Reducción del Tiempo de Procesamiento para una Máquina Recogido y Depósito Aplicando Sistemas de Hormigas

Por

Andrés Fernando Uribe Sánchez

Tesis sometida en cumplimiento parcial
de los requisitos para el grado de

MAESTRO EN CIENCIAS

en

Ingeniería Industrial

UNIVERSIDAD DE PUERTO RICO
RECINTO UNIVERSITARIO DE MAYAGÜEZ
2006

Aprobado por:

_____	_____
Pedro Resto Batalla, Ph.D. Miembro, Comité Graduado	Fecha
_____	_____
William Hernández Rivera, Ph.D. Miembro, Comité Graduado	Fecha
_____	_____
Sonia Bartolomei Suárez, Ph.D. Miembro, Comité Graduado	Fecha
_____	_____
Noel Artiles Léon, Ph.D. Presidente, Comité Graduado	Fecha
_____	_____
Freya Toledo, M.S. Representante, Estudios Graduados	Fecha
_____	_____
Agustín Rullán Toro, Ph.D. Director del Departamento	Fecha

Abstract

This thesis uses the main concepts of "Ant Colony" optimization to develop a heuristic to find a setup that minimizes the pick time for a Fuji IP III pick and place machine. This heuristic can be used for board assembling process at the "Model Factory" in the University of Puerto Rico at Mayagüez. Furthermore, an optimization model has been formulated to minimize the pick time of the Fuji IP III pick and place machine.

Considering the importance of parameter selection on the performance of the heuristic a methodology has been proposed to estimate these parameters such that consistent construction of good solutions is guaranteed and the negative effect of random fluctuations is diminished.

Resumen

En esta tesis de maestría se utilizan las ideas generales de los meta heurísticos de los "Sistemas de Hormigas" para desarrollar una rutina heurística que ayuda a encontrar la configuración inicial que minimice el tiempo de recogido para una máquina de recogido y depósito Fuji IP III, y que puede ser utilizada para los productos que actualmente se producen en la "Fábrica Modelo". Adicionalmente, se ha planteado un modelo de formulación matemática para la optimización del tiempo de recogido en la máquina de recogido y depósito con las características de la máquina Fuji IP III.

Considerando la importancia de la selección de los parámetros en la implementación del heurístico, se ha propuesto una metodología para la estimación de los mismos, de tal forma que se garantice la construcción consistente de buenas soluciones, tratando de minimizar el efecto negativo de las fluctuaciones aleatorias.

Para mi padre y madre, quienes me han enseñado lo es que el sacrificio por las personas que se aman.

Agradecimientos

El primero agradecimiento es para el Dr. Noel Artiles, mi consejero por estos dos años de estudio en Mayagüez, que no solamente logró extraer lo mejor de mí en el aspecto académico, retándome en todo momento, sino que se convirtió en un amigo incondicional, de aquellos que no evita elogios cuando es necesario, pero tampoco regaños cuando se amerita. Gran parte de mi decisión de continuar con el programa doctoral, se lo debo a él. Gracias por su paciencia y las múltiples oportunidades en la que metió las manos en el fuego.

Al Dr. Pedro Resto incansable trabajador, que en su contacto como profesor y consejero siempre me animó a continuar con mis metas, ahora puede estar seguro que estoy siguiendo sus recomendaciones. A la Dr. Sonia Bartolomei, que me dio la oportunidad de trabajar al lado de ella y con la que siempre sentí una motivación adicional en las clases por su experiencia y conocimiento que es capaz de transmitir. Al Dr. William Hernández, quien como jefe directo, siempre se intereso por hacer más cómoda la adaptación a Mayagüez.

Merecen especial mención el Dr. David González, quien más allá de sus excelentes virtudes como maestro siempre me hizo sentir como en mi casa, el Dr. Agustín Rullan por sus enseñanzas, y sin poder olvidar a la Profesora Mercedes Ferrer, a la cual en estos momentos considero como una amiga. A Jacqueline Orsini por sus largas charlas de nuestras diferencias, al igual que a Laura González y Mayra Colon quienes siempre me colaboraron en toda circunstancia.

De mis compañeros de Maestría todos han sido importantes en este logro. Sin embargo no puedo dejar de mencionar a Wilfredo Yushimito, eterno compañero de estudio y de conversaciones, porque siempre me enseño la luz cuando todo parecía caer, a Joel Rivera por siempre alegrar nuestros días además de sus conocimientos en cuanto programara se refiere, y a Jesús Gomez cuyas enseñanzas sobrepasan los salones de clases, amigos como ustedes siempre estarán ahí. A Lina, Liliana, Angela y Paola, amigas incondicionales conmigo, sin poder olvidar a la Rubia, por aguantar todos mis berrinches y estar siempre ahí.

A mi familia le debo la fortaleza y los ánimos para terminar con esta meta, porque aún cuando hemos pasado el año más difícil de mi vida, nunca me dejaron caer. A mi papa y mi mama, porque esto es para ustedes, a Anita, Mapis y Mauro porque me hicieron los que soy ahora.

Por último a Dayna Lee, mi amor, mi compañera, mi soporte. Para ti también es este triunfo, ahora debemos continuar juntos para la próxima meta.

Tabla de Contenido

Lista de Tablas

Tablas **Página**

Lista de Figuras

Figuras **Página**

1 Introducción

1.1 Justificación y Propósito General

La automatización de procesos se ha convertido en el pilar de gran parte de las actividades que se desarrollan en las empresas alrededor del mundo. La mejora continua de los procesos productivos que involucran no solamente labores manuales, sino las operaciones automatizadas, es la principal herramienta competitiva con la que se cuenta para hacer frente a las condiciones cambiantes de los mercados internacionales.

