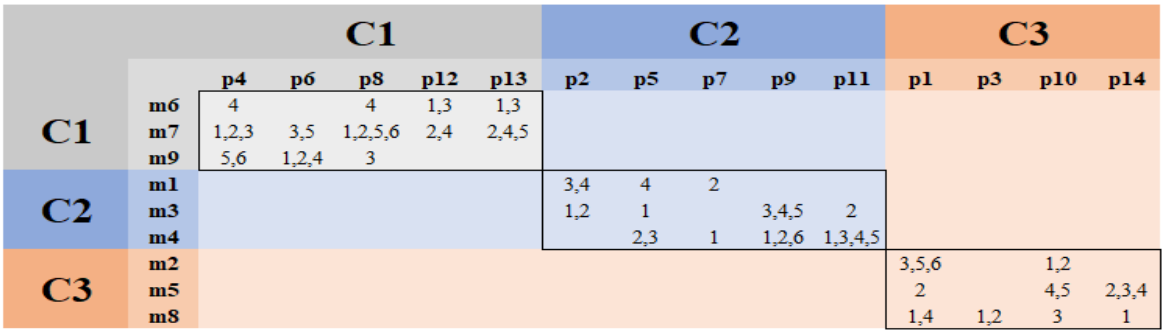
Tabla 10.
Resultados del modelo MILP para las instancias nivel bajo

Datos	Costo de operación	Costo interno de celda	Costo de transporte entre celdas	Costo de transporte entre máquinas	Costo total	GAP Relativo (%)
1	$ 26.835	$ 13.500	$ 6.600	$ 60.100	$107.035	14,380
2	$ 18.508	$ 13.500	$ 0	$ 43.200	$ 75.208	3,1867
3	$ 25.420	$ 13.500	$ 15.300	$ 62.900	$117.120	14,275
4	$19.902	$13.500	$54.900	$36.600	$124.902	39,373
5	$ 28.639	$ 13.500	$ 13.200	$59.100	$114.439	13,294

En la figura 29, figura 30 y figura 31, se presenta un ejemplo de la asignación de operaciones, productos y máquinas para cada una de las celdas en los diferentes periodos establecidos, tomados de la instancia pequeña 1.

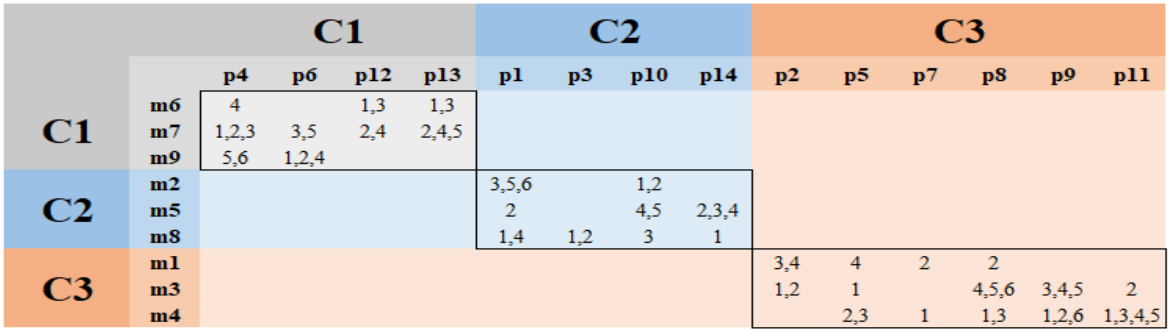
		C1				C2								C3				
		p1	p4	p8	p13	p1	p2	p5	p6	p8	p9	p11	p12	p3	p6	p7	p10	p14
C1	m1	3	6															
	m6	2,4,6	4		1,3													
	m7	5	1,2,3,5	1,2	2,4,5													
C2	m2						3		3			4,5	2,3					
	m3						1,2,4	1,4		4,5,6	3,4,5	2,3						
	m4					1		2,3	4,5	3	1,2,6	1	1,4					
C3	m5																4,5	2,3,4
	m8													1,2		1	3	
	m9														1,2	2	1,2	1

Figura 29. Representación de CM, conjunto 1 de instancias bajas, periodo h_1 en GAMS.



		C1					C2					C3			
		p4	p6	p8	p12	p13	p2	p5	p7	p9	p11	p1	p3	p10	p14
C1	m6	4		4	1,3	1,3									
	m7	1,2,3	3,5	1,2,5,6	2,4	2,4,5									
	m9	5,6	1,2,4	3											
C2	m1						3,4	4	2						
	m3						1,2	1		3,4,5	2				
	m4							2,3	1	1,2,6	1,3,4,5				
C3	m2											3,5,6		1,2	
	m5											2		4,5	2,3,4
	m8											1,4	1,2	3	1

Figura 30. Representación de CM, conjunto 1 de instancias bajas, periodo h_2 en GAMS.



		C1				C2				C3					
		p4	p6	p12	p13	p1	p3	p10	p14	p2	p5	p7	p8	p9	p11
C1	m6	4		1,3	1,3										
	m7	1,2,3	3,5	2,4	2,4,5										
	m9	5,6	1,2,4												
C2	m2					3,5,6		1,2							
	m5					2		4,5	2,3,4						
	m8					1,4	1,2	3	1						
C3	m1									3,4	4	2	2		
	m3									1,2	1		4,5,6	3,4,5	2
	m4										2,3	1	1,3	1,2,6	1,3,4,5

Figura 31. Representación de CM, conjunto 1 de instancias bajas periodo h_3 en GAMS.

