

Pontificia Universidad Católica de Valparaíso  
Facultad de Ingeniería  
Escuela de Ingeniería Informática

**RESOLUCIÓN DEL MANUFACTURING CELL DESIGN  
PROBLEM MEDIANTE LA METAHEURÍSTICA BIG BANG  
BIG CRUNCH**

**Edgardo Andrés Zelada Segovia**

Profesor Guía: **Ricardo Soto De Giorgis**  
Profesor Co-referente: **Boris Almonacid Gutiérrez**

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

Abril 2017

Pontificia Universidad Católica de Valparaíso

Facultad de Ingeniería

Escuela de Ingeniería Informática

**RESOLUCIÓN DEL MANUFACTURING CELL DESIGN  
PROBLEM MEDIANTE LA METAHEURÍSTICA BIG BANG  
BIG CRUNCH**

**Edgardo Andrés Zelada Segovia**

Profesor Guía: **Ricardo Soto De Giorgis**

Profesor Co-referente: **Boris Almonacid Gutiérrez**

Carrera: **Ingeniería de Ejecución en Informática**

Abril 2017

## Dedicatoria

*Proyecto dedicado a todos los que me dieron su apoyo.  
Gracias a mi familia que siempre estuvieron presentes para darme palabras de aliento.  
A mi padre por mostrarme el hombre que quiero ser.  
A mi madre por su amor incondicional.  
A mi pareja que en poco tiempo se transformó en un pilar fundamental en mi vida.  
Y amigos, de los cuales nunca dude en poder contar con ellos.*

Edgardo Zelada Segovia.

## Resumen

El presente proyecto tiene por objetivo modelar y resolver el Manufacturing Cell Design Problem (MCDP) mediante la implementación de la metaheurística Big Bang Big Crunch (BB-BC). El MDCP consiste en la creación de un diseño óptimo de una planta de producción. Esta se compone de celdas que a su vez se componen de máquinas, las cuales fabrican distintas piezas. El objetivo es minimizar el flujo de movimiento de piezas entre celdas para así reducir los costos de producción y aumentar la productividad. Para resolver el MCDP se utiliza el algoritmo de optimización BB-BC, el cual se basa en la teoría del origen del universo. Se realizaron pruebas experimentales en noventa instancias, alcanzando la totalidad de los óptimos globales.

**Palabras Claves:** Manufacturing Cell Design Problem, Metaheurística, Big Bang Big Crunch.

## Abstract

This project aims to model and solve the Manufacturing Cell Design Problem (MCDP) by the implementation of the Big Bang Big Crunch (BB-BC) metaheuristic. The MCDP consists in creating an optimal production plant layout. The production plant is composed of cells which in turn are composed of machines that process families of product parts. The goal is to minimize the part flow among cells in order to reduce production costs and increase productivity. To this end we use the BB-BC optimization algorithm, which is based on the theory of the universe origin. We perform experimental tests on ninety instances where all the global optimums were reached.

**Key-words:** Manufacturing cell design, Metaheuristic, Big Bang Big Crunch, Optimization .

# Índice

<b>1</b>	<b>Introducción</b>	<b>1</b>
<b>2</b>	<b>Definición de Objetivos</b>	<b>2</b>
2.1	Objetivo General	2
2.2	Objetivos Específicos	2
<b>3</b>	<b>Estado del Arte</b>	<b>3</b>
<b>4</b>	<b>Manufacturing Cell Design Problem</b>	<b>5</b>
4.1	Definición del problema	5
4.2	Modelo matemático	5
<b>5</b>	<b>Metaheurística</b>	<b>8</b>
5.1	Características	9
<b>6</b>	<b>Big Bang Big Crunch</b>	<b>10</b>
6.1	Fase del Big Bang	10
6.2	Fase del Big Crunch	10
<b>7</b>	<b>Resolución de MCDP con Big Bang Big Crunch</b>	<b>12</b>
7.1	Big Bang	12
7.2	Big Crunch	13
7.3	Funciones de Binarización	13
7.4	Pseudocódigo Big Bang Big Crunch	15
<b>8</b>	<b>Experimentación</b>	<b>16</b>
8.1	Resultados	16
8.2	Testing de funciones de binarización	20
<b>9</b>	<b>Conclusión</b>	<b>22</b>
	<b>Referencias</b>	<b>23</b>

## Lista de Figuras

4.1	Matriz de incidencia inicial . . . . .	6
4.2	Izquierda: Matriz Máquina-Celda $y_{ik}$ . Derecha: Matriz Parte-Celda $z_{jk}$ . . . . .	7
4.3	Matriz de incidencia reordenada. . . . .	7
7.4	Vector inicial . . . . .	12
7.5	Vector intermedio . . . . .	12
7.6	Nuevo vector candidato . . . . .	13
7.7	Visualización de las funciones de tranferencia V-Shaped y S-Shaped . . . . .	14
8.8	Gráfico de convergencia del problema 4 de Boctor con $C=3$ y $M_{\text{máx}}=6$ . . . . .	19
8.9	Gráfico de convergencia del problema 4 de Boctor con $C=2$ y $M_{\text{máx}}=8$ . . . . .	19

## Lista de Tablas

8.1	Resultados Experimentales con $C=2$ . . . . .	16
8.2	Resultados Experimentales con $C=2$ . . . . .	17
8.3	Resultados Experimentales con $C=3$ . . . . .	17
8.4	Resultados Experimentales con $C=3$ . . . . .	18
8.5	Testing de funciones de binarización instancias B01 y B02. BB-BC FCM, BB-BC utilizando la fórmula 8 como centro de masa. BB-BC ACM, BB-BC utilizando la aptitud como centro de masa. . . . .	20
8.6	Testing de funciones de binarización instancias B51 y B52. BB-BC FCM, BB-BC utilizando la fórmula 8 como centro de masa. BB-BC ACM, BB-BC utilizando la aptitud como centro de masa. . . . .	21

# 1 Introducción

Es cada vez más frecuente encontrar empresas con un alto grado de automatización, las cuales utilizan celdas de manufactura en sus procesos de fabricación. El uso de tales dispositivos les permite obtener altos grados de eficiencia en la producción, mantener estándares elevados de calidad y la capacidad de realizar con rapidez las modificaciones que requiere el proceso productivo. Esto, con la finalidad de adecuarse a nuevas necesidades del mercado. Aunque se pretende que una celda de manufactura permita adecuaciones eficientes, en la realidad, el volver a programar una celda es una labor delicada y compleja. Adecuar una celda para la fabricación de un nuevo producto, requiere de tiempo y de personal especializado. Además, la celda debe detenerse durante el tiempo que dure la adecuación, interrumpiendo así el proceso productivo.