La primera alternativa que se debe considerar para cualquier mejora debe ir dirigida a la evaluación y posible implementación de modificaciones en los procesos de producción que hagan más eficiente el uso de los recursos disponibles.

Cuando múltiples componentes electrónicos de diferentes formas y tamaños deben ser colocados en una ubicación específica sobre una superficie de montaje, la tarea de seleccionar una rutina óptima para el posicionamiento de los mismos que minimice los tiempos de procesamiento es complicada y requiere la inversión de un tiempo considerable para la solución. Problemas con estas características han sido denominados problemas de Tecnología de Montaje sobre Superficie *(Surface Mount Technology – SMT)*, y han sido desarrollados para una amplia gama de máquinas de montaje con diferentes características tecnológicas.

Las máquinas de recogido y depósito son usadas en procesos industriales donde la precisión en la ubicación de componentes es un variable crítica y que además no pueden ser realizados fácilmente y rápidamente con operaciones manuales. Estas máquinas se usan en la producción de componentes electrónicos tales como tarjetas, tarjetas de memoria, entre muchos otros.

La mejora de productividad en estas máquinas no es un tema reciente. Tradicionalmente, éstas se han centrado en encontrar secuencias óptimas para el montaje de los múltiples componentes, de tal forma que se minimice el tiempo total de ensamblaje, partiendo de una configuración inicial. Simultáneamente se han minimizado los tiempos de preparación de las máquinas considerando el abastecimiento de los componentes y la disponibilidad de

herramientas. Estos problemas de optimización son combinatorios y las instancias más comunes son difíciles de solucionar en un tiempo razonable, por lo que se usan algoritmos heurísticos que generan buenas soluciones eficientemente a unos costos computacionalmente razonables.

Existen varios algoritmos para resolver problemas combinatorios de optimización. Entre los más recientes, está la familia de algoritmos de "Sistemas de Hormigas" (Dorigo et al., 1996), propuestos inicialmente a principios de los 90´s (Dorigo 1992). La formulación de "Algoritmos de Hormigas" ha sido resultado de la analogía del funcionamiento real de las colonias de hormigas, como modelo para la solución de problemas combinatorios de optimización utilizando la aproximación de múltiples agentes.

Como resultado de la colaboración entre compañías locales y el Decanato de Ingeniería del Recinto de Mayagüez de la Universidad de Puerto Rico, el Departamento de Ingeniería Industrial ha creado un proyecto piloto denominado Fábrica Modelo, liderado por el Dr. Pedro Resto, en la cual se producen tarjetas con circuitos impresos (*"Printed Circuit Board"*) utilizando la tecnología de SMT. En la actualidad la Fábrica Modelo ha sido subcontratada por la compañía "Electro Biology-EBI" con sede en Guaynabo – Puerto Rico, y tiene una producción mensual de tres tipos de tarjetas: 390 displays, 366 chargers y 414 drivers.

El propósito de este estudio es aplicar los algoritmos de "Sistemas de Hormigas" para encontrar buenas configuraciones iniciales para el posicionamiento de componentes y herramientas, dado un objetivo de producción en un corrida (una sola configuración), a un costo computacional razonable, para máquinas de recogido y depósito de iguales características a las utilizadas en la Fábrica Modelo, minimizando la distancia total que se debe recorrer para acceder a los componentes de una tarjeta. El implementar este algoritmo heurístico permite a los encargados de la Fábrica Modelo tener buenas opciones frente a las configuraciones que calcula la programación integrada a la máquina.

1.2 Producción de Tarjetas en la Fábrica Modelo - RUM

El proceso de producción al interior de la Fábrica Modelo está compuesto por tres estaciones en serie unidas por una única correa de alimentación, y cuenta con cuatro tipos de máquinas (Figura 1). Cada unidad de procesamiento está compuesta por seis tarjetas de iguales características, que son procesadas simultáneamente.