- ***Instancias nivel alto:***

Para las instancias constituidas por 5 paquetes de datos conformados por 20 partes y 13 máquinas, se encuentra una solución factible para cada uno de estos. Los resultados se presentan en la Tabla 11.

Tabla 11.
Resultados del modelo MIP para las instancias nivel alto.

Datos	Costo de operación	Costo interno de celda	Costo de transporte entre celdas	Costo de transporte entre máquinas	Costo total	GAP Relativo (%)
1	$ 26.835	$ 13.500	$ 6.600	$ 60.100	$107.035	14,380
2	$ 18.508	$ 13.500	$ 0	$ 43.200	$ 75.208	3,1867
3	$ 25.420	$ 13.500	$ 15.300	$ 62.900	$117.120	14,275
4	$19.902	$13.500	$54.900	$36.600	$124.902	39,373
5	$ 28.639	$ 13.500	$ 13.200	$59.100	$114.439	13,294

A continuación, se presenta un ejemplo gráfico de la asignación de operaciones de cada uno de los productos, a las máquinas en cada una de las celdas para cada uno de los 3 periodos establecidos, como se puede observar en la figura 32, figura 33 y figura 34.

		C1														C2			C3			
		p1	p2	p5	p8	p9	p10	p12	p13	p14	p15	p16	p17	p18	p19	p7	p10	p20	p3	p4	p6	p11
C1	m3	1			5	1	3		1,2	1		1		5	5,6							
	m4		3,4	2	2			3		2	3	2	1	4								
	m6	2		1		3		4	5				3	2	2,3							
	m7				1,3			5	4		4	3	2	3,6	4							
	m8						4	1,2	3		1,2,6				1							
	m9		1,2		4	2					5		4	1								
C2	m2															1,2		1,3,5				
	m5																	4				
	m10															3	2	2, 6				
	m11															4	1					
C3	m1																		3,5,6	2,3	3	4
	m12																		1			3,5
	m13																		2,4	1	1,2	1,2

Figura 32. Representación de CM, conjunto 1 de instancias grandes, periodo h_1 en GAMS.

		C1			C2															C3		
		p10	p11	p19	p2	p3	p4	p5	p7	p8	p9	p11	p12	p13	p14	p15	p16	p17	p20	p1	p6	p18
C1	m3	3		3,5,6																		
	m7	1		1,2,4																		
	m10	2,4	3																			
	m13		1,2																			
C2	m1					5,6	2,3			2		4	4		2			2	6			
	m2								1,2	4,5		5				3,4						
	m4				3,4	2,4		2					3,5	2,4			2	1	1			
	m6					3		1		3	1,3			5	1		1	3				
	m8					1			4	1			1,2	1,3		1,2,6			2,3			
	m9				1,2		1		3		2					5	3	4	4,5			
C3	m5																			1	1	6
	m11																				3	2,3
	m12																			2	2	1,4,5

Figura 33. Representación de CM, conjunto 1 de instancias grandes, periodo h_2 en GAMS.

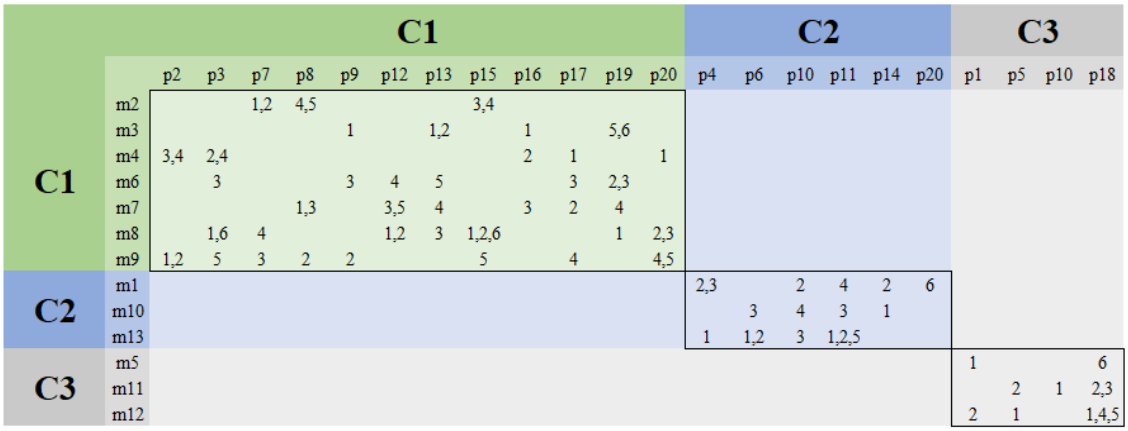


Figura 34. Representación de CM, conjunto 1 de instancias grandes, periodo h_3 en GAMS.

7.2 Implementación del Algoritmo Genético Híbrido (GAPSO) en MATLAB.

La metaheurística propuesta se desarrolla en el software Matlab R2018b, mediante la adaptación de las instancias utilizadas para la implementación del modelo MILP.

Cabe resaltar que para el correcto funcionamiento del algoritmo es necesario adaptar n número de poblaciones de tamaño n con el fin de buscar compatibilidad entre el Algoritmo Genético y la Optimización por Enjambre de Partículas (ver Figura 18), lo cual incide en el tiempo computacional requerido para dar solución al problema. Los resultados son divididos entre los tamaños de instancias como se muestra a continuación:

7.2.1 Instancias nivel bajo. Para los 5 conjuntos de instancias pequeñas o nivel bajo se encuentran soluciones factibles dentro los requerimientos del problema abordado, para estas instancias se utilizó el tamaño de población 200 y se realizaron 50 iteraciones (resultados de la calibración del algoritmo), los costos totales y tiempo computacional se muestran en la

Tabla 12. En la figura 35 se observa un ejemplo de vector solución para los 3 periodos de la instancia 1 del conjunto.

Tabla 12.
Resultados de GAPSO para las instancias nivel bajo

Instancia	Costo total	Tiempo computacional [s]
1	$ 54.930	3.936,799
2	$ 57.785	3.770,684
3	$ 51. 514	3.974,250
4	$ 61.412	3.754,200
5	$ 69.098	4.069,181

En la figura 36, figura 37 y figura 38, se muestran la distribución de las celdas para cada uno de los periodos de planeación.

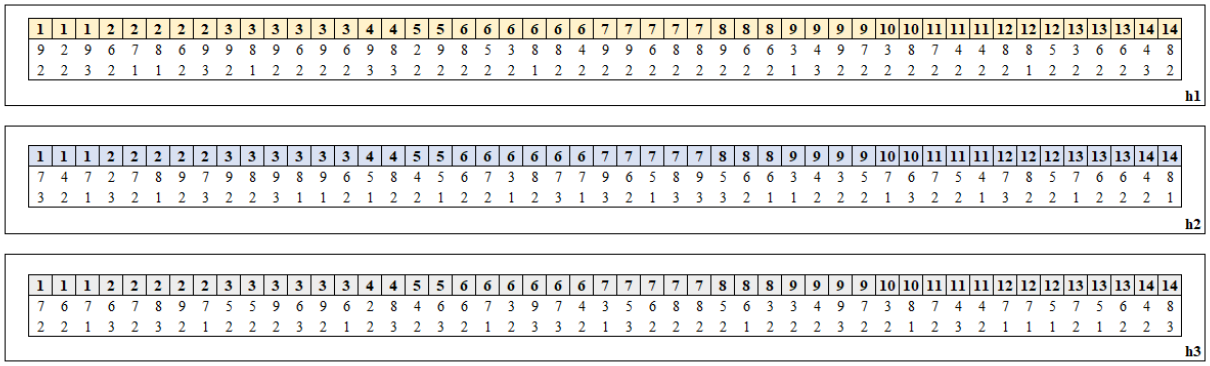


Figura 35. Vector solución proporcionado por GAPSO para instancias 1 de niveles bajo.

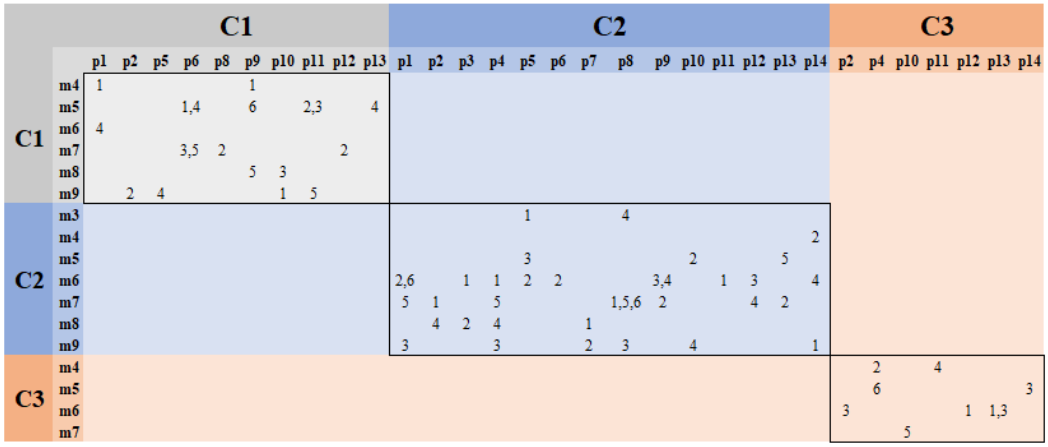


Figura 36. Distribución de VCM, instancia baja 1, periodo h_1, desarrollada por GAPSO.

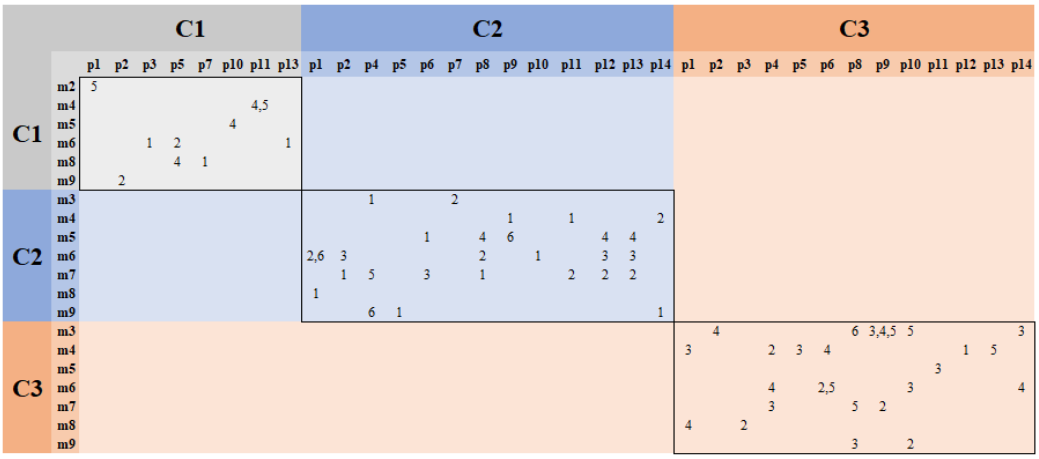


Figura 37. Distribución de VCM, instancia baja 1, periodo h_2, desarrollada por GAPSO.

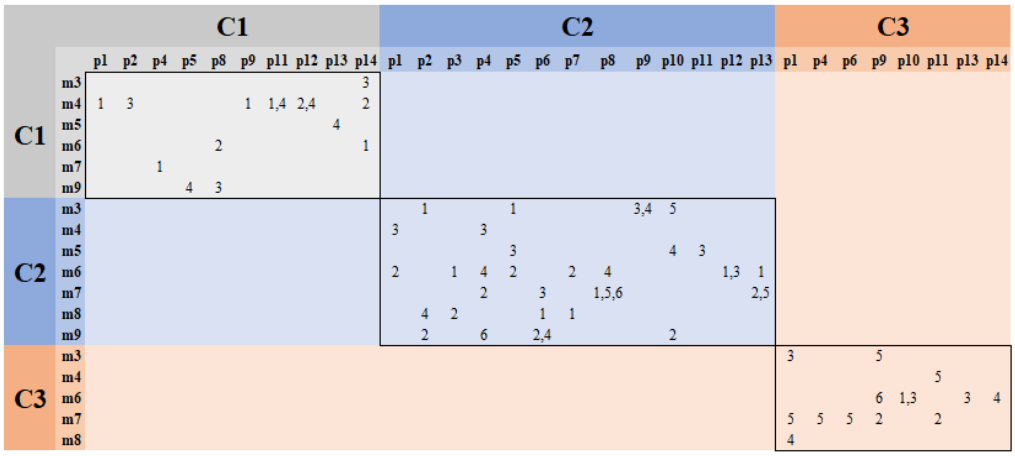


Figura 38. Distribución de VCM, instancia baja 1, periodo h_3, desarrollada por GAPSO.

7.2.2 Instancias nivel alto. De los 5 conjuntos de instancias considerados por su tamaño como nivel alto, todos presentan soluciones factibles, en la Tabla 13 se muestran los resultados obtenidos. En la Figura 39, Figura 40 y figura 41 se presenta la asignación de las operaciones de cada producto a las máquinas en cada una de las celdas para las instancias 1 de manera gráfica.

Tabla 13.
Resultados de GAPSO para las instancias nivel alto.

Instancia	Costo total	Tiempo computacional [s]
1	$87.492	3.635,888
2	$88.672	2.484,444

Continuación Tabla 13. *Resultados de GAPSO para las instancias nivel alto.*

3	**$95.632**	**2.400,353**
4	$88.798	2.583,439
5	$101.630	2.588,319

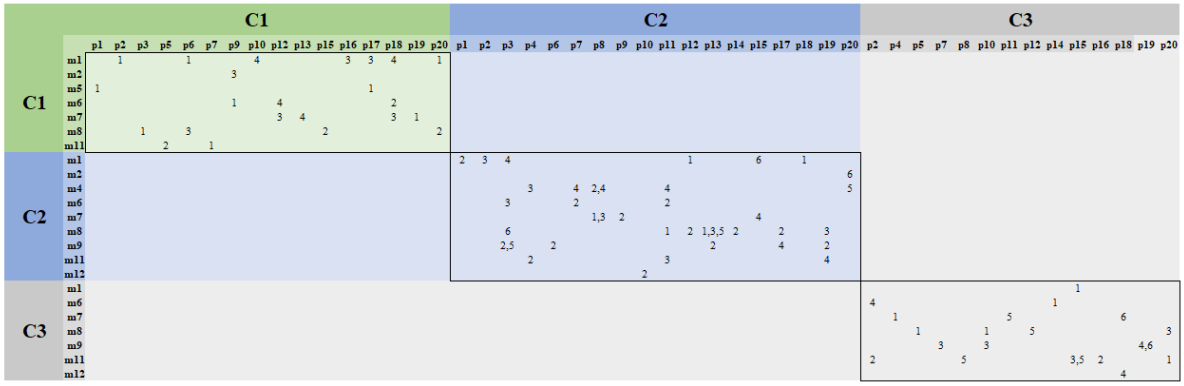


Figura 39. Distribución de VCM, instancia grande 1, periodo h_1, desarrollada por GAPSO.

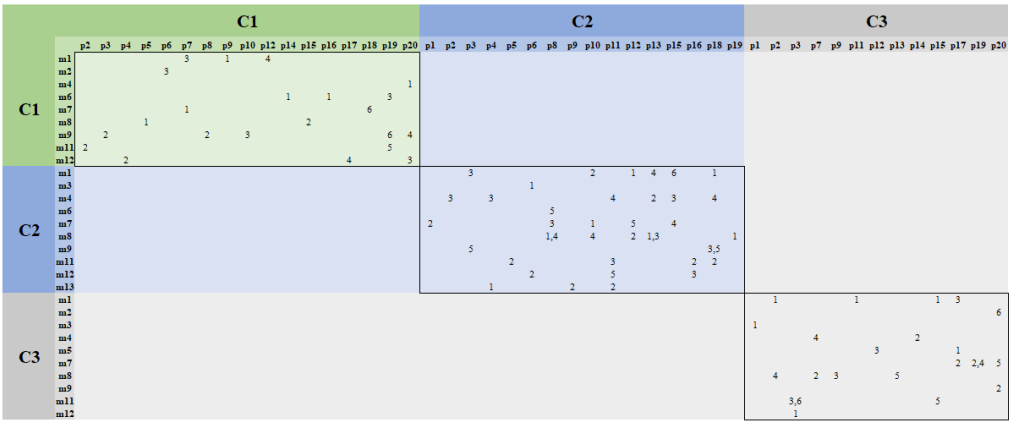


Figura 40. Distribución de VCM, instancia grande 1, periodo h_2, desarrollada por GAPSO.

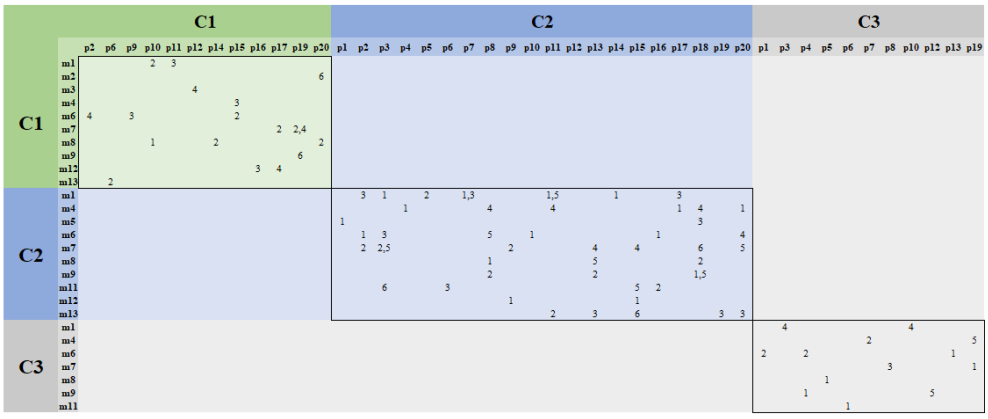


Figura 41. Distribución de VCM, instancia grande 1, periodo h_3, desarrollada por GAPSO.

La Tabla 14 presenta los resultados para los niveles alto y bajo de cada uno de los métodos de solución, el GAMS que representa el modelo exacto y el GAPSO que representa la metaheurística híbrida programada.

Para las instancias nivel bajo, se logra observar que en temas de costos el algoritmo GAPSO mejora la solución en porcentajes desde el 65% al 78%, en comparación con el modelo exacto obtenido en GAMS, satisfaciendo la misma demanda con costos más bajos. En cuanto al tiempo computacional se muestra que el algoritmo duró más tiempo que los 3.600 segundos que se tomaron de referencia para el modelo exacto (los valores se colocan negativos para mostrar que el tiempo en GAMS fue menor), aun así, en el momento en que transcurría una hora, el modelo exacto aún presentaba valores de GAP relativo en porcentajes desde 3% hasta 39%, indicando que aún faltaba tiempo para exploración.

En cuanto las instancias nivel alto, el algoritmo GAPSO también presenta mejores resultados en términos de costos, con diferencias porcentuales desde el 37%. Cabe resaltar que, para el tiempo computacional, el algoritmo logró disminuirlo en valores desde 28% hasta 31%, excepto por la instancia 1, para la cual la diferencia de tiempos fue menor solo por el 1%, aun así, es importante aclarar que el modelo exacto fue interrumpido a los 3.600 segundos, y presentó un GAP relativo de 23% para ese tiempo estipulado.

Tabla 14.
Comparación de resultados bajo los dos métodos realizados

Instancias Nivel Bajo				
Instancia	GAMS	GAPSO	% diferencia (costos)	% diferencia (tiempo computacional)
1	$ 201.242	$ 54.930	73%	-9%
2	$ 260.340	$ 57.785	78%	-5%
3	$ 147.946	$ 51. 514	65%	-10%
4	$ 221.324	$ 61.412	72%	-4%

Continuación Tabla 14. *Comparación de resultados bajo los dos métodos realizados*

5	**$ 291.637**	**$ 69.098**	**76%**	**-13%**
Instancias Nivel Alto				
1	*$ 163.208*	*$87.492*	*46%*	*-1%*
2	*$ 180.556*	*$88.672*	*51%*	*31%*
3	*$ 151.932*	*$95.632*	*37%*	*33%*
4	*$ 140.380*	*$88.798*	*37%*	*28%*
5	*$ 175.709*	*$101.630*	*42%*	*28%*

8. Conclusiones

- A partir de la revisión de literatura realizada, se logra identificar que el DVCFP es una estrategia de distribución de planta con alto interés por parte de la comunidad científica, y que, además, es un tema que ha tomado importancia a lo largo del tiempo, gracias a sus ventajas en cuanto a costos de producción y reconfiguración.
- En el modelo MILP implementado en GAMS, se tienen en cuenta diferentes componentes de costos, asociados a la transferencia de materiales entre celdas y entre máquinas, a diferencia de Lv, Yuan, & Han (2013), quienes tienen en cuenta únicamente el primero. Para los 10 conjuntos de instancias clasificados en 2 grupos, bajas y altas, debido a la cantidad de máquinas y celdas, se generan soluciones factibles; sin embargo, cabe destacar que por el alto tiempo computacional es necesario interrumpir su implementación pasados 60 minutos, aun así, al interrumpir el proceso, se mantienen valores de GAP relativos que van desde el 3% hasta el 39% para los datos en instancias bajas y de 9% a 23% para las instancias nivel alto.
- Mediante el diseño de experimentos con el cual se calibran los parámetros de entrada compartidos por los dos algoritmos, se logra evidenciar que el tamaño de población es un factor clave en la obtención de buenas soluciones en materia de costos para el algoritmo. Por lo tanto, se establece el factor tamaño de población en su nivel alto (200) para ambos conjuntos de instancias, mientras que para las instancias nivel alto se toma el factor número de iteraciones en su nivel bajo, ya que, mediante el diseño de experimentos del conjunto, se logra observar que no presenta una incidencia significativa en el valor de los

costos, pero si presenta menores valores en cuestión del tiempo computacional del algoritmo para estas instancias.

- Para la solución del DVCFP a través de la metaheurística, se utiliza el Algoritmo Genético junto con la Optimización por Enjambre de partículas discreta, mediante un híbrido (GAPSO), el DPSO opera luego de la etapa de cruce del AG, debido a que fue necesario introducir n poblaciones de tamaño n (en el caso de las instancias evaluadas, 200 poblaciones de tamaño 200) para la operación conjunta de ambos algoritmos, gastando tiempo computacional por la memoria del computador, luego el DPSO entregaba una sola población con 200 vectores, permitiendo la continuación de las demás etapas del AG.
- Mediante la implementación de las instancias y la comparación de los resultados, fue posible observar que para ambos conjuntos definidos de datos el algoritmo mejora los resultados obtenidos por el GAMS en cuestión de costos y tiempo computacional. Para las instancias clasificadas como nivel bajo, el algoritmo encuentra resultados factibles mejorando la solución desde un 65% a 78% para todos los datos, en cuanto a tiempos computacionales, a pesar de que el algoritmo no termina antes de los 3.600 segundos en los cuales GAMS trabajaba, la diferencia porcentual de tiempo no supera el 13%. Para las instancias nivel alto, el GAPSO mejora las soluciones en cuanto a costos de un 37% a un 46% y en cuanto a tiempos computacionales también son menores que los utilizados por GAMS (de 28 a 31% de mejora). Por lo tanto, se puede concluir que el algoritmo GAPSO mejora los resultados del modelo exacto para las instancias evaluadas tanto en temas de costos como en tema de tiempo computacional.

9. Recomendaciones

Para futuros trabajos se recomienda:

- Integrar otro tipo de consideraciones al DVCFP presentado, tales como programación de las tareas, parámetros difusos, consideración de averías, manejo de inventario tanto terminado como en proceso, trabajadores, limitación de la cantidad de máquinas asociada a cada operación, entre otros; que permitirán acercar cada vez más el modelo a la industria real.
- Crear un modelo de celdas de manufactura dinámicas convencionales de tal manera que sea posible realizar una comparación entre las celdas virtuales, con el fin de observar cual estrategia es más conveniente para el proceso de producción.
- Realizar pruebas con instancias de tamaños más grandes, ejecutando la codificación binaria del problema, de tal manera que, a medida que aumenten las columnas del vector (cantidad de operaciones del periodo) no se presenten costos elevados debido a que no sea posible evadir las penalizaciones o restricciones del problema. Otra posible propuesta, sería realizar una programación que permita rastrear operaciones y corregir las infactibilidades que se presentan a medida que el problema avanza en tamaño.
- Se recomienda realizar pruebas de comparación de tal manera que se logre evidenciar en qué etapa de hibridación del algoritmo genético, el DPSO produce mejores resultados, ya sea en la generación de la población inicial, o después de la etapa de mutación.
- Abordar el DVCFP mediante otros enfoques de solución, con el fin de comparar el rendimiento de diversas técnicas para este problema.

Referencias Bibliográficas

Aalaei, A., & Davoudpour, H. (2016). Revised multi-choice goal programming for incorporated dynamic virtual cellular manufacturing into supply chain management: A case study. *Engineering Applications of Artificial Intelligence*, *47*, 3–15. https://doi.org/10.1016/j.engappai.2015.04.005

Ahkioon, S., Bulgak, A. A., & Bektas, T. (2009). Cellular manufacturing systems design with routing flexibility, machine procurement, production planning and dynamic system reconfiguration. *International Journal of Production Research*, *47*(6), 1573–1600. https://doi.org/10.1080/00207540701581809

Baquela, E. G., & Redchuck, A. (2013). *Optimización Matemática con R Vol. I*. 153.

Baykasoglu, A., & Gorkemli, L. (2017). Dynamic virtual cellular manufacturing through agent-based modelling. *International Journal of Computer Integrated Manufacturing*, *30*(6), 564–579. https://doi.org/10.1080/0951192X.2016.1187294

Chamorro, V. G., Villagra, M., & Baran, B. (n.d.). *Optimización por Enjambre de Partículas para Satisfacción de Fórmulas Optimización por Enjambre de Partículas para Satisfacción de Fórmulas Booleanas*. (March 2014).

Dodge, C. F. V., & Lavalle, M. M. (2016). Analysis of Particle Swarm Optimization Algorithm Using a 3D Application. *Research in Computing Science, 116*(2016), 135–140.

Elanchezhian, C., Shanmuga Sundar, G., & Sundar Selwyn, T. (2007). Group Technology. *Computer Aided Manufacturing*, *9*, 14–18 (573).

Gen, L. A. (1859). *Introducci ´ on El Algoritmo Gen ´ etico Simple*. 1–34.

Han, W., Chen, Y., & Zhao, J. (2013). *A methodology based on resource elements for equipment capability analysis oriented virtual cellular manufacturing systems*. *715*, 3153–3160. https://doi.org/10.4028/www.scientific.net/AMR.712-715.3153

Han, W., & Tong, D. (2015). *Research on Detection and Resolution of Resource Conflict of Virtual Cell Considering New Task Insertion*. *6*. https://doi.org/10.1051/matecconf/20152201046

Han, W., Wang, F., & Lv, J. (2014). Virtual Cellular Multi-period Formation under the Dynamic Environment. *IERI Procedia*, *10*, 98–104. https://doi.org/10.1016/j.ieri.2014.09.097

Holland, J. (2012). *Algoritmo genético*. 42–48.

Huang, X., Guan, Z., & Yang, L. (2018). *An effective hybrid algorithm for multi-objective flexible job-shop scheduling problem*. *10*(9), 1–14. https://doi.org/10.1177/1687814018801442

Jayachitra, R., Revathy, A., & Prasad, P. S. S. (2010). A Fuzzy Programming approach for formation of Virtual Cells under dynamic and uncertain conditions. *International Journal of …*, *2*(6), 1708–1724.

Kelly, J., & Lawrence, D. (1991). A Hybrid Genetic Algorithm for Classification. *International Joint Conferences on Artificial Intelligence*, *91*, 645–650.

Kesen, S. E., Das, S. K., & Gungor, Z. (2010). A mixed integer programming formulation for scheduling of virtual manufacturing cells (VMCs). *International Journal of Advanced Manufacturing Technology*, *47*(5–8), 665–678. https://doi.org/10.1007/s00170-009-2231-4

Kesen, S. E., Das, S. K., & Güngör, Z. (2010). A genetic algorithm based heuristic for scheduling of virtual manufacturing cells (VMCs). *Computers and Operations Research*, *37*(6), 1148–1156. https://doi.org/10.1016/j.cor.2009.10.006

Kheirkhah, A. S., & Ghajari, A. (2018). *Uncertain Supply Chain Management*. *6*, 213–228. https://doi.org/10.5267/j.uscm.2017.7.001

Lee, M., & Su, L. (2014). *Social Accounting Matrix Balanced Based on Mathematical Optimization Method and General Algebraic Modeling System*. *4*(8), 1174–1190.

Liang, F., & Fung, R. Y. K. (2016). Coordination mechanism in real-time scheduling of Virtual Cellular Manufacturing Systems. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *230*(3), 534–547. https://doi.org/10.1177/0954405414555559

Lovay, M., Romero, E., & Peretti, G. (2016). *Aplicación del algoritmo de Optimización por Enjambre de Partículas en el dimensionamiento óptimo de componentes para Filtros Activos*. 13–24.

Lv, J., Yuan, J., & Han, W. M. (2013). Multi-Period Virtual Cell Formation: A New Methodology Based on Non-Linear Mixed-Integer Programming Model. *Applied Mechanics and Materials*, *278–280*, 2210–2217. https://doi.org/10.4028/www.scientific.net/AMM.278-280.2210

Mahdavi, I., Aalaei, A., Paydar, M. M., & Solimanpur, M. (2009). Production planning and cell formation in dynamic virtual cellularmanufacturing systems with worker flexibility. *2009 International Conference on Computers and Industrial Engineering, CIE 2009*.

Mahdavi, I., Aalaei, A., Paydar, M. M., & Solimanpur, M. (2014). Multi-objective cell formation and production planning in dynamic virtual cellular manufacturing systems. *International Journal of Production Research*, *49*(21), 6517–6537. https://doi.org/10.1080/00207543.2010.524902

Mak, K. L., Lau, J. S. K., & Wang, X. X. (2005). A genetic scheduling methodology for virtual cellular manufacturing systems: An industrial application. *International Journal of Production Research*, *43*(12), 2423–2450. https://doi.org/10.1080/00207540500046020

Mak, K. L., Ma, J., & Cui, L. X. (2011). Integrated Multi-period Production Scheduling and Cell Formation for Virtual Cellulara Manufacturing Systems. *Engineering Letters*, *19*(4).

Mak, K. L., Ma, J., & Su, W. (2010). Production scheduling for virtual cellular manufacturing systems with workforce constraints using a hybrid algorithm. *Proceedings - 2010 6th International Conference on Natural Computation, ICNC 2010*, *8*(Icnc), 4445–4449. https://doi.org/10.1109/ICNC.2010.5583491

Mak, K. L., Peng, P., Wang, X. X., & Lau, T. L. (2007). An ant colony optimization algorithm for scheduling virtual cellular manufacturing systems. *International Journal of Computer Integrated Manufacturing*, *20*(6), 524–537. https://doi.org/10.1080/09511920600596821

Mak, K. L., & Wang, X. X. (2002). *Sandoval: Celdas Peltier: Una alternativa para sistemas... - Google Académico*. 144–152.

Mertins, K., Friedland, R., & Rabe, M. (2000). Capacity assignment of virtual manufacturing cells by applying lot size harmonization. *International Journal of Production Research*, *38*(17), 4385–4391. https://doi.org/10.1080/00207540050205163

Miguel, A. S. De. (n.d.). *Computacional*.

Moradgholi, M., Mohammad Mahdi Paydar, B., Iraj Mahdavi, B., & Jouzdani, J. (2016). A genetic algorithm for a bi-objective mathematical model for dynamic virtual cell formation problem. *Journal of Industrial Engineering International*, *12*, 343–359.

https://doi.org/10.1007/s40092-016-0151-0

Moujahid, A., & Inza, I. (2008). Algoritmos genéticos. *Metodos Matematicos En Ciencias de La Computacion*, 1–34.

Mungwattana, A., Ellis, K. P., & Sturges, R. H. (2000). *Design of Cellular Manufacturing Systems for Dynamic and Uncertain Production Requirements with Presence of Routing Flexibility*.

Murali, R. V. (2012). Workforce assignment into virtual cells using learning vector quantization (LVQ) approach. *Research Journal of Applied Sciences, Engineering and Technology*, *4*(15), 2427–2435. https://doi.org/10.4028/www.scientific.net/AMR.576.705

Nandurkar, K. N., & Thomas, A. (2006). *Development of virtual cellular manufacturing systems for SMEs*.

Nikoofarid, E., & Aalaei, A. (2013). Production planning and worker assignment in a dynamic virtual cellular manufacturing system. *International Journal of Management Science and Engineering Management*, *7*(2), 89–95. https://doi.org/10.1080/17509653.2012.10671211

Nomden, G., Slomp, J., & Suresh, N. C. (2006). *Virtual manufacturing cells : A taxonomy of past research and identification of future research issues*. 71–92. https://doi.org/10.1007/s10696-006-8122-1

Nsakanda, A. L., Diaby, M., & Price, W. L. (2006). Hybrid genetic approach for solving large-scale capacitated cell formation problems with multiple routings. *European Journal of Operational Research*, *171*(3), 1051–1070. https://doi.org/10.1016/j.ejor.2005.01.017

Paydar, M. M., & Saidi-Mehrabad, M. (2015). Revised multi-choice goal programming for integrated supply chain design and dynamic virtual cell formation with fuzzy parameters.

International Journal of Computer Integrated Manufacturing, *28*(3), 251–265. https://doi.org/10.1080/0951192X.2013.874596

Paydar, M. M., & Saidi-Mehrabad, M. (2017). A hybrid genetic algorithm for dynamic virtual cellular manufacturing with supplier selection. *International Journal of Advanced Manufacturing Technology*, *92*(5–8), 3001–3017. https://doi.org/10.1007/s00170-017-0370-6

Rabbani, M., Farrokhi-Asl, H., Rafiei, H., & Khaleghi, R. (2017). Using metaheuristic algorithms to solve a dynamic cell formation problem with consideration of intra-cell layout design. *Intelligent Decision Technologies*, *11*, 109–126. https://doi.org/10.3233/IDT-160281

Rameshkumar, K., Suresh, R. K., & Mohanasundaram, K. M. (2005). *Discrete Particle Swarm Optimization (DPSO) Algorithm for Permutation Flowshop Scheduling to*. 572–581.

Ratchev, S. M. (2001). Concurrent process and facility prototyping for formation of virtual manufacturing cells. *Integrated Manufacturing Systems*, *12*(4), 306–315. https://doi.org/10.1108/09576060110393406

Rezazadeh, H., Mahini, R., & Zarei, M. (2011). Solving a dynamic virtual cell formation problem by linear programming embedded particle swarm optimization algorithm. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2010.12.018

Taylor, P., Arkat, J., & Ghahve, H. (2014). *International Journal of Computer Integrated Manufacturing Scheduling of virtual manufacturing cells with outsourcing allowed*. (September), 37–41. https://doi.org/10.1080/0951192X.2013.874581

Tesic, Z., Stevanov, B., Jovanovic, V., Tomic, M., & Kafol, C. (2016). Period Batch Control - A

production planning system applied to virtual manufacturing cells. *International Journal of Simulation Modelling*, *15*(2), 288–301. https://doi.org/10.2507/IJSIMM15(2)8.337

YIN, Y., & YASUDA, K. (2006). Similarity coefficient methods applied to the cell formation problem: A taxonomy and review. *International Journal of Production Economics*, *101*(2), 329–352. https://doi.org/10.1016/j.ijpe.2005.01.014