Esta problemática nace a través de la constante búsqueda de empresas y organizaciones de encontrar soluciones que generen una mayor productividad, reduciendo los costes de producción. Dado esto, el problema de formulación de celdas ha sido objeto de diversas investigaciones, en donde Burbidge [7], con su análisis de flujo de producción en el año 1963, convierte su procedimiento en uno de los primeros para resolver esta problemática.

La presente investigación busca encontrar una solución a un problema clásico de agrupación de celdas, el Manufacturing Cell Design Problem (MCDP). El MCDP es una estrategia de manufacturación que consiste en la creación de un diseño óptimo de plantas de producción, buscando una asignación de máquinas y piezas de manera tal que se minimice el movimiento e intercambio de éstas entre familias llamadas celdas. Para poder resolver el MCDP se utilizará la metaheurística Big Bang Big Crunch (BB-BC) [9]. BB-BC es un algoritmo que se basa en la teoría de la evolución del universo, llamada la teoría del Big Bang Big Crunch. En la fase del Big Bang, la disipación de energía produce desorden y la aleatoriedad es la principal característica de esta fase; mientras que en la fase del Big Crunch, las partículas distribuidas de forma aleatoria son llevadas a un orden.

Esta investigación se estructura de la siguiente forma: En el capítulo 2, se especifican los objetivos generales y específicos del proyecto. En el capítulo 3 se exponen investigaciones previas concernientes al MCDP. En el capítulo 4 se describe el problema de MCDP. En el capítulo 5 se describen conceptos básicos de las metaheurísticas. En el capítulo 6 se explica la metaheurística BB-BC. En el capítulo 7 se explica la integración de la metaheurística BB-BC con el problema de MCDP. En el capítulo 8 se presentan los resultados obtenidos al implementar la metaheurística BB-BC para resolver el MCDP. Finalmente en el capítulo 9, se presentan las conclusiones y directrices para el trabajo futuro.



## 2 Definición de Objetivos

### 2.1 Objetivo General

Comprender y resolver el Manufacturing Cell Design Problem mediante la metaheurística Big Bang Big Crunch para poder en un punto mas avanzado de la investigación comparar y analizar los resultados obtenidos.

### 2.2 Objetivos Específicos

- Comprender el Manufacturing Cell Design Problem.
- Resolver el Manufacturing Cell Design Problem usando Big Bang Big Crunch.
- Estudiar y analizar los resultados arrojados por la implementación desarrollada.

### 3 Estado del Arte

Las celdas de manufactura han emergido en estas últimas dos décadas como innovación para la estrategia de manufacturación, la cual recoge las ventajas de la fabricación en serie de productos. Sin embargo, la independencia entre celdas es difícil de generar en la práctica, debido a que algunas partes necesitan ser procesadas en más de una máquina. Por lo tanto, la meta del problema para la formación de celdas, reside en agrupar las máquinas de manera que se minimice el flujo entre las mismas.

En relación con las investigaciones realizadas, el problema de la formulación de celdas ha tenido dos líneas de investigación complementarias. Éstas se pueden organizar en dos grupos: Métodos aproximados y optimización global. Los métodos aproximados, como metaheurísticas, se centran en la búsqueda de una solución aproximada en una cantidad fija de tiempo; por lo tanto, no pueden garantizar un óptimo global. Por el contrario, la optimización global tiene como objetivo analizar el espacio de búsqueda, con el fin de garantizar un óptimo global, como consecuencia, el coste computacional en términos de memoria y el tiempo consumido es mayor.

Algunos métodos sólo intentan encontrar la familia de piezas, dando como resultado una solución parcial al problema; ya que la identificación de las familias de piezas requiere máquinas que procesen todas las partes dentro de una misma celda. Por lo general, estos métodos están basados en la clasificación y codificación de esquemas, y grupo de partes de acuerdo a su proceso para determinar coeficientes similares. Una formulación p-mediana para formar familias de piezas es dada por Kusiak [12].

Otros enfoques tratan de determinar no sólo las familias de piezas, sino también los grupos de máquinas. La mayoría de estos métodos se basan en la matriz de incidencia máquina×pieza, y se pueden dividir en agrupación jerárquica y no jerárquica. Como ejemplo a esto se tiene a Shargal [21]; Seifoddini y Hsu [19]; Srinivasan [27]; Atmani [4]; Adil [1]; Kusiak y Chow [13]; Purcheck [17]; Olivia López y Purcheck [16]; y Boctor [5].

La programación por metas o Goal Programming (GP), es otro paradigma de optimización global. GP se puede ver como una generalización de la programación lineal utilizada para el manejo de múltiples funciones objetivo (Sankaran [18]; y Shafer y Rogers [20]). Las técnicas híbridas y metaheurísticas de optimización global también se pueden encontrar en este grupo, por ejemplo Boulif y Atif [6] combinan la técnica de ramificación y poda o Branch and Bound (BB) con un algoritmo genético o Genetic Algorithms (GA) [10].

Por otro lado, se han utilizado diferentes metaheurísticas para la formación de celdas, por ejemplo Aljaber, Baek, y Chen [2] y Lozano, Díaz, Eguía y Onieva [14], utilizan Tabu Search. Wu, Chang y Chung [29] presentan un enfoque de enfriamiento simulado o Simulated Annealing (SA) combinándolo con GA [10].

Durán, Rodríguez, y Consalter [8] combinan las partículas de optimización de enjambres con una técnica de minería de datos. Venugopal y Narendran [28] proponen el uso de GA. James, Brown, y Keeling [11] presentan una solución híbrida que incluye una búsqueda local y GA [10]. Finalmente, Nsakanda, Diaby, y Price [15] proponen una solución metodológica basada en la combinación de GA [10] y técnicas de optimización a gran escala.

Se han realizado diversas investigaciones para resolver el problema del Manufacturing Cell Design Problem utilizando metaheurísticas, entre las que se pueden destacar Migration Birds Optimization Algorithm [22], Dolphin Echolocation Algorithm [23], Firefly Algorithm [24], Flower Pollination Algorithm [25], Modified Binary Firefly Algorithm and Egyptian Vulture Optimization Algorithm [3].

## 4 Manufacturing Cell Design Problem

### 4.1 Definición del problema

El Manufacturing Cell Design Problem (MCDP) es una estrategia de manufactura que fracciona un sistema de producción en pequeños grupos denominados celdas. Las celdas se enfocan en la producción de un conjunto de piezas o componentes. Esta estrategia de producción es fuertemente influenciada por la filosofía Group Technology (GT) [26], la cual postula que cosas similares deberían ser fabricadas de manera similar. El objetivo de las celdas de manufactura es distribuir la totalidad de piezas fabricadas y máquinas utilizadas para este proceso, de manera tal que se formen familias determinadas de acuerdo a la similitud entre partes.

### 4.2 Modelo matemático

El objetivo del MCDP consiste en minimizar los movimientos e intercambio de partes entre celdas, con el fin de reducir costos de producción y aumentar la productividad [5]. La idea es representar los requisitos de procesamiento de las piezas de máquinas a través de una matriz de incidencia máquina-pieza. Esta matriz es de dominio binario y se denota como  $a_{ij}$  en donde:

$$a_{ij} = \begin{cases} 1 & \text{Si la parte } j_{th} \text{ visita la máquina } i_{th}. \\ 0 & \text{En otro caso.} \end{cases} \quad (1)$$

Una formulación matemática rigurosa del problema de agrupamiento máquina-pieza viene dada por el modelo de optimización que se indica a continuación. Sea:

- $M$ , el número de máquinas.
- $P$ , el número de piezas.
- $C$ , el número de celdas.
- $i$ , el índice de máquinas ( $i = 1, 2, \dots, M$ ).
- $j$ , el índice de piezas ( $j = 1, 2, \dots, P$ ).
- $k$ , el índice de celdas ( $k = 1, 2, \dots, C$ ).
- $M_{max}$ , el número máximo de máquinas por celda.
- $A = [a_{ij}]$ , la matriz de incidencia binaria máquina-pieza.
- $B = [b_{ik}]$ , la matriz de incidencia binaria máquina-celda.
- $C = [c_{jk}]$ , la matriz de incidencia binaria pieza-celda.

$$b_{ik} = \begin{cases} 1 & \text{Si la máquina } i \text{ esta en la celda } k. \\ 0 & \text{En otro caso.} \end{cases} \quad (2)$$

$$c_{jk} = \begin{cases} 1 & \text{Si la pieza } j \text{ esta en la celda } k. \\ 0 & \text{En otro caso.} \end{cases} \quad (3)$$

La función objetivo está dada por la ecuación 4

$$\min \sum_{k=1}^C \sum_{i=1}^M \sum_{j=1}^P a_{ij} c_{jk} (1 - b_{ik}) \quad (4)$$

Para finalizar, la función objetivo presentada de manera previa, está sujeta a las siguientes restricciones, en donde cada una de ellas asegura que:

- Solo una máquina sea asignada a una celda.

$$\sum_{k=1}^C b_{ik} = 1 \quad \forall i \quad (5)$$

- Solo una parte sea asignada a una celda.

$$\sum_{k=1}^C c_{jk} = 1 \quad \forall j \quad (6)$$

- Que un número máximo de máquinas sean asignadas a una celda.

$$\sum_{i=1}^M b_{ik} \leq M_{max} \quad \forall k \quad (7)$$

A modo de ejemplo se presenta en la figura 4.1 una matriz máquina-pieza inicial, la cual contiene el detalle sobre qué máquinas elaboran ciertas piezas.

	Pieza 1	Pieza 2	Pieza 3	Pieza4	Pieza 5	Pieza 6	Pieza 7
Máquina 1	0	1	0	1	1	1	0
Máquina 2	1	0	1	0	0	0	0
Máquina 3	1	0	1	0	0	0	1
Máquina 4	0	1	0	1	0	1	0
Máquina 5	1	0	0	0	0	0	1

**Figura 4.1** Matriz de incidencia inicial

Las matrices de solución para el ejemplo pueden ser visualizadas en la figura 4.2. La matriz satisface las restricciones 5, 6 y 7. Usando las matrices de solución de la figura 4.2, la matriz de la figura 4.1 es ordenada para generar una nueva matriz de incidencia, la cual se puede apreciar en la figura 4.3.

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

	Celda 1	Celda 2
Pieza 1	1	0
Pieza 2	0	1
Pieza 3	1	0
Pieza 4	0	1
Pieza 5	0	1
Pieza 6	0	1
Pieza 7	1	0

**Figura 4.2** Izquierda: Matriz Máquina-Celda  $y_{ik}$ . Derecha: Matriz Parte-Celda  $z_{jk}$ .

		Celda 1			Celda 2			
		Pieza 1	Pieza 3	Pieza 7	Pieza 2	Pieza 4	Pieza 5	Pieza 6
Celda 1	Máquina 2	1	1	0	0	0	0	0
	Máquina 3	1	1	1	0	0	0	0
	Máquina 5	1	0	1	0	0	0	0
Celda 2	Máquina 1	0	0	0	1	1	1	1
	Máquina 4	0	0	0	1	1	0	1

**Figura 4.3** Matriz de incidencia reordenada.

## 5 Metaheurística

Desde los años sesenta han ido apareciendo diferentes métodos de resolución de problemas de optimización. Estos métodos son conocidos por el nombre de heurísticas y metaheurísticas. Estas técnicas son de especial interés en el caso de variables enteras aunque también se aplican a problemas con variables reales.

Las técnicas heurísticas se utilizan cuando no existe un método exacto de resolución, cuando existe un método exacto que consume mucho tiempo para ofrecer la solución óptima, cuando existen limitaciones de tiempo o como paso intermedio para obtener una solución inicial para la aplicación de otra técnica. Algunos autores proponen la siguiente clasificación de métodos de resolución mediante heurísticas:

- Métodos constructivos, que se caracterizan por construir una solución definiendo diferentes partes de ella en sucesivos pasos.
- Métodos de descomposición, dividen el problema en varios más pequeños y la solución final se obtiene a partir de la solución de cada uno de estos.
- Métodos de reducción, tratan de identificar alguna característica de la solución que permita simplificar el tratamiento del problema.
- Métodos de manipulación del modelo, obtienen una solución del problema original a partir de una solución de otro problema simplificado (con menos restricciones, dejando lineal el problema, entre otras.)
- Métodos de búsqueda por entornos, en las que se parte de una solución inicial a la que se le realizan modificaciones en iteraciones sucesivas para obtener una solución final. En cada iteración existe un conjunto de soluciones vecinas candidatas a ser nueva solución en el proceso. En este grupo se encuentran las técnicas metaheurísticas.

Las técnicas metaheurísticas son procedimientos de búsqueda que tampoco garantizan la obtención del óptimo del problema y que también se basan en la aplicación de reglas relativamente sencillas. A diferencia de las heurísticas, las técnicas metaheurísticas tratan de huir de los óptimos locales. La aplicación de dichas técnicas es especialmente interesante en problemas de optimización combinatorial; problemas en los que las variables de decisión son enteras o discretas, o en los cuales el espacio de soluciones está formado por ordenaciones de valores de dichas variables. Sin embargo, las técnicas metaheurísticas se pueden aplicar también a problemas de otro tipo, por ejemplo, como con variables continuas.

La lógica de estas técnicas es similar: el punto de partida es una solución (o conjunto de soluciones) que típicamente no es óptima. A partir de ella se obtienen otras parecidas, de entre las cuales se elige una que satisface algún criterio, y a partir de la cual comienza de nuevo el proceso. Este proceso se detiene cuando se cumple alguna condición establecida previamente.

## 5.1 Características

Las técnicas metaheurísticas comparten las siguientes características:

- Son ciegas, no saben si llegan a la solución óptima. Por lo tanto, se les debe indicar cuándo deben detenerse.
- Son algoritmos aproximativos, por lo tanto, no garantizan la obtención de la solución óptima.
- Aceptan ocasionalmente malos movimientos. Es decir, se trata de procesos de búsqueda en los que cada nueva solución no es necesariamente mejor. Esto, comparando en términos de la nueva aptitud de la función objetivo, respecto a la anterior. Algunas veces aceptan, incluso, soluciones no factibles como paso intermedio para acceder a nuevas regiones no exploradas.
- Son relativamente sencillas. Todo lo que se necesita es una representación adecuada del espacio de soluciones, una solución inicial (o un conjunto de ellas) y un mecanismo para explorar el campo de soluciones.
- Son generales. Prácticamente se pueden aplicar en la resolución de cualquier problema de optimización de carácter combinatorio. Sin embargo, la definición de la técnica será más o menos eficiente en la medida en que las operaciones tengan relación con el problema considerado.
- La regla de selección depende del instante del proceso y de la historia hasta ese momento. Si en dos iteraciones determinadas, la solución es la misma, la nueva solución de la siguiente iteración no tiene por qué ser necesariamente la misma. En general, no lo será.

Aunque las soluciones que ofrecen los técnicas metaheurísticas no son las óptimas, en general, ni siquiera es posible conocer la proximidad de las soluciones al óptimo. Estos algoritmos permiten estudiar problemas de gran complejidad de una manera sencilla y obtener soluciones suficientemente buenas en tiempos razonables.

A pesar de que estas técnicas son relativamente recientes, los campos de aplicación de las técnicas metaheurísticas son numerosos (electrónica, telecomunicaciones, electromagnetismo, entre otros.).



## 6 Big Bang Big Crunch

El Big Bang Big Crunch es una metaheurística basada en la teoría del origen del universo, llamada teoría del Big Bang Big Crunch, en la cual una serie de explosiones y contracciones dan origen a un equilibrio. En la fase del Big Bang, la disipación de energía produce desorden y lo randómico es la principal característica de esta fase. Luego, en la fase del Big Crunch, las partículas distribuidas de forma randómica son llevadas a un orden al converger hacia un punto denominado centro de masa [9]. El comportamiento de este algoritmo demuestra superioridad frente a algoritmos genéticos de búsqueda desarrollados por el mismo autor de esta metaheurística.

La aleatoriedad puede ser vista como el equivalente a la energía de disipación en la naturaleza mientras que la convergencia hacia un punto óptimo local o global puede ser vista como la atracción gravitacional. Dado que la disipación de energía crea desorden en partículas ordenadas, la metaheurística usa la aleatoriedad como una transformación desde una solución; generada por la convergencia, hasta la generación de nuevas soluciones candidatas (desorden o caos). Este método es similar a los algoritmos genéticos (GA) [10] en la manera randómica de crear la población inicial. Se compone de dos fases, Big Bang y Big Crunch, las cuales son explicadas a continuación.

### 6.1 Fase del Big Bang

En esta fase se crea la población inicial. Las soluciones candidatas son esparcidas por el espacio de búsqueda de manera uniforme. Debido a que la generación de números randómicos puede producir números mayores a los permitidos, es necesario limitarlos para mantenerlos dentro del espacio de búsqueda.

### 6.2 Fase del Big Crunch

El Big Crunch es un operador de convergencia que recibe muchas entradas pero tiene solo una salida. Denominada centro de masa, debido a que la única salida es el resultado de haber calculado el centro de masa. Este cálculo se realiza con la siguiente fórmula:

$$x^c = \frac{\sum_{i=1}^N \left( \frac{1}{f^i} x^i \right)}{\sum_{i=1}^N \frac{1}{f^i}} \quad (8)$$

Donde  $x^i$  es un punto dentro del espacio de búsqueda generado,  $f^i$  es aptitud (fitness) en ese punto y  $N$  es el tamaño de la población en la fase del Big Bang. Este algoritmo elimina la necesidad de hacer cálculos de dos en dos, ya que toma a la población como un todo, en vez de seleccionar dos miembros y calcular su centro de gravedad.

Posterior a la fase del Big Crunch, el algoritmo debe crear nuevos miembros para ser usados en la fase Big Bang de la próxima iteración. Lo mas fácil sería saltar al primer paso y crear nuevamente de forma aleatoria la población inicial, pero hacer esto desecharía el conocimiento ganado en las iteraciones anteriores lo que desencadenaría en una probabilidad de convergencia muy baja. Un algoritmo de optimización debe converger hacia un óptimo, pero al mismo tiempo para ser clasificado como un algoritmo global, debe contener ciertos puntos distintos dentro del espacio de búsqueda con una probabilidad de decrecer para un problema de minimización. Mas precisamente, el algoritmo debe generar un número importante de soluciones cercanas al punto óptimo pero mantener algunas soluciones esparcidas en el espacio de búsqueda después de un cierto número de pasos. El radio de soluciones lejanas al óptimo debe ir decreciendo a medida que las iteraciones aumentan, pero en ningún caso deben ser igual a 0.

El uso del conocimiento ganado en las iteraciones anteriores (centro de masa) puede verse reflejado al esparcir las nuevas soluciones alrededor del mismo. Aquí encontramos la función de movimiento de la metaheurística:

$$x^{new} = x^c + \frac{l * r}{k} \quad (9)$$

Donde  $x^c$  corresponde al centro de masa,  $l$  es el limite superior del parámetro,  $r$  es un número randómico y  $k$  es es el número de la iteración.

## 7 Resolución de MCDP con Big Bang Big Crunch

Para esta investigación, el problema MCDP es modelado con vectores y matrices que serán del tipo  $[M]$  y  $[P \times C]$ , en donde  $[M]$  representa en cada una de sus casillas la posición de una máquina en una determinada celda, y  $[P \times C]$  es una matriz de tipo pieza-celda.

BB-BC es una metaheurística que no posee variables de inicio ya que todas son calculadas y utilizadas en tiempo de ejecución, por lo que no es necesario inicializar ninguna variable. El hecho de haber inicializado la solución inicial con vectores nos permite evitar la restricción 5, ya que los índices del vector son únicos, por lo que una máquina no podrá estar en dos celdas. A continuación se procede a modelar el MCDP con la metaheurística BB-BC.

### 7.1 Big Bang

La primera fase de la metaheurística es la fase del Big Bang, en donde las soluciones iniciales son generadas de formas randómica. En la primera iteración, no se usa ningún tipo de información o conocimiento previo, pero en las siguientes iteraciones el vector máquina-pieza (nuevas soluciones candidatas) es modificado tomando en cuenta la aptitud (centro de masa) de la iteración anterior.

Vector máquina-celda

M1	M2	M3	M4	M5	M6
2	3	1	1	3	2

**Figura 7.4** Vector inicial

Tomando el vector de la figura 7.4, ese vector es una solución inicial generada totalmente de forma randómica. En la segunda iteración, luego de encontrar el centro de masa (que será descrito en la fase del Big Crunch) el algoritmo vuelve a la fase del Big Bang generando nuevas soluciones candidatas según la fórmula 9. Supongamos que en la segunda iteración la aptitud (centro de masa) es igual a 29, aplicando la fórmula el nuevo vector máquina-celda quedaría de la siguiente manera:

Vector máquina-celda

M1	M2	M3	M4	M5	M6
29.7	29.05	29.43	29.78	29.9	29.07

**Figura 7.5** Vector intermedio

Como se aprecia en la figura 7.5 la metaheurística arroja valores enteros, por lo que se debe aplicar un proceso de binarización (ver sección 7.3).

$$x^{new} = discretizacion(v4(x^{new})) \quad (10)$$

Para el ejemplo analizado, se utiliza una función de transferencia v4 de la clase V-Shaped y luego se realiza una discretización del valor obtenido para que no contenga valores entre 0 y 1, sino que entre 1 y el número de celdas. En consecuencia a lo descrito anteriormente, la nueva solución finalmente quedaría como una solución candidata válida.

Vector máquina celda					
M1	M2	M3	M4	M5	M6
1	1	3	2	2	3

**Figura 7.6** Nuevo vector candidato

El  $x^{new}$  es en realidad  $x_i^{new}$ , por lo que la fórmula 10 debe aplicarse a cada posición del vector máquina-pieza para obtener la solución candidata.

## 7.2 Big Crunch

Como se mencionó en la sección 5.1, en esta fase se calcula el centro de masa según la ecuación 8, pero opcionalmente el autor señala que la mejor aptitud de esa iteración puede ser usado como centro de masa. A lo largo de la investigación, se llevo a cabo una serie de pruebas tomando ambas opciones (ver sección 8.2). Los resultados fueron bastante similares, por lo que el investigador decide por temas de facilidad de cálculo utilizar la mejor aptitud como centro de masa, como se muestra en la ecuación 11.

$$x^c = mejorAptitud \quad (11)$$

## 7.3 Funciones de Binarización

El proceso de binarización se utiliza para transformar una solución no binaria en una binaria. Aplicar este proceso es necesario cuando las ecuaciones de movimiento de una metaheurística determinada generan una solución no-binaria al resolver un problema de dominios binarios, como es el caso del MCDP. Este problema ocurre a menudo, dado que la mayoría de las metaheurísticas han sido diseñadas para trabajar en dominios reales.

A continuación, se detallan las funciones de binarización y de transferencia existentes.

$$T(d_w^j) = \frac{1}{1 + e^{-2d_w^j}} \quad (12)$$

$$T(d_w^j) = \frac{1}{1 + e^{-d_w^j}} \quad (13)$$

$$T(d_w^j) = \frac{1}{1 + e^{-\frac{d_w^j}{2}}} \quad (14)$$

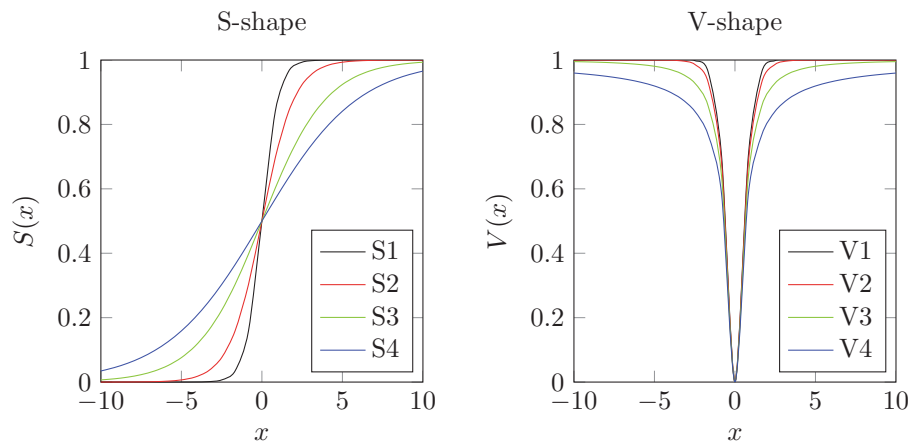
$$T(d_w^j) = \frac{1}{1 + e^{-\frac{d_w^j}{3}}} \quad (15)$$

$$T(d_w^j) = \left| \operatorname{erf} \left( \frac{\sqrt{\pi}}{2} d_w^j \right) \right| = \left| \frac{\sqrt{2}}{\pi} \int_0^{\frac{\sqrt{\pi}}{2} d_w^j} e^{-t^2} dt \right| \quad (16)$$

$$T(d_w^j) = |\tanh(d_w^j)| \quad (17)$$

$$T(d_w^j) = \left| \frac{d_w^j}{\sqrt{1 + d_w^{j2}}} \right| \quad (18)$$

$$T(d_w^j) = \left| \frac{2}{\pi} \arctan \left( \frac{\pi}{2} d_w^j \right) \right| \quad (19)$$



**Figura 7.7** Visualización de las funciones de transferencia V-Shaped y S-Shaped

## 7.4 Pseudocódigo Big Bang Big Crunch

A continuación se describe el pseudocódigo de BB-BC.

---

**Algoritmo 7.1** Big Bang Big Crunch

---

```
1: solución_inicial ← genera soluciones randomicas
2: while  $i < \text{max\_iteración}$  do
3:   Calcular centro de masa (el mejor fitness puede ser utilizado como solución inicial)
4:   Calcular nuevas soluciones candidatas( $x^{new}$ )
5:   Binarizar nuevas soluciones candidatas ( $x^{new} = \text{Discretizacion}(v4(x^{new}))$ )
6: end while
7: return mejor_solución
```

---

## 8 Experimentación

### 8.1 Resultados

El proceso de testeo del algoritmo fue llevado a cabo en un notebook con procesador Intel Core i5-3230M a 2.60GHz, 8GB de RAM y un sistema operativo Windows 7. El lenguaje utilizado es Java y el entorno de desarrollo es el IDE Eclipse.

Se han utilizado las instancias de prueba publicadas por Boctor [5] La primera configuración puesta a prueba tiene un número de celdas  $C = 2$  y un  $M_{máx} = 8, 9, 10, 11, 12$ , mientras que la segunda configuración tiene un número de celdas  $C = 3$  y un  $M_{máx} = 6, 7, 8, 9$ . Cada instancia de prueba se ejecutó 31 veces. Los resultados obtenidos se pueden apreciar en la tabla 8.1 y 8.2 para  $C = 2$ , 8.3 y 8.4 para  $C = 3$ , con un número de población igual a 100 y un número de iteraciones igual a 10000.

P	C=2											
	$M_{max} = 8$				$M_{max} = 9$				$M_{max} = 10$			
	O	BB-BC	A	RPD	O	BB-BC	A	RPD	O	BB-BC	A	RPD
B1	11	11	11	0.00	11	11	11	0.00	11	11	11	0.00
B2	7	7	7	0.00	6	6	6	0.00	4	4	4	0.00
B3	4	4	4	0.00	4	4	4	0.00	4	4	4	0.00
B4	14	14	14	0.00	13	13	13	0.00	13	13	13	0.00
B5	9	9	9	0.00	6	6	6	0.00	6	6	6	0.00
B6	5	5	5	0.00	3	3	3	0.00	3	3	3	0.00
B7	7	7	7	0.00	4	4	4	0.00	4	4	4	0.00
B8	13	13	13	0.00	10	10	10	0.00	8	8	8	0.00
B9	8	8	8	0.00	8	8	8	0.00	8	8	8	0.00
B10	8	8	8	0.00	5	5	5	0.00	5	5	5	0.00

**Tabla 8.1** Resultados Experimentales con C=2

		C=2							
P	$M_{max} = 11$				$M_{max} = 12$				
	O	BB-BC	A	RPD	O	BB-BC	A	RPD	
B1	11	11	11	0.00	11	11	11	0.00	
B2	3	3	3	0.00	3	3	3	0.00	
B3	3	3	3	0.00	1	1	1	0.00	
B4	13	13	13	0.00	13	13	13	0.00	
B5	5	5	5	0.00	4	4	4	0.00	
B6	3	3	3	0.00	2	2	2	0.00	
B7	4	4	4	0.00	4	4	4	0.00	
B8	5	5	5	0.00	5	5	5	0.00	
B9	5	5	5	0.00	5	5	5	0.00	
B10	5	5	5	0.00	5	5	5	0.00	

**Tabla 8.2** Resultados Experimentales con C=2

		C=3							
P	$M_{max} = 6$				$M_{max} = 7$				
	O	BB-BC	A	RPD	O	BB-BC	A	RPD	
B1	27	27	28.16	0.00	18	18	19.81	0.00	
B2	7	7	7,52	0.00	6	6	7,45	0.00	
B3	9	9	9,35	0.00	4	4	6,74	0.00	
B4	27	27	27	0.00	18	18	18,06	0.00	
B5	11	11	11,19	0.00	8	8	11,03	0.00	
B6	6	6	11,03	0.00	4	4	5,52	0.00	
B7	11	11	12.16	0.00	5	5	6.65	0.00	
B8	14	14	15.35	0.00	11	11	14.55	0.00	
B9	12	12	14.81	0.00	12	12	13.13	0.00	
B10	10	10	10.58	0.00	8	8	10.06	0.00	

**Tabla 8.3** Resultados Experimentales con C=3

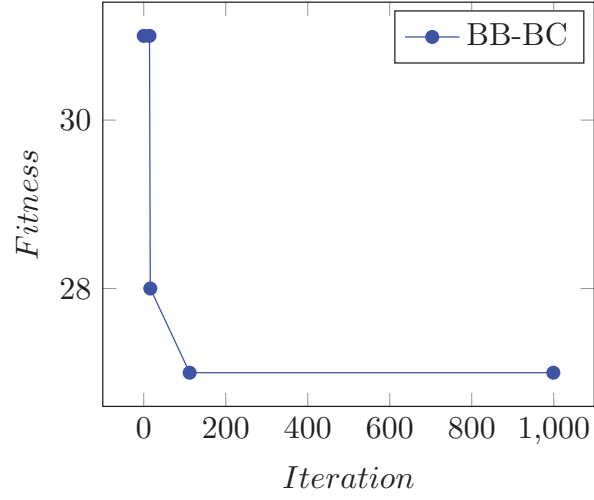


P	C=3							
	$M_{max} = 8$				$M_{max} = 9$			
	O	BB-BC	A	RPD	O	BB-BC	A	RPD
B1	11	11	14.58	0.00	11	11	13.97	0.00
B2	6	6	7.39	0.00	6	6	6.81	0.00
B3	4	4	5.61	0.00	4	4	5.68	0.00
B4	14	14	14.9	0.00	13	13	14.79	0.00
B5	8	8	9.68	0.00	6	6	8.87	0.00
B6	4	4	4.77	0.00	3	3	4.65	0.00
B7	5	5	7.26	0.00	4	4	6.45	0.00
B8	11	11	13.42	0.00	10	10	11.52	0.00
B9	8	8	11.26	0.00	8	8	8.87	0.00
B10	8	8	8.74	0.00	5	5	7.9	0.00

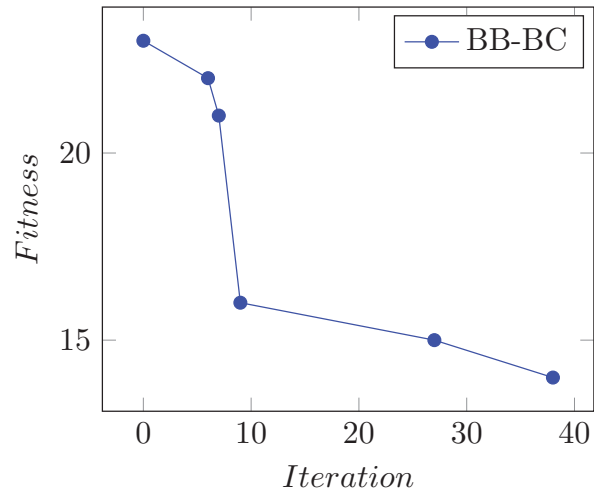
**Tabla 8.4** Resultados Experimentales con C=3

Se puede visualizar que en ambas configuraciones, C=2 y C=3, la implementación alcanzó todos los óptimos esperados. En el primer caso, las 50 instancias tuvieron un promedio igual al óptimo, es decir de las 31 veces que se iteró cada configuración, en cada una de ellas se alcanzó un promedio igual al óptimo de Boctor. Para el segundo caso, solo 1 instancia tuvo un promedio igual al óptimo, es decir. de las 31 veces que se iteró cada configuración, solo 1 instancia alcanzó el óptimo esperado las 31 veces.

Se muestra a continuación unas gráficas del algoritmo estudiado tomando en cuenta el fitness de cada iteración. Como parámetros se usaron 1000 iteraciones, una población de 100 y un Mmáx igual a 8 en el caso de C=2 y un Mmáx igual a 6 en el caso de C=3. El eje y de cada gráfico corresponde al fitness y el eje x a las iteraciones. En ambos casos se puede apreciar la convergencia del algoritmo desde el inicio, por el brusco descenso que tiene el fitness en las primeras iteraciones. En el caso de C=2, la metaheurística alcanza el óptimo propuesto por Boctor antes de las 50 iteraciones. En el caso de C=3, se alcanzó el óptimo dentro de las primeras 150 iteraciones.



**Figura 8.8** Gráfico de convergencia del problema 4 de Boctor con  $C=3$  y  $M_{\text{máx}}=6$



**Figura 8.9** Gráfico de convergencia del problema 4 de Boctor con  $C=2$  y  $M_{\text{máx}}=8$

## 8.2 Testing de funciones de binarización

También se realizó unas pruebas de las funciones de binarización para determinar si alguna de ellas mejora el desempeño de la metaheurística. Como parámetros se usaron las siguientes configuraciones:

- $C$  igual a 2 y  $Mmax$  igual a 8,9 .
- $C$  igual a 3 y  $Mmax$  igual a 6,7 .
- Un número de población igual a 100.
- Un número de iteraciones igual a 1000.
- $k$  igual a 3.

Instancia	Binarización	Óptimo Global	BB-BC FCM	Peor	BB-BC ACM	Peor
B01	s1	11	11	11	11	11
B01	s2	11	11	11	11	11
B01	s3	11	11	11	11	15
B01	s4	11	11	11	11	11
B01	v1	11	11	11	11	11
B01	v2	11	11	11	11	11
B01	v3	11	11	11	11	11
B01	v4	11	11	11	11	11
B02	s1	11	11	11	11	11
B02	s2	11	11	11	11	11
B02	s3	11	11	11	11	11
B02	s4	11	11	11	11	11
B02	v1	11	11	11	11	11
B02	v2	11	11	11	11	11
B02	v3	11	11	11	11	11
B02	v4	11	11	11	11	11

**Tabla 8.5** Testing de funciones de binarización instancias B01 y B02. BB-BC FCM, BB-BC utilizando la fórmula 8 como centro de masa. BB-BC ACM, BB-BC utilizando la aptitud como centro de masa.

Instancia	Binarización	Óptimo Global	BB-BC FCM	Peor	BB-BC ACM	Peor
b51	s1	27	30	31	28	32
b51	s2	27	28	31	30	32
b51	s3	27	31	31	29	30
b51	s4	27	29	31	30	30
b51	v1	27	28	31	28	30
b51	v2	27	29	32	29	31
b51	v3	27	30	32	28	31
b51	v4	27	30	32	28	32
b52	s1	18	23	25	23	24
b52	s2	18	22	22	22	25
b52	s3	18	21	23	18	23
b52	s4	18	22	25	23	25
b52	v1	18	20	23	19	22
b52	v2	18	21	24	19	22
b52	v3	18	19	26	22	25
b52	v4	18	21	27	19	23

**Tabla 8.6** Testing de funciones de binarización instancias B51 y B52. BB-BC FCM, BB-BC utilizando la fórmula 8 como centro de masa. BB-BC ACM, BB-BC utilizando la aptitud como centro de masa.

A través del análisis de las tablas 8.5 y 8.6 se pueden observar y determinar distintas apreciaciones. Para las instancias B01 y B02 no se puede distinguir alguna función de binarización que haya optimizado la implementación de la metaheurística, debido a que en todos los casos se alcanzó el óptimo global reportado por Boctor. Lo único que podemos observar es que la función S3 de la clase S-Shaped tuvo un desempeño más bajo ya que el peor fitness no fue igual al óptimo global. Para las instancias B51 y B52 los resultados son variados, ya si bien con el centro de masa calculado según la fórmula ninguna instancia llegó al óptimo, algunos resultados estuvieron cerca del óptimo global. En el caso de igualar el centro de masa al mejor fitness, se puede apreciar que para la instancia B52, la función S3 alcanza el óptimo global. Además, cabe recalcar que las funciones de la clase V-Shaped tuvieron un mayor tiempo de ejecución que las de la clase S-Shaped.

Por todo el análisis anteriormente detallado, se optó por implementar la metaheurística con la función S4 de la clase S-Shaped y usar el mejor fitness como CM.

## 9 Conclusión

Después de concluir la investigación, hemos podido determinar el comportamiento de la metaheurística Big Bang Big Crunch al ser implementada en la resolución del Manufacturing Cell Design Problem. Mediante un gran número de pruebas se logró visualizar que la metaheurística llegó a los óptimos esperados pero con un valor alto de población e iteraciones. Respecto a si se escoge el mejor fitness como centro de masa o se calcula según la fórmula 8, no se apreciaron cambios significativos respecto a mejores resultados o menor tiempo de ejecución, por lo que no es posible indicar con total seguridad que para este tipo de problemas es más conveniente usar una o la otra.

Para concluir, cabe mencionar que todos los objetivos definidos al comienzo de esta investigación fueron cumplidos. El investigador fue capaz de entender el MCDP así como la metaheurística BB-BC, implementar ésta última en el desarrollo del MCDP y analizar los resultados obtenidos.

Se plantea como futuros trabajos la posibilidad de implementar Autonomous Search en la metaheurística BB-BC. Si bien ésta no tiene parámetros iniciales, los resultados obtenidos variaron en función del número de población e iteraciones, por lo que sería interesante ajustar esos parámetros dinámicamente para mejorar los tiempos de ejecución del algoritmo.

## Referencias

- [1] G. K. Adil, D. Rajamani, and D. Strong. A mathematical model for cell formation considering investment and operational costs. *European journal of operational research*, 69(3):330–341, 1993.
- [2] N. Aljaber, W. Baek, and C.-L. Chen. A tabu search approach to the cell formation problem. *Computers & industrial engineering*, 32(1):169–185, 1997.
- [3] B. Almonacid, F. Aspée, R. Soto, B. Crawford, and J. Lama. Solving manufacturing cell design problem using modified binary firefly algorithm and egyptian vulture optimization algorithm. *IET Software*, 2016.
- [4] A. Atmani, R. Lashkari, and R. Caron. A mathematical programming approach to joint cell formation and operation allocation in cellular manufacturing. *The International Journal of Production Research*, 33(1):1–15, 1995.
- [5] F. F. Boctor. A jinear formulation of the machine-part cell formation problem. *The International Journal of Production Research*, 29(2):343–356, 1991.
- [6] M. Boulif and K. Atif. A new branch-&-bound-enhanced genetic algorithm for the manufacturing cell formation problem. *Computers & operations research*, 33(8):2219–2245, 2006.
- [7] J. L. Burbidge. Production flow analysis. *Production engineer*, 42(12):742–752, 1963.
- [8] O. Durán, N. Rodriguez, and L. A. Consalter. Collaborative particle swarm optimization with a data mining technique for manufacturing cell design. *Expert Systems with Applications*, 37(2):1563–1567, 2010.
- [9] O. K. Erol and I. Eksin. A new optimization method: big bang–big crunch. *Advances in Engineering Software*, 37(2):106–111, 2006.
- [10] D. Goldberg. Genetic algorithms in optimization, search and machine learning. *Addison Wesley*, 905:205–211, 1989.
- [11] T. L. James, E. C. Brown, and K. B. Keeling. A hybrid grouping genetic algorithm for the cell formation problem. *Computers & Operations Research*, 34(7):2059–2079, 2007.
- [12] A. Kusiak. The part families problem in flexible manufacturing systems. *Annals of Operations Research*, 3(6):277–300, 1985.
- [13] A. Kusiak and W. S. Chow. Efficient solving of the group technology problem. *Journal of manufacturing systems*, 6(2):117–124, 1987.

- [14] S. Lozano, B. Adenso-Diaz, I. Eguia, and L. Onieva. A one-step tabu search algorithm for manufacturing cell design. *Journal of the Operational Research Society*, 50(5):509–516, 1999.
- [15] A. L. Nsakanda, M. Diaby, and W. L. Price. Hybrid genetic approach for solving large-scale capacitated cell formation problems with multiple routings. *European Journal of Operational Research*, 171(3):1051–1070, 2006.
- [16] E. Oliva-Lopez and G. Purcheck. Load balancing for group technology planning and control. *International Journal of Machine Tool Design and Research*, 19(4):259–274, 1979.
- [17] G. F. Purcheck. A linear-programming method for the combinatorial grouping of an incomplete power set. 1975.
- [18] S. Sankaran and E. Y. Rodin. Multiple objective decision making approach to cell formation: a goal programming model. *Mathematical and Computer Modelling*, 13(9):71–81, 1990.
- [19] H. Seifoddini and C.-P. Hsu. Comparative study of similarity coefficients and clustering algorithms in cellular manufacturing. *Journal of Manufacturing Systems*, 13(2):119–127, 1994.
- [20] S. M. Shafer and D. F. Rogers. A goal programming approach to the cell formation problem. *Journal of Operations Management*, 10(1):28–43, 1991.
- [21] M. Shargal, S. Shekhar, and S. Irani. Evaluation of search algorithms and clustering efficiency measures for machine-part matrix clustering. *IIE transactions*, 27(1):43–59, 1995.
- [22] R. Soto, B. Crawford, B. Almonacid, and F. Paredes. A migrating birds optimization algorithm for machine-part cell formation problems. In *Mexican International Conference on Artificial Intelligence*, pages 270–281. Springer, 2015.
- [23] R. Soto, B. Crawford, C. Carrasco, B. Almonacid, V. Reyes, I. Araya, S. Misra, and E. Olguín. Solving manufacturing cell design problems by using a dolphin echolocation algorithm. In *International Conference on Computational Science and Its Applications*, pages 77–86. Springer, 2016.
- [24] R. Soto, B. Crawford, J. Lama, and B. Almonacid. A firefly algorithm to solve the manufacturing cell design problem. In *Information Systems and Technologies (CISTI), 2016 11th Iberian Conference on*, pages 1–7. IEEE, 2016.
- [25] R. Soto, B. Crawford, R. Olivares, M. De Conti, R. Rubio, B. Almonacid, and S. Niklander. Resolving the manufacturing cell design problem using the flower pollination algorithm. In *International Workshop on Multi-disciplinary Trends in Artificial Intelligence*, pages 184–195. Springer, 2016.

- [26] R. Soto, H. Kjellerstrand, O. Durán, B. Crawford, E. Monfroy, and F. Paredes. Cell formation in group technology using constraint programming and boolean satisfiability. *Expert Systems with Applications*, 39(13):11423–11427, 2012.
- [27] G. Srinivasan. A clustering algorithm for machine cell formation in group technology using minimum spanning trees. *The International Journal of Production Research*, 32(9):2149–2158, 1994.
- [28] V. Venugopal and T. Narendran. A genetic algorithm approach to the machine-component grouping problem with multiple objectives. *Computers & Industrial Engineering*, 22(4):469–480, 1992.
- [29] T.-H. Wu, C.-C. Chang, and S.-H. Chung. A simulated annealing algorithm for manufacturing cell formation problems. *Expert Systems with Applications*, 34(3):1609–1617, 2008.