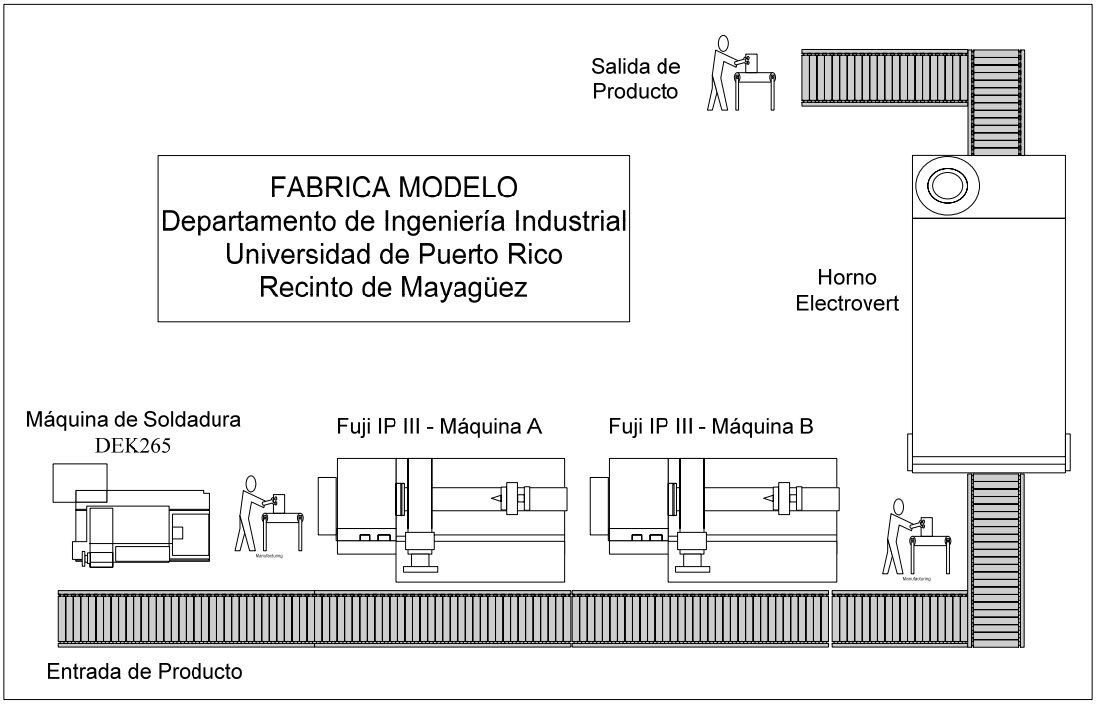


Figura 1 - Despliegue de la Fábrica Modelo

En la primera estación se encuentra una máquina DEK265, que deposita pasta de soldadura a las seis tarjetas de cada unidad de acuerdo a las especificaciones del producto que se va a fabricar. La pasta sobre la cual se instalaran directamente los componentes, debe ser aplicada siguiendo una configuración específica de acuerdo al producto en proceso. Con el propósito de verificar que este proceso haya sido exitoso, una máquina SENTRY realiza una comparación utilizando fotos digitales de la unidad actual, frente un modelo "ideal" de producción (el espesor y la ubicación de la pasta son los parámetros que se comparan). En caso de presentarse alguna diferencia, el proceso se detiene y se activa una señal luminosa en la parte superior de la máquina para indicarlo.

La segunda estación esta compuesta por dos máquinas FUJI IP III de recogido y depósito, que tienen la función de instalar sobre cada unidad los componentes electrónicos requeridos para el producto en proceso. Este proceso de instalación se realiza en serie, es decir, una vez que la primera máquina finaliza la instalación de un grupo de componentes sobre una unidad, ésta es transportada por la correa alimentadora a la segunda máquina para instalar los componentes restantes. Cada vez que se completa la instalación de un componente, se verifica el éxito de la acción utilizando un mecanismo de fotos digitales, similar al descrito previamente. Al igual que en la primera estación, al encontrarse con una diferencia entre el resultado y las especificaciones del producto, se enciende una señal luminosa informando el problema y deteniendo la producción. La Figura 2 presenta una de las máquinas Fuji IP III.

Figura 2 - Máquina de recogido y depósito Fuji IP III

Las unidades que han sido completadas exitosamente con todos los componentes, continúan por la correa de alimentación hasta la última estación, compuesta por un horno Electrovert, el cual fija los componentes a las tarjetas. Al salir del horno, las unidades son inspeccionadas, para verificar la integridad del producto y el cumplimiento con las especificaciones. Es importante señalar que entre cada estación se realiza una verificación visual del producto por parte del operador de la estación.

1.3 Optimización en la tecnología de SMT

En el proceso de ensamblaje de un tablero de circuitos impresos *(printed circuit board - PCB)*, perteneciente al ya nombrado grupo de SMT, que requiere la instalación de múltiples componentes electrónicos con diferentes tamaños y formas en ubicaciones especificas, el determinar una óptima configuración inicial para los carretes de los componentes y una buena ruta de procesamiento de las tareas, disminuye significativamente los tiempos de procesamiento y correspondientemente representa un aumento en la capacidad de producción y la eficiencia de los procesos. Además, en los casos en los cuales se dispone de más de una máquina, la asignación de componentes a las mismas, es el punto de partida para la búsqueda de una secuencia óptima de instalación.

La optimización en estos procesos de ensamblaje involucra el análisis de tres subproblemas asociados:

- Asignación de componentes a ser instalados en cada una de las máquinas de recogido y depósito (En el caso que se dispongan de dos o más máquinas).
- Determinación del número de carretes a utilizarse por tipo de componente, y la asignación de los mismos a las ranuras de almacenaje en cada máquina (atado a un número requerido de unidades a producir con una misma configuración).
- Diseño de una secuencia óptima para la instalación de los componentes dada una configuración previa.

Dadas sus características combinatorias, estos problemas han sido clasificados como NP-hard, lo que implica que el solucionarlos óptimamente en un tiempo razonable es una tarea compleja, haciendo atractiva la utilización de “buenas” soluciones heurísticas rápidamente y con un menor costo computacional.

1.4 Objetivos

- Revisar publicaciones en las cuales se aborde el problema de asignación y secuenciación de componentes en máquinas de recogido y depósito con características similares a las utilizadas en la Fábrica Modelo para la fabricación de tableros de circuitos impresos.
- Revisar las ideas fundamentales de los heurísticos de los "Sistemas de Hormigas" y algunas de las aplicaciones a problemas de carácter combinatorio.
- Describir detalladamente el funcionamiento y operación de la máquina de recogido y depósito Fuji IP III que se utiliza en la Fábrica Modelo de la UPRM.
- Desarrollar una formulación matemática para la asignación de carretes de componentes y herramientas de instalación, para la fabricación de tableros de circuitos impresos, que refleje el funcionamiento de las máquinas de recogido y depósito Fuji IP III que son utilizadas en la Fábrica Modelo.
- Implementar los "Sistema de Hormigas" para la formulación matemática desarrollada con el propósito de plantear un nuevo heurístico para el problema en cuestión.
- Determinar los parámetros asociados a los "Sistemas de Hormigas" que inciden en el comportamiento del heurístico en la búsqueda de buenas soluciones al problema combinatorio propuesto, y encontrar la mejor combinación de estos.
- Resolver el modelo de optimización propuesto con un software especializado de optimización, y hacer una comparación con los resultados obtenidos por el heurístico propuesto de "Sistemas de Hormigas".
- Presentar a los encargados de la Fábrica Modelo el funcionamiento del heurístico desarrollado, para que pueda ser utilizado en la planeación de producción.
- Despertar el interés por una investigación futura de la implementación de los "Sistemas de Hormigas" al problema de secuenciar las operaciones de recogido y depósito en máquinas con características similares a las estudiadas.

1.5 Organización

Posterior a está breve introducción, en el capítulo 2 se presenta lo más relevante de la revisión de literatura asociada a SMT y de los "Sistemas de Hormigas". Inicialmente se señala una clasificación general de las máquinas de la tecnología de SMT con el propósito de mostrar que las particularidades de cada una afectan el modo de abordar el problema de optimización en estos tipos de procesos. Se explica las ideas fundamentales en cuanto a la optimización de SMT y se define el problema de minimización del tiempo de recogido para una máquina de instalación dual, tomado como referencia para el desarrollo de la formulación propuesta en este documento. Por último, se explica el origen y fundamento de la familia de meta heurísticos de los "Sistemas de Hormigas" y se plantea la rutina generada con dicha metodología para la solución del problema del agente viajero.

En el capítulo 3, se hace una descripción detallada de la Máquina Fuji IP III y se plantea una formulación matemática que representa el problema de minimización del tiempo de recogido. Un pequeño ejemplo numérico es presentado con el propósito de hacer clara la formulación y para mostrar el tiempo de ejecución asociado a su resolución óptima. Para completar el capítulo, se explica el heurístico desarrollado siguiendo la metodología de los "Sistemas de Hormigas" para encontrar buenas soluciones al problema de minimización del tiempo de recogido para una máquina Fuji IP III.

En el último capítulo, se presenta los resultados de un diseño de experimentos realizado para determinar el valor de los parámetros que deben ser utilizados para encontrar soluciones consistentemente "buenas" en la solución de instancias similares a las tres tarjetas que son producidas en la actualidad de la "Fábrica Modelo". Una vez encontrados dichos parámetros, se presentan las mejores configuraciones encontradas para los tres productos en cuestión. En la parte final se presentan las conclusiones.

En los apéndices se presentan los programas desarrollas en Matlab para completar las rutinas propuestas y los archivos generados en Lingo para la solución del ejemplo pequeño.

2 Revisión de Literatura

En el presente Capítulo se presenta una descripción general de las ideas más relevantes después de realizada la revisión de literatura pertinente.

Inicialmente se describe la tecnología de SMT, mencionando una clasificación de equipos de acuerdo a sus características físicas y de operación, al igual que algunas de las herramientas utilizadas para la optimización de sus operaciones.

La segunda parte presenta la aproximación tradicional al problema de optimización en la tecnología de SMT, de donde nace la idea de agrupar dos de estos problemas en uno solo, denominado el problema de minimización del tiempo de recogido. Una formulación de este problema para una Máquina Dual es descrita en la tercera sección.

Por último se presentan generalidades asociadas a la teoría de los "Sistemas de Hormigas", con el propósito de mostrar los fundamentos que son utilizados para la implementación realizada en este proyecto de tesis.

2.1 Descripción de algunas Máquinas de SMT

En el mercado existen múltiples tipos de máquinas asociadas a la tecnología de SMT. Ayob y otros (Ayob et al., 1990) presentan un vistazo general a los avances en el campo de acuerdo a la clasificación de la máquina. Mencionan algunos de los modelos que han sido propuestos para cada caso, con variaciones particulares.

En el trabajo mencionado, se proponen cinco categorías de máquinas de SMT de acuerdo a sus especificaciones y los métodos operacionales. Las máquinas clasificadas tienen ranuras de almacenaje para la ubicación de los carretes de alimentación (contienen los componentes), una mesa de trabajo sobre la cual se ubica la unidad a ser ensamblada, un brazo de instalación acompañado de una cabeza que ejecuta las operaciones de recogido (*"Pick"*) y de depósito (*"Place"*), y un almacenaje de herramientas donde se encuentran las boquillas (*"Nozzles"*) de instalación.

Los carretes de alimentación son dispositivos en los cuales se ubican cintas de un componente en específico. Estos carretes de alimentación son fijados físicamente en la

máquina en las ranuras de almacenaje, que bajo algunas especificaciones pueden desplazarse en una dirección, haciendo más fácil el acceso a los mismos. Generalmente, el brazo de instalación tiene en su extremo una cabeza, y se mueve hasta un punto específico para hacer la operación de recogido y se desplaza a otro para la operación de depósito; su movimiento puede ser restringido o no en una única dirección. Algunas de las variaciones entre máquinas por ejemplo, incluyen la existencia de más de un brazo, o un único brazo con dos o más cabezas de instalación.

La mesa de trabajo puede tener movimiento en una o dos direcciones. Las boquillas son herramientas que se fijan en las cabezas de instalación y pueden ser utilizados para la instalación de un único o varios tipos de componentes.

2.1.1 Máquina de Instalación Dual

Típicamente, la mesa de trabajo puede desplazarse en las direcciones de X y Y, para alinearse bajo la cabeza del brazo de instalación y realizar la tarea de depósito. Existe un único brazo con movimiento sobre el eje Y, dotado con una cabeza de instalación en cada uno de sus extremos (2 cabezas en total). Las ranuras de almacenamiento están divididas en dos grupos, arriba y abajo de la mesa de trabajo, de tal forma que cada cabeza de instalación solamente accede a único grupo de carretes. Cada operación de recogido y depósito es alternada entre las dos cabezas, mientras una de ellas esta realizando una operación de depósito, la otra esta realizando un recogido desde las ranuras asociadas.

Se menciona el desarrollo de un algoritmo “Adaptive Simulated Annealing” para solucionar el problema de optimización de SMT. En cada iteración del algoritmo se genera una solución candidata y se verifica su factibilidad.

2.1.2 Máquinas de Instalación de Múltiples Estaciones

Esta máquina esta compuesta por varias estaciones de instalación, con características mecánicas idénticas y capaces de trabajar concurrentemente. Las estaciones están conectadas por una cadena transportadora. Los procesos de instalación son realizados simultáneamente por todas las estaciones sobre cada pieza en la línea y una vez finalizado se pone en

movimiento la cadena. La revisión de literatura presenta una solución enfocada en dos problemas: el asignar componentes a cada estación y el arreglo (configuración y secuenciación) de componentes al interior de cada una, con el propósito de maximizar la producción al minimizar el número de movimientos en de las cabezas.

2.1.3 Máquina de Instalación Estilo Torre

Presentan un mecanismo similar al de un carrusel con múltiples cabezas de instalación que giran sobre el mismo eje, y realizan la operación de recogido en una misma zona. En este caso, las ranuras de almacenamiento se mueven hasta el punto fijo para que se realice un recogido, simultáneamente la mesa de trabajo se mueve hasta una ubicación especifica para que la cabeza que previamente había recogido un componente, lo instale. El tiempo de instalación es dado por los componentes con menor tasa de rotación en el mecanismo, combinado con el problema de instalación de las boquillas para cada componente.

Una aproximación mencionada para solucionarlo es la del Problema del Agente Viajero – TSP, donde el objetivo es el minimizar el tiempo promedio en la secuencia de viaje, pero sujeto a que las distancias entre los nodos del problema son otras variables del mismo.

2.1.4 Máquina de Instalación Múltiple con una Cabeza

A diferencia de los modelos anteriores, cada vez que la cabeza del brazo de instalación realiza un recogido, toma más de un solo componente, y consecuentemente hace la instalación secuencial de varios. Después de realizar la instalación de todos los componentes que utilizan una misma boquilla, se hace el cambio de éste para comenzar con otra nueva secuencia. El brazo y la cabeza tienen movimientos independientes sobre los ejes X y Y. Básicamente se menciona el análisis de dos circunstancias: la primera en la que no es necesaria la utilización de otros carretes, y otra en la que es necesario el cambio ellos.

2.1.5 Máquina de Instalación Secuencial

Típicamente estas maquinas tiene la cabeza de instalación ubicada en el extremo de un brazo. Mientras que el brazo tiene un movimiento en el sentido X, la cabeza lo hace en el Y.

La secuencia comienza al instalarse en la cabeza al boquilla adecuada, ir a los carretes de almacenamiento, hacer la operación de recogido y moverse hasta un punto fijo en la tabla y ejecutar un depósito. Ahmadi y Mamer (1998), modelan el problema de secuenciar los tipos de parte por instalación y la secuencia de movimientos entre puntos en la mesa de trabajo, como una colección interdependiente de Problemas del Agente Viajero.

2.2 Optimización de Máquinas Recogido y Depósito

A continuación se describen algunas metodologías que reflejan la forma tradicional con la cual se ha abordado el tema de optimización para las máquinas de tecnología SMT, y de las cuales se extrajeron ideas para formular el problema de minimizar el tiempo de las operaciones de recogido de los componentes necesarios para el ensamblaje de una tarjeta.

Típicamente los problemas de SMT son divididos en tres subproblemas:

- Asignación de tipos de componentes y cantidad de carretes a cada máquina disponible.
- Asignación de las cintas de componentes a los carretes de almacenamiento.
- Secuenciación de la instalación de los componentes.

Aún cuando podrían ser considerados como problemas independientes, la secuenciación de las instalaciones pasa a depender de la ubicación en los carretes de almacenamiento de cada tipo de componente. Cuando la asignación de ranuras a los carretes no es la adecuada, aún cuando la secuencia de instalación sea la óptima, la velocidad del sistema no será la mejor posible.

Altinkemer y otros (Altinkemer et al., 2000) plantean un modelo matemático que considera simultáneamente los tres subproblemas enumerados previamente. Es importante señalar supuestos claves que son utilizados por los autores al momento de plantear el modelo.

- La cabeza de instalación completa un viaje volviendo a los carretes de almacenamiento, antes de dirigirse al almacén de herramientas de instalación para el cambio de la boquilla.

- Todos los componentes de un tipo de componentes son asignados al mismo carrete de almacenamiento.
- El viaje entre dos coordenadas es aproximadamente lineal a la distancia. La distancia es calculada como la distancia de Chebychev (mayor distancia en movimientos).

En otro enfoque propuesto, Ahmadi y Mamer (1999) describe un algoritmo heurístico que tiene como objetivo el alcanzar una secuencia óptima para que una máquina de recogido y depósito dual ubique múltiples tipos de componentes en una tarjeta, durante un proceso de ensamblaje. Ellos desarrollan un sistema de control para una producción diaria aproximada de 12,000 tarjetas. El propósito principal es aumentar las tasas de producción, disminuyendo los tiempos de procesamiento en los equipos automatizados.

Los autores dividen el tiempo total de procesamiento, en dos tiempos independientes: un tiempo de carga de materia prima y/o *"setup"* previo a la actividad productiva y un tiempo de instalación durante la actividad productiva.

El tiempo de carga de materia prima fue abordado inicialmente por Ahmadi, Grotzinger y Johnson (1990), utilizando un modelo analítico de operaciones en el cual se realiza la asignación de los componentes a los alimentadores, de los alimentadores a las cintas y de las cintas a los transportadores. El propósito de este modelo es el minimizar el tiempo de preparación y de recarga de componentes.

R. Amahadi (1998) presenta un algoritmo heurístico para ordenar la disposición de la materia prima, y el movimiento del mecanismo con el objetivo de minimizar los tiempos de ensamblaje de la tarjeta. La aproximación desarrollada parte de solucionar una colección de problemas TSP independientes con diferentes categorías de ciudades. El primero de los problemas plantea el ordenar óptimamente los tipos de partes para que estos sean seleccionados por la máquina, mientras que el segundo se centra en la programación de los movimientos del mecanismo para seleccionar y colocar los tipos de componentes.

En cuanto al tiempo de instalación se refiere, este es segmentado en dos actividades, la primera correspondiente al tiempo de movimiento del mecanismo de instalación desde la

cinta de alimentación hasta el lugar exacto en la tarjeta y volver a la siguiente cinta de alimentación, y un tiempo de instalación o colocación del componente sobre la tarjeta (una vez el mecanismo de instalación esta en el punto indicado) el cual es constante y es definido por las características de la máquina a estudiar.

2.3 Minimización del Tiempo de Recogido para Máquina Dual

La minimización del tiempo de recogido, comprende los dos problemas iniciales en la optimización de la tecnología de SMT, es decir el problema de "Asignación de Componentes" y el problema de la "Partición", tal y como son definidos por Ahmadi (Ahmadi et al., 1990). Sin embargo la formulación propuesta en dicho documento está concebida para una máquina dual de recogido y depósito y solamente puede ser tomada como un punto de referencia para desarrollar un modelo para la máquina Fuji IP III. A continuación se presenta una descripción de las formulaciones de cada uno de estos problemas.

2.3.1 El problema de Asignación de Componentes

Este el primer asunto que debe ser considerado en lo concerniente a maximizar la producción al reducir los tiempos de proceso. El propósito de este problema es el determinar el número de carretes de cada tipo de componente que deben ser instalados, de tal forma que se maximice el número de tarjetas que pueden ser procesadas con una sola configuración, reduciendo simultáneamente la demanda de operaciones de acarreo de materiales.

Ahmadi, Grotzinger y Johnson (1990) presenta una formulación de programación entera para la maximización del número de tarjetas completadas, con una única asignación de carretes sin reemplazo. Se presentan dos escenarios. El primero plantea la maximización del número de tarjetas producidas como función del total de componentes disponibles de cada tipo. Mientras que en el segundo se agrega la restricción que el número de componentes que son alcanzados para completar una unidad desde cierta ubicación es constante, es decir, en cada ensamblaje se sigue la misma secuencia. Este último caso refleja mejor la realidad, donde la máquina toma la pieza *n* (sin importar el tipo) de una tarjeta siempre desde la misma ranura.

Dadas las características de las máquinas Fuji IP III, conviene hacer una explicación del modelo propuesto para el segundo escenario.

Definiciones relevantes del modelo propuesto para el segundo escenario son:

- b_i = Número de componentes de tipo *i* requeridos en una tarjeta.
- v_i = Número de componentes de tipo *i* que pueden ser almacenados en un carrete.
- w_i = Número de ranuras estándar que son utilizadas por un carrete con componentes del tipo *i*.
- L = Número total de ranuras estándar disponibles en la máquina (deber ser un número par).
- A_i = Conjunto de los posibles números de carretes que podrían ser asignados del componente *i*.
- d_i = Número de elementos del conjunto A_i.
- a_{ik} = Número de carretes con componentes de tipo *i* que deben ser asignados y que son señalados por la componente *k* del conjunto A_i.
- R = Número de distintos tipos de componentes que son usados para completar una tarjeta.
- R = {1, 2, 3, ……..R}.

Con las cuales se pueden generar las siguientes expresiones:

- $u_{ik} = \text{límite superior}\left(\frac{b_i}{a_{ik}}\right)$, número de componentes que son tomados desde cada carrete tipo *i* para completar una tarjeta, si se asignan a_{ik} carretes para dicho tipo.
- $n_{ik} = \text{límite inferior}\left(\frac{v_i}{u_{ik}}\right)$, número de tarjetas que pueden ser procesadas de acuerdo a la restricción de capacidad del componente *i* al cual se le han asignado un total de a_{ik} carretes.

Mientras que las variables de decisión del modelo son definidas como:

- $y_{ik} = \begin{cases} 1 & \text{Si } a_{ik} \text{ carretes del componente } i \text{ son asignados.} \\ 0 & \text{De lo contrario.} \end{cases}$

Como proceso de inicialización, se deben construir los vectores $A_i = \{a_{1,i}, a_{i,2},\ldots,a_{i,di}\}$ y consecuentemente los vectores de los posibles números de tarjetas que puedan ser completadas $N_i = \{n_{1,i}, n_{i,2},\ldots,n_{i,di}\}$. Si la componente k del vector A_i es igual a un valor c, significa que una posible solución del número de carretes que se podrían instalar del componente i es igual a c. El valor de c debe ser menor o igual y divisor de b_i, de tal forma se garantiza que desde cada uno de los carretes se extraiga al menos un componente, y que este número sea el mismo para todos.

El modelo tiene como función objetivo (Ec. 1) el maximizar el número de tarjetas que pueden ser ensambladas. Sujeto a tres restricciones, la primera (Ec. 2) indica que el número de tarjetas ensambladas es el menor número del conjunto del total que podrían completarse dada la cantidad disponible de cada tipo de componente, la segunda (Ec. 3) se asocia con la restricción física del número de ranuras que pueden ser instaladas, y la tercera (Ec. 4) garantiza que se asigna un único número de carretes por cada tipo de componente.

$$Max \;\; n \qquad \text{Ec. 1}$$

Sujeto a:

$$n \leq \sum_{k=1}^{d_i} n_{ik} y_{ik} \quad \forall i \in R \qquad \text{Ec. 2}$$

$$\sum_{i=1}^{R} w_i \left(\sum_{k=1}^{d_i} a_{ik} y_{ik} \right) \leq L \qquad \text{Ec. 3}$$

$$\sum_{k=1}^{d_i} y_{ik} = 1 \quad \forall i \in R \qquad \text{Ec. 4}$$

Una consideración que nace al revisar el modelo propuesto se asocia con el acotamiento del conjunto de soluciones. La restricción que se activa al momento de encontrar una solución óptima es la Ec. 3 y se asocia a la limitación física de espacio. Aún cuando este

modelo encuentra el valor óptimo y los resultados que sean obtenidos sirven como parámetros de entrada para el segundo problema, podría ser poco práctico el encontrar una solución utilizando una rutina de optimización. Una buena aproximación al número máximo de tarjetas, puede ser encontrada con una rutina iterativa (asignar al menos un carrete de cada tipo y aumentar el número en los tipos que limiten la producción, mientras se tenga capacidad).

Otra aproximación sería el incluir este problema de maximización como una restricción en el problema de partición. De tal forma que el número de carretes que se requieren de cada componente sean variables de decisión en este último, y lo que se haga es el fijar un número mínimo de tarjetas a ser completadas con dicha configuración.

Además, aún cuando el planteamiento propuesto aplica a una máquina que tiene un total de dos bancos de almacenamiento (máquina dual, tiene dos grupos de ranuras de almacenamiento a cada lado de la máquina), se plantea la restricción de espacio como un todo, sin considerar la posibilidad que alguna solución pueda utilizar el total de ranuras, pero que no puedan ser acomodadas en los dos bancos dadas las distancias. Por ejemplo, si se tuviera un total de 10 ranuras estándar y tres tipos de productos (w_1=3, w_2=3, w_3=4), una solución factible de acuerdo a la formulación sería un carrete de cada componente, y aún cuando no sobrepasan el total de ranuras disponibles, no se pueden organizar en dos grupos con 5 ranuras en cada uno.

2.3.2 El Problema de Partición

Una vez que se ha determinado el número de carretes a_i de cada tipo de componente que deben ser utilizados para maximizar el número de tarjetas, se debe tomar la decisión de determinar cuales ranuras serán asignadas a cada uno de los carretes, de tal forma que se minimice el tiempo total empleado para acceder a todos los componentes requeridos en una tarjeta (sobre un punto de referencia en cada banco de almacenamiento), con lo cual se termina minimizando el tiempo de recogido.

Ahmadi, Grotzinger y Johnson (1990) presentan un modelo de programación entera en para minimizar el tiempo muerto causado por el desequilibrio en la asignación de carretes y por la utilización innecesaria de cambios de boquillas, ambas causantes del aumento en el tiempo de recogido.

Algunas consideraciones importantes con respecto a este problema son:

- Si el número de componentes (sin importar el tipo) que son tomados desde cada lado de la máquina son iguales, el proceso de instalación se puede definir como concurrente. Es decir, mientras uno de los brazos ejecuta un depósito, el otro puede estar haciendo un recogido (operación en paralelo), no obstante la presencia de un desequilibrio implica que algunas de las operaciones se deban realizar en serie, introduciendo tiempo muerto de operación para alguno de los brazos. Si se quiere hacer una eficiente asignación de secuencias de instalación (tercer problema en la optimización de tecnología de SMT), se debe buscar un balance en la asignación de las ranuras que conlleve a un balance en el número de componentes que se extraen desde cada lado.
- Una buena solución debe incorporar pocos cambios de boquillas. Dadas las características físicas de cada tipo de componente las boquillas de instalación no son estándar y por tanto se debe incluir en la formulación dicha variabilidad. Adicionalmente se debe tener en cuenta la restricción del número de boquillas que pueden ser instalados en cada lado de la máquina.

Para explicar la formulación es necesario definir las variables siguientes:

- b_i = Número de componentes de tipo *i* requeridos en una tarjeta
- a_i = Número de carretes con componentes de tipo *i* a ser instalados.
- Se utiliza el valor conocido de b_i para hacer la siguiente definición, $b_i = q_i a_i + r_i$ donde $0 \le r_i \le a_i$.

- l_i = Número de carretes de componentes tipo *i* que son asignados al lado izquierdo, donde se cumple que $0 \le l_i \le a_i, \ \forall \ i$.

- O_i = Número de componentes de tipo *i* usados desde el lado izquierdo para completar una tarjeta, después que $q_i l_i$ componentes han sido usados, donde se cumple que $0 \le O_i \le l_i, \ \forall \ i$.

- $\lambda_i = q_i l_i + O_i$, corresponde al número de componentes de tipo *i* que son tomados desde el lado izquierdo para completar una tarjeta. El primer valor corresponde al número de componentes de tipo *i* que han sido usados en la tarjeta y el segundo al número de componentes del mismo tipo que faltan para completarla.

- El número de componentes utilizados desde el lado derecho para completar una tarjeta es puede definir de la forma $\rho_i = q_i(a_i - l_i) + (r_i - O_i)$, donde el primer término corresponde al número de componentes que han sido usados desde el lado derecho y el segundo al número de componentes que faltan por instalarse desde el mismo lado para completar una tarjeta.

- I = Tiempo en segundos para que el conjunto brazo-cabeza se mueva entre dos puntos fijos mas el tiempo en segundos del movimiento vertical de las cabezas de instalación. Se asocia al tiempo muerto causado por el desequilibrio en el número de componentes que se extraen desde cada lado.

- Γ = Tiempo empleado para el cambio de una boquilla y corresponde al tiempo muerto causado por los cambios de herramientas.

- $e^t = \{1,1,.....,1\}$.

- t = número total de boquillas que pueden ser acomodados a cada lado de la máquina.

- Se debe crear una matriz de información C, donde la componente (i,j) es:

$$c_{ij} = \begin{cases} 1 & \text{Si el componente tipo } i \text{ puede ser acomodado con un nozzle tipo j.} \\ 0 & \text{De lo contrario.} \end{cases}$$

Se define además la matriz de decisión T, donde la componente (j,c) se define como:

$$t_{jc} = \begin{cases} 1 & \text{Si el nozzle } j \text{ esta disponible en el lado } c \ (c = 1 \text{ izquierda, } c = 2 \text{ derecha}). \\ 0 & \text{De lo contrario.} \end{cases}$$

La formulación propuesta por Ahmadi, Grotzinger y Johnson (1990) es:

$$Min \ \ \mathrm{I}\left|\sum_{i=1}^{\mathrm{R}}(\lambda_i - \rho_i)\right| + \Gamma e^t (T_{.1} + T_{.2}) \qquad \text{Ec. 5}$$

Sujeto a:

$$\lambda_i = q_i l_i + O_i \quad \forall i \in R \qquad \text{Ec. 6}$$

$$\rho_i = q_i (a_i - l_i) + (r_i - O_i) \quad \forall i \in R \qquad \text{Ec. 7}$$

$$\lambda_i \le b_i (C_{i.} T_{.2}) \quad \forall i \in R \qquad \text{Ec. 8}$$

$$\rho_i \le b_i (C_{i.} T_{.1}) \quad \forall i \in R \qquad \text{Ec. 9}$$

$$\sum_{i=1}^{\mathrm{R}} w_i l_i \le \frac{W}{2} \qquad \text{Ec. 10}$$

$$\sum_{i=1}^{\mathrm{R}} w_i (a_i - l_i) \le \frac{W}{2} \qquad \text{Ec. 11}$$

$$e^t T_{.1} \le t \qquad \text{Ec. 12}$$

$$e^t T_{.2} \le t \qquad \text{Ec. 13}$$

$$0 \le O_i \le l_i \le a_i \quad \forall i \in R \qquad \text{Ec. 14}$$

$$O_i \le r_i \quad \forall i \in R \qquad \text{Ec. 15}$$

$$l_i - O_i \le a_i - r_i \quad \forall i \in R \qquad \text{Ec. 16}$$

$$l_i, O_i \in Z_+ \quad \forall i \in R \qquad \text{Ec. 17}$$

$$t_{j,c} \in \{0,1\} \quad \forall j, c \qquad \text{Ec. 18}$$

Es importante señalar que en el documento mencionado, aún cuando se utiliza el conjunto de variables q_i, no se le da una interpretación.

La función objetivo (Ec. 5) se compone de dos términos, el primero asociado al tiempo muerto causado por el desequilibrio, mientras que el segundo corresponde al tiempo muerto por los cambios de boquillas. Las ecuaciones 6 y 7, sirven para definir el total de componentes de cada tipo que son extraídos desde cada lado. Las ecuaciones 8 y 9 fijan un límite al número de componentes que son extraídos desde cada lado y asocian la existencia de una herramienta de instalación que pueda ser utilizada. Las ecuaciones 10 y 11 corresponden a las limitaciones de espacio disponible en cada lado de la máquina. La restricción del número de boquillas que pueden ser ubicados en cada banco de almacenamiento se refleja en las restricciones 12 y 13. Las restantes restricciones corresponden a la definición de variables anteriormente definidas.

El modelo está concebido bajo la idea de equilibrar el número de componentes que son extraídos desde cada banco de almacenamiento y básicamente asigna cual debe ser el lado en el cual se deben ubicar. No obstante, aún cuando las restricciones en términos generales reflejan las condiciones de funcionamiento de las máquinas Fuji IP III, no da la posibilidad de determinar el orden que debe tener cada carrete con el propósito de minimizar el tiempo total de instalación.

2.4 Los Sistemas de Hormigas

2.4.1 Generalidades de los Sistemas de Hormigas

Los algoritmos de "Sistemas de Hormigas" (Dorigo et al., 1996), son una aproximación para la solución de problemas combinatorios de optimización. El propósito no es el simular el comportamiento de estos sistemas reales, sino el de usar "colonias artificiales de hormigas" como una herramienta de optimización.

La idea de los "Sistemas de Hormigas" nace de la observación del comportamiento de las colonias de hormigas reales y las relaciones de cooperación existentes entre sus integrantes. ¿Cómo es posible que animales prácