PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

RESOLUCIÓN DE MANUFACTURING CELL DESIGN PROBLEM UTILIZANDO LION OPTIMIZATION ALGORITHM

DANIEL LEÓN ESTAY MUÑOZ

Profesor Guía: **Ricardo Soto**
Profesores Co-referentes: **Rodrigo Olivares**

INFORME FINAL DE PROYECTO
PARA OPTAR AL TÍTULO PROFESIONAL DE
INGENIERO DE EJECUCIÓN EN INFORMÁTICA

Diciembre, 2017

Resumen

Este documento se enfoca en el modelado y resolución del problema de diseño de celdas de manufactura (Manufacturing Cell Design Problem, MCDP) a través del algoritmo de optimización de león (Lion Optimization Algorithm, LOA). El MCDP es un problema asociado a la producción lineal, en donde se busca ordenar las máquinas entre las celdas existentes de forma óptima, entendiéndose por óptimo la reducción del flujo entre celdas. En este proyecto se utiliza la metaheurística LOA para resolver este problema, la cual es una técnica inspirada en el comportamiento de los leones (nómadas y manada) en la naturaleza para buscar una solución. Se ilustran resultados experimentales donde se obtiene el óptimo global para varias instancias de este problema.

Palabras Clave: Problema de diseño de manufactura de celdas, óptimo global, producción lineal, Lion Optimization Algorithm.

Abstract

This document focuses on the modeling and resolution of the Manufacturing Cell Design Problem (MCDP), through the Lion Optimization Algorithm (LOA). The MCDP is a problem associated with linear production, where the idea is to organize the machines within the existing cells in an optimal way, that is, to minimize the part flow among cells. In this project, the LOA metaheuristics is used to solve this problem, which is a technique inspired by the behavior of lions (nomads and herds) in nature to find a solution. Experimental results are illustrated where the global optimum for several instances of this problem is obtained.

Keywords: Manufacturing cell design problem, optimum global, linear production, Lion Optimization Algorithm.

Índice

Lista de Figuras

Lista de Tablas

1. Introducción

Una de las principales preocupaciones para la industria manufacturera reside en determinar procesos eficientes que les permitan maximizar la producción y minimizar los gastos, los tiempos y los desechos. Debido a esta necesidad, surge la tecnología de grupos (Group Technology, GT), la que consiste en la agrupación de piezas pertenecientes a una familia de componentes que serán procesadas por máquinas dentro de una celda.

A partir de esto, surge el Manufacturing Cell Design Problem (MCDP), el que tiene por objetivo agrupar una serie de máquinas en una o más celdas tomando en cuenta las piezas que pueden ser procesadas en cada una de éstas, de tal manera que las celdas quedan los más independientemente posibles unas de otras. De esta manera se minimiza la cantidad de movimientos e intercambios de las piezas entre las celdas.

Debido a que el problema maneja un gran número de variables, demanda la utilización de métodos aproximados, como las metaheurísticas, las que a diferencia de los métodos exactos, no exploran todo el espacio de búsqueda, sino que aplican fórmulas de movimiento que controlan el proceso de exploración para disminuir la carga computacional y el tiempo de respuesta. Existen muchas investigaciones que involucran este problema con alguna metaheurística, en esta ocasión se utilizará Lion Optimization Algorithm.

Lion optimization algorithm o algoritmo de optimización de león, es una metaheurística basada en población, la que se inspira de la conducta de los leones frente a la naturaleza. La implementación de esta metaheurística para el problema se basa en las investigaciones hechas por Maziar Yazdani y Fariborz Jolai [14]. Respecto a la conducta de esta metaheurística frente al problema, cabe decir, que los resultados obtenidos fueron relativamente buenos, pero con un tiempo de ejecución un poco alto, más adelante se abordará este tema de mejor manera.

Este documento se estructura de la siguiente manera. En el segundo y tercer capítulo se presentan los objetivos y el estado del arte respectivamente. En el cuarto, se describe el modelo matemático del MCDP. En el quinto, se describe el comportamiento que posee Lion Optimization Algorithm. El análisis de los resultados obtenidos con la integración se presenta en el sexto capítulo. Finalmente las conclusiones son mostradas en el séptimo capítulo.

2. Objetivos

A continuación se presentan los objetivos asociados a la investigación (general y específicos).

2.1. Objetivo general

- Resolver el Manufacturing Cell Design Problem (MCDP) utilizando la metaheurística Lion Optimization Algorithm (LOA).

2.2. Objetivos específicos

- Analizar y comprender el MCDP.
- Analizar y comprender la metaheurística LOA.
- Integrar LOA con el MCDP.
- Realizar y analizar pruebas experimentales, haciendo uso de los problemas de Boctor.
- Ajustar parámetros de LOA para la obtención de mejores resultados.

3. Estado del Arte

La manufactura celular se deriva de los principios de la tecnología de grupo que fueron propuestos por Flander en 1925 [5]. Desde esa fecha se han realizado múltiples investigaciones, siendo la más influyente la realizada por Burbidge (análisis del flujo de producción) [3]. En esta se utiliza la matriz de incidencia máquina-pieza, y se reorganiza en una forma diagonal de bloque.

Visto desde un punto de vista algorítmico, este problema se ha tratado de resolver de dos maneras, utilizando métodos aproximados y optimización global. Los métodos aproximados, como la metaheurística, buscan encontrar una solución considerando el tiempo y los recursos disponibles; por lo tanto no es posible garantizar que se encuentre el óptimo global. En cambio, la optimización global, explora todo el espacio de búsqueda, para encontrar el óptimo global, pero debido a la inmensidad del espacio de las soluciones, este enfoque toma una gran cantidad de tiempo y recursos.

Existen muchas investigaciones realizadas a este problema que incluyen una determinada metaheurística, como por ejemplo; Migrating Birds Optimization (MBO) [4], propuesta por Ekrem Duman, Mitat Uysal y Ali Fuat Alkaya en el 2011, basada en la naturaleza, inspirada por la formación de vuelo en forma de "V" de las aves migratorias; Simulated Annealing (SA) [6], propuesta por Scott Kirkpatrick, C. Daniel Gelatt y Mario P. Vecchi en 1983, toma como inspiración el proceso de recocido del acero y cerámicas, la cual consiste en calentar y enfriar lentamente el material para variar sus propiedades físicas; Particle Swarm Optimization (PSO) [10] [11], propuestas originalmente por los investigadores Kennedy, Eberhart [10] y Shi [11], es una metaheurística que se inspira en el comportamiento de las partículas en la naturaleza; Bat Algorithm (BAT) [13], propuesto por Xin-She Yang, basado en el comportamiento de ecolocación de los murciélagos; Artificial Fish Swarm Algorithm (AFSA) [9] que estudia el comportamiento inteligente de un cardumen de peces; y, Shuffled Frog Leaping (SFL) [8], inspirado en las características meméticas de las ranas, donde las ranas tratan de saltar en el espacio de búsqueda en busca de un mejor resultado hasta que se cumpla la condición de detención.

Finalmente, para este trabajo de investigación, se ha utilizado la metaheurística inspirada en el comportamiento social de los leones (LOA) [7] para la resolución e integración del MCDP, propuesto por Maziar Yazdani y Fariborz Jolai.

4. Descripción de MCDP

El MCDP es un problema de manufactura presente en casi todas las fábricas de producción serializada donde se emplean 1 ó más máquinas para elaborar 1 ó más partes en 2 ó más celdas. El problema reside en la disminución del flujo interceldario, lo que permite disminuir tiempos, costos de producción y utilización de máquinas además de incrementar la productividad. El "flujo interceldario" hace referencia a los viajes que debe hacer una parte de una celda a otra para ser tratada por una máquina específica. Por lo tanto, el objetivo es la independencia de cada celda, y si no es posible, agrupar las máquinas de forma tal que la interacción de las mismas se reduzca al máximo.

4.1. Modelo matemático

Como en la mayoría de los problemas de optimización, para lograr resolverlo es necesario de un modelo matemático que involucra parámetros, variables, restricciones y función objetivo. Los parámetros y las variables se detallan a continuación:

Parámetros:

- M: Número de máquinas.
- P: Número de partes.
- C: Número de celdas.
- Mmax: Número máximo de máquinas por celdas
- A: Matriz de incidencia Máquinas x Partes con dominio binario..
- i: Índice de máquinas (i=1,...,M)
- j: Índice de partes (j=1,...,P)

Variables:

- k: Índice de celdas (k=1,...,C)
- Y: matriz Máquinas $\times$ Celdas con dominio binario.
- Z: matriz Partes $\times$ Celdas con dominio binario.

Los valores de las matrices estarán denotadas como:

$$A_{ij} = \begin{cases} 1 & \text{Si la máquina } i \text{ elabora la pieza } j \\ 0 & \text{En otro caso} \end{cases} \tag{1}$$

$$Y_{ik} = \begin{cases} 1 & \text{Si la máquina } i \text{ pertenece a la celda } k \\ 0 & \text{En otro caso} \end{cases} \tag{2}$$

$$Z_{jk} = \begin{cases} 1 & \text{Si la pieza } j \text{ es procesada en la celda } k \\ 0 & \text{En otro caso} \end{cases} \tag{3}$$

4.2. Restricciones y función objetivo

Las restricciones son condiciones que guían un problema matemático a la solución indicada, en cambio la función objetivo es utilizada para indicar el "valor" de la solución con el objetivo de elegir la mejor. EL MCDP posee 3 restricciones y una función objetivo, las cuales se dan a continuación:

- Una máquina puede pertenecer sólo a una celda:

$$\sum_{k=1}^{C} Y_{ik} = 1 \quad \forall i \tag{4}$$

- Una pieza pertenece a una familia de componentes:

$$\sum_{k=1}^{C} Z_{jk} = 1 \quad \forall j \tag{5}$$

- El número de máquinas en una celda no debe ser mayor al máximo permitido:

$$\sum_{i=1}^{M} Y_{ik} \leq M_{max} \quad \forall k \tag{6}$$

Como el MCDP busca minimizar las interacciones entre las celdas, la función objetivo estará denotada por:

$$Z = \sum_{k=1}^{C} \sum_{i=1}^{M} \sum_{j=1}^{P} A_{ij} Z_{jk} (1 - Y_{ik}) \tag{7}$$

4.3. Representación

La figura 1 indica la matriz máquinas-piezas inicial, antes de la optimización. Las matrices de la figura 2, representan las soluciones arrojadas por la metaheurística, después de la optimización, específicamente la matriz máquinas-celdas, que es a la cual se le aplicarán los movimientos matemáticos, la matriz piezas-celdas es resultado de un cálculo matemático entre la matriz inicial y la matriz máquinas-celdas. Una vez que el proceso de optimización termina, se genera la figura 3 a través de las matrices de la figura 2, en donde las celdas son claramente visibles debido a los "1" agrupados. Por lo tanto la celda 1 abarca los valores de la primera columna hasta la segunda columna y desde la primera fila hasta la tercera fila. Por el contrario la celda 2 abarca los valores de la columna tres hasta la columna 4 y desde la fila 4 hasta la fila 6. Los "1" que quedan fuera de estas celdas son excepciones, esto quiere decir que la pieza deberá viajar entre celdas para poder ser procesada.

Máquina \ Pieza	1	2	3	4
1	1	0	0	0
2	0	1	1	0
3	0	0	1	1
4	0	1	0	0
5	1	1	0	0
6	0	0	1	1

Figura 1: Ejemplo de MCDP - Matriz máquina-pieza inicial.

Máquina \ Celda	1	2
1	1	0
2	0	1
3	0	1
4	1	0
5	1	0
6	0	1

(a)

Pieza \ Celda	1	2
1	1	0
2	1	0
3	0	1
4	0	1

(b)

Figura 2: Ejemplo de MCDP - Matrices máquina-celda y pieza-celda.

Máquina \ Pieza	1	2	3	4
1	1	0	0	0
5	1	1	0	0
4	0	1	0	0
2	0	1	1	0
3	0	0	1	1
6	0	0	1	1

Figura 3: Ejemplo de MCDP - Matriz máquina-pieza final.

Para finalizar, tenemos que en la primera celda se encontrarán las máquinas 1, 5 y 4, las que procesarán las piezas 1 y 2; en la segunda celda, las máquinas 2, 3 y 6, las que trabajarán las piezas 2, 3 y 4.

5. Lion Optimization Algorithm

Lion optimization algorithm (LOA) es una metaheurística basada en población. Fue recientemente propuesta por Maziar Yazdani y Fariborz Jolai [7] en junio de 2015 utilizando las investigaciones posteriores de Wang [14] y Rajakumar [1]. Esta metaheurística se inspira en el comportamiento de los leones. Como se mencionó anteriormente, trabaja con un grupo de soluciones. Cada solución en LOA es representada por un león, dicho león puede pertenecer o no a una manada, sino pertenece a una, es un nómada. Los leones se dividen en masculino y femenino, los que se comportarán de distinta manera. LOA posee un número elevado de parámetros en comparación a otros algoritmos, estos están dados en la siguiente tabla:

Tabla 1: Parámetros de la metaheurística LOA.

Parámetros	Definición
Number of pride	Hace referencia al número de manadas, este valor no cambia durante la ejecución
Percent of nomad lions	Porcentaje de leones nómadas
Romaing Percent	Porcentaje de territorios a visitar por un león macho residente
Mutate Probability	Porcentaje que determina la probabilidad de que una cría salga con una mutación en sus genes
Sex Rate	Un porcentaje que se utiliza para determinar la cantidad de hembras dentro de una manada, en los nómadas este porcentaje determina la cantidad de machos
Mating probability	Probabilidad que las leonas de una manada se apareen con uno o más machos
Inmigrate rate	Porcentaje que determina la cantidad de leones hembras que saldrán de una manada para convertirse en nómadas y posteriormente lograr entrar a otra

Con respecto al procedimiento que utiliza LOA en su ejecución, se tienen los siguientes:

1. Inicialización de parámetros.

2. Hunting.
3. Moving toward to safe place.
4. Roaming.
5. Mating.
6. Defense.
7. Migration.

5.1. Inicialización de parámetros

Cada uno de los parámetros que deben inicializarse (además de la población y el número de iteraciones) tiene su determinado dominio. Number of pride debe ser un número entero mayor a 0; percent of nomad lions, un número decimal entre 0 y 1; Roaming percent, un número entre 0 y 1; Mutate probability, un número entre 0 y 1; Sex rate, un número entre 0 y 1; Mating probability, un número entre 0 y 1; y finalmente, Inmigrate rate, un número entre 0 y 1.

5.2. Hunting

Hunting es una palabra en inglés que significa "casería", como la palabra lo indica cada manada envía un grupo de hembras a cazar. Estas hembras son escogidas al azar y quedan inhabilitadas para realizar el siguiente movimiento. En este movimiento, se genera una presa ficticia (prey), la cual es el promedio de las posiciones de las cazadoras como se muestra en la fórmula:

$$Prey_j = \frac{\sum_{j=1}^{NC} Cazador_j}{NC} \tag{8}$$

Una vez generada la presa, cada hembra tendrá su turno para "atacarla", dicho ataque dependerá de la posición en la que se encuentre la hembra. Estas posiciones pueden ser centro, ala izquierda y derecha, sus movimientos están dadas por las fórmulas 9 y 10 respectivamente.

$$Cazador'_j = \begin{cases} \text{rand}(cazador_j, prey_j) & (2 \times prey_j - cazador_j) < prey_j \\ \text{rand}(prey_j, cazador_j) & (2 \times prey_j - cazador_j) > prey_j \end{cases} \tag{9}$$

$$Cazador'_j = \begin{cases} \text{rand}((2\times prey_j - cazador_j), prey_j) & cazador_j < prey_j \\ \text{rand}(prey_j, (2\times prey_j - cazador_j)) & cazador_j > prey_j \end{cases} \quad (10)$$

Una vez que una hembra se mueve hacia la presa, esta tiene la probabilidad de huir o no dependiendo del nuevo fitness o calidad del león (evaluación respecto a la función objetivo) de esta (fórmula 11).

$$Prey'_j = Prey_j + \text{rand}(0,1)\times\text{PI}\times(Prey_j - cazador_j) \quad (11)$$

5.3. Moving toward to safe place

Las hembras que no fueron a cazar se moverán aleatoriamente por los límites del territorio que tuvieron una mejora en su fitness desde la última iteración. Su movimiento estará denotado por la siguiente fórmula:

$$Hembra'_j = Hembra_j + 2\text{D}\times\text{rand}(0,1)\times\text{R1}+\text{U}(-1,1)\times\tan(\theta)\times\text{D}\times\text{R2} \quad (12)$$

Donde $Hembra_j$ es la posición actual de la leona, D muestra la distancia entre la posición de la leona y el punto escogido dentro del territorio. *R1* es un vector director con punto de inicio la posición actual de la leona y su dirección es hacia el punto escogido. *R2* es perpendicular a *R1*.

5.4. Roaming

Los machos dentro de una manada y también los leones de ambos sexos en los nómadas, se moverán según el grupo en el que estén. Por un lado, los nómadas se moverán aleatoriamente por el espacio, tanto hembras como machos. Por otro lado, los machos dentro de una manada se moverán hacia algunos puntos de su territorio, dichos puntos son seleccionados utilizando el parámetro Roaming percent, este movimiento será de x unidades, en donde x esta dado en la siguiente fórmula:

$$Macho'_j = \text{U}(0.2,\text{D}) \quad (13)$$

D es la distancia entre la posición actual del león y el punto seleccionado.

5.5. Mating

Un porcentaje de las hembras de cada manada serán seleccionadas para aparearse con uno o más machos pertenecientes a su misma manada (el porcentaje de selección corresponde al parámetro Mating probability) engendrando a su vez 2 nuevos cachorros. El sexo de estos cachorros será seleccionado al azar, también existe una probabilidad que uno de los cachorros mute (Mutate probability), la posición de las crías estarán dadas por las siguientes fórmulas:

$$cria1_j = \beta \times Hembra_j + \sum_{i=1}^{NM} \frac{1-\beta}{\sum_{k=1}^{NM} S_k} \times Macho_j \times S_i \tag{14}$$

$$cria2_j = (1-\beta) \times Hembra_j + \sum_{i=1}^{NM} \frac{\beta}{\sum_{k=1}^{NM} S_k} \times Macho_j \times S_i \tag{15}$$

Donde j es la dimensión, S es igual a 1 si el macho i está seleccionado para el apareamiento (azar), de lo contrario es igual a 0, NR es el número de machos residentes en una manda, β es un número generado al azar con una distribución normal con valor medio 0.5 y desviación estándar 0. En caso de que la cría mute, cada gen (dimensión) será un número al azar.

5.6. Defense

Cada macho de cada una de las manadas deberá defenderse de dos tipos de ataques. En primer lugar, la defensa contra los nuevos machos maduros de la manda, donde los que tengan los peores fitness serán expulsados de la manada y se convertirán en nómadas. En segundo lugar, la defensa contra los machos nómadas, cada macho nómada atacará un número aleatorio de manadas, si este es mejor a uno de los machos residente, entonces cambiarán de lugar.

5.7. Migration

Un porcentaje (Inmigrate rate) de las hembras de cada manada migrará y se convertirá en nómada. Una vez que las hembras migren, las hembras se clasificarán, las mejores serán enviadas a las manadas para rellenar el espacio de las hembras que migraron, de esta manera se mantiene el equilibrio en cada manada. Los peores nómadas machos y hembras serán eliminados respetando las proporciones iniciales de estos.

5.8. Pseudocódigo Lion Optimization Algorithm

De acuerdo a los métodos descritos anteriormente se tiene el siguiente pseudocódigo:

Algorithm 1 Lion Optimization Algorithm

1: *Generar* muestra aleatoria de leones N
2: *Inicializar manadas y leones nómadas*
3: **while** $(t < MaxIteration)$ **do**
4: *Por cada manada* algunas hembras son seleccionadas al azar para ir a cazar
5: *Por cada manada* las hembras que no fueron seleccionadas van hacia una de las mejores posiciones del territorio
6: *Por cada manada* Se selecciona un %R de territorios al azar para ser visitados por los machos residentes
7: *Por cada manada* Una porcentaje de %Ma (probabilidad de apareamiento) de hembras se aparea con uno o mas de los machos residentes
8: *Por cada manada* El macho más débil es expulsado de ella y se vuelve nómada
9: *Por cada nómada* Ambos hembra y macho se mueven randómicamente por el espacio de búsqueda
10: *Por cada nómada* Una porcentaje de %Ma (probabilidad de apareamiento) de hembras se aparea con uno de los mejores machos
11: *Por cada nómada* Los machos atacan aleatoriamente las mandas, si tiene éxito reemplazan al residente
12: *Por cada manada* Según la tasa de inmigración (I) algunas hembras abandonas la manada y se vuelven nómadas
13: *Hacer* En base al valor del fitness cada genero de nómadas se clasifica por separado. Las mejores hembras entre ellas son seleccionadas y distribuidas a las manadas llenando los vacíos de las hembras que migraron
14: *Hacer* Respecto al numero máximo permitido de leones nómadas de cada genero, los leones con el menor fitness serán eliminados
15: **end while**

6. Resultados Experimentales

La metaheurística LOA integrada con el MCDP fue codificada en JAVA JDK versión 1.8. El programa fue ejecutado en una laptop HP con las siguientes características: un procesador AMD A8-7410 con 2.2 GHz, 4Gb de RAM, sistema operativo Windows 10 de 64 bits. Las pruebas experimentales constan de 90 instancias de prueba (10 problemas considerando 5 valores para Mmax con C=2 y 10 problemas considerando 4 valores de Mmax con C=3) pertenecientes a las investigaciones de Boctor [2]. Los experimentos para LOA fueron aplicados 30 veces para cada problema de Boctor [2], los valores de los parámetros son dados a continuación:

Tabla 2: Valores de parámetros para pruebas de LOA.

Parámetros	Valores
Población	120
Iteraciones	200
Number of pride	3
Percent of nomad lions	0.5
Romaing Percent	0.5
Mutate Probability	0.4
Sex Rate	0.8
Mating probability	0.8
Inmigrate rate	0.8

Los valores obtenidos tras las pruebas están resumidas en las tablas 3 y 4. Los valores de la tabla 3 corresponden a los 10 problemas con C=2, y la tablas 4 representa los 10 problemas con C=3. La descripción de cada columna de las tablas se muestra a continuación. La primera columna indica la ID del problema de Boctor [2]. La segunda columna corresponde al número de máquinas. Las siguientes dos columnas son los resultados de LOA, en donde la primera indica el mejor valor al cual se llegó (óptimo) y la segunda al promedio de los óptimos obtenidos en las 30 ejecuciones. Por último, las columnas siguientes son resultados obtenidos a través de otras metaheurística como Migrating Birds Optimization(MBO), Simulated Annealing(SA) y Particle Swarm Optimization (PSO) respectivamente, dichos valores fueron extraídos de la investigación, Resolucion del manufacturing cell design problem utilizando flower pollination algorithm [12].

Tabla 3: Experimentos de Boctor usando C = 2

Boctor Problem	Mmax	Valor Óptimo	LOA		SA	PSO	MBO	SFLA
			Óptimo	Promedio				
1	8	11	11	11.00	11	11	11	11
1	9	11	11	11.00	11	11	11	11
1	10	11	11	11.00	11	11	11	11
1	11	11	11	11.00	11	11	11	11
1	12	11	11	11.00	11	11	11	11
2	8	7	7	7.00	7	7	7	7
2	9	6	6	6.17	6	6	6	6
2	10	4	4	4.77	10	5	4	4
2	11	3	3	3.10	4	4	3	3
2	12	3	3	3.10	3	4	3	3
3	8	4	4	4.07	5	5	4	4
3	9	4	4	4.00	4	4	4	4
3	10	4	4	4.00	4	5	4	4
3	11	3	3	3.33	4	4	3	3
3	12	1	1	1.73	4	3	1	1
4	8	14	14	14.00	14	15	14	14
4	9	13	13	13.00	13	13	13	13
4	10	13	13	13.00	13	13	13	13
4	11	13	13	13.00	13	13	13	13
4	12	13	13	13.00	13	13	13	13
5	8	9	9	9.13	9	10	9	9
5	9	6	6	6.33	6	8	6	6
5	10	6	6	6.27	6	6	6	6
5	11	5	5	5.50	7	5	5	5
5	12	4	4	4.37	4	5	4	4
6	8	5	5	5.00	5	5	5	5
6	9	3	3	3.13	3	3	3	3
6	10	3	3	3.27	5	3	3	3
6	11	3	3	3.07	3	4	3	3
6	12	2	2	2.43	3	4	2	2
7	8	7	7	7.00	7	7	7	7
7	9	4	4	4.10	4	5	4	4
7	10	4	4	4.00	4	5	4	4
7	11	4	4	4.13	4	5	4	4
7	12	4	4	4.33	4	5	4	4
8	8	13	13	13.00	13	14	13	13
8	9	10	10	10.00	20	11	10	10
8	10	8	8	8.03	15	10	8	8
8	11	5	5	5.53	11	6	5	5
8	12	5	5	5.27	7	6	5	5
9	8	8	8	8.00	13	9	8	8
9	9	8	8	8.00	8	8	8	8
9	10	8	8	8.07	8	8	8	8
9	11	5	5	5.90	8	5	5	5
9	12	5	5	5.70	8	8	5	5
10	8	8	8	8.00	8	9	8	8
10	9	5	5	5.20	5	8	5	5
10	10	5	5	5.03	5	7	5	5
10	11	5	5	5.20	5	7	5	5
10	12	5	5	5.33	5	6	5	5

Tabla 4: Experimentos de Boctor usando C = 3

Boctor Problem	Mmax	Valor Óptimo	LOA		SA	PSO	MBO	SFLA
			Óptimo	Promedio				
1	6	27	27	28.83	28	-	27	-
1	7	18	18	19.03	18	-	18	-
1	8	11	11	11.56	11	-	11	-
1	9	11	11	11.20	11	-	11	-
2	6	7	7	9.37	7	-	7	-
2	7	6	6	7.96	6	-	6	-
2	8	6	6	7.30	7	-	6	-
2	9	6	6	6.40	12	-	6	-
3	6	9	9	10.43	8	-	9	-
3	7	4	4	7.17	8	-	4	-
3	8	4	4	5.33	4	-	4	-
3	9	4	4	4.53	27	-	4	-
4	6	27	27	27.63	18	-	27	-
4	7	18	18	18.50	14	-	18	-
4	8	14	14	14.10	13	-	14	-
4	9	13	13	13.43	11	-	13	-
5	6	11	11	12.30	9	-	11	-
5	7	8	8	12.07	9	-	8	-
5	8	8	8	10.00	8	-	8	-
5	9	6	6	7.23	8	-	6	-
6	6	6	6	7.80	5	-	6	-
6	7	4	4	5.97	5	-	4	-
6	8	4	4	5.30	4	-	4	-
6	9	3	3	3.97	11	-	3	-
7	6	11	11	13.23	5	-	11	-
7	7	5	5	7.60	5	-	5	-
7	8	5	5	6.93	5	-	5	-
7	9	4	4	5.53	14	-	4	-
8	6	14	14	15.10	11	-	14	-
8	7	11	11	11.93	11	-	11	-
8	8	11	11	13.27	10	-	11	-
8	9	10	10	11.17	12	-	10	-
9	6	12	12	14.67	12	-	12	-
9	7	12	12	13.46	13	-	12	-
9	8	8	8	10.16	8	-	8	-
9	9	8	8	9.33	8	-	8	-
10	6	10	10	13.14	10	-	10	-
10	7	8	8	9.83	8	-	8	-
10	8	8	8	8.50	8	-	8	-
10	9	5	5	6.53	5	-	5	-

Como se puede observar que en la tabla 3, en todos los problemas se alcanzó el óptimo. De forma más detallada, se tiene que 21 problemas lograron un promedio igual al óptimo, esto quiere decir que en estos problemas la probabilidad de que se logre sacar el mejor resultado es de más de un 95 %. Respecto al porcentaje de mejora, se tiene que los problemas con celda 2 mejoraron sus resultados hasta un 34 %, logrando obtener una variación respecto al promedio mínima de 0 y máxima de 0.76, mejor que los anteriores 0.16 y 2.27.

Por otro lado, en la tabla 4, en todos los problemas se alcanzó el óptimo a diferencia de la vez pasada donde sólo se encontraron 38 de 40, lo que también trajo consigo una mejora significativa en los promedios. A diferencia de los problemas con celda 2 no se lograron promedios perfectos, pero se redujeron en gran medida. Se logró una mejora de hasta el 47 % respecto a los resultados anteriores, con un mínimo de 0.2 y un máximo de 4.07, en donde los anteriores fueron 2.67 y 10.67 respectivamente.

7. Conclusión

Como se pudo observar durante todo el documento, el trabajo de integración entre el MCDP y la metaheurística, requiere de un buen entendimiento en ambas partes. La parte que requiere mayor atención es el momento en el que se le aplican los movimientos de la metaheurística a la matriz máquinas-piezas, puesto que algún error dentro de esto casi siempre tiene grandes consecuencias en los resultados que se obtienen.

En cuanto a los resultados obtenidos en las pruebas, se logró encontrar todos los óptimos globales de los problemas de Boctor. De forma más detallada, se tiene que la brecha entre los promedios y el óptimo global de los problemas han disminuido considerablemente. Con respecto al tiempo de ejecución, cada problema se demora aproximadamente 40 segundos en ejecutarse, cifra que es mayor que en los experimentos anteriores, pero se entiende por la existencia de más ciclos y de cambios realizados en el algoritmo.

Respecto a los cambios hechos en el algoritmo, se solucionaron algunos errores y además se añadieron ciclos "while" para la generación de nuevas posiciones, aumentando la probabilidad de que se obtenga una solución válida por cada movimiento. Se modificó el número de cazadoras seleccionadas en el método Hunting a 50 % de las hembras en la manada en comparación a la cantidad invariable de 7, aumentando la efectividad en manadas grandes.

Para finalizar, aunque los resultados obtenidos mejoraron considerablemente, aún hay mucho trabajo por hacer. En primer lugar se tiene pensado probar 35 problemas además de los de Boctor, si los resultados obtenidos no son buenos, se procederá a mejorar el método ajustador. Y en segundo lugar, se tiene pensado implementar Autonomus Search, primero en algunos parámetros y dependiendo de los resultados se irán aumentando el número de parámetros que variaran. Por último, aunque fue un tanto complicado lograr buenos resultados, me siento satisfecho con los obtenidos y en lo que el tiempo me permita, me gustaría continuar trabajando en este problema.

Referencias

[1] B. Cheng B. Wang, X. Jin. Lion pride optimizer: an optimization algorithm inspired by lion pride behavior. *Sci. China Inf. Sci.*, 55(10):2369–2389, 2012.

[2] F. F. Boctor. A linear formulation of the machine-part cell formation problem. *International Journal of Production Research.*, 29(2):343–356, 1991.

[3] J.L. Burbidge. Production flow analysis. *Production Engineer*, 42(12):742–752, 1963.

[4] Mitat Uysal y Ali Fuat Alkaya Ekrem Duman. Migrating Birds Optimization: A New Meta-heuristic Approach and Its Application to the Quadratic Assignment Problem. *European Conference on the Applications of Evolutionary Computation*, 6624(1):254–263, 2011.

[5] R. Flanders. Design, manufacture, and production control of a standard machine. *Transactions of ASME*, 46, 1925.

[6] Gelatt C. D. y Vecchi M. P. Kirkpatrick S. Optimization by Simulated Annealing. *Science*, 4598(220):671–680, 1983.

[7] B. Rajakumar. Lions Algorithm: a new nature-inspired search algorithm. *Procedia Technol.*, 6(10):126–135, 2012.

[8] R. Soto, B. Crawford, E. Vega, and F. Paredes. Solving manufacturing cell design problems using a shuffled frog leaping algorithm. *Beni Suef, Egypt Volume 407 of the series Advances in Intelligent Systems and Computing*, pages 253–261, 2015.

[9] R. Soto, B. Crawford, E. Vega, and F. Paredes. Solving manufacturing cell design problems using an artificial fish swarm algorithm. *Volume 9413 of the series Lecture Notes in Computer Science*, pages 282–290, 2015.

[10] Kennedy J. y Eberhart R. Particle Swarm Optimization. *Proceedings of IEEE International Conference on Neural Networks*, 4(1):1942–1948, 1995.

[11] Shi Y. y Eberhart R. A modified particle swarm optimizer. *Proceedings of IEEE International Conference on Evolutionary Computation*, 1(1):69–73, 1995.

[12] M. M. De Conti Rivara y R. A. Rubio Hurtado. Resolucion del manufacturing cell design problem utilizando flower pollination algorithm. *Pontificia Universidad Catolica de Valparaiso*, 1(1):15–20, 2017.

[13] Xin-She Yang. A new metaheuristic bat-inspired algorithm. *Department of Engineering, University of Cambridge*, 284:65–74, 2010.

[14] Maziar Yazdani and Fariborz Jolai. Lion Optimization Algorithm (LOA): A nature-inspired metaheuristic algorithm. *Computational Design and Engineering*, 3(1):24–36, 2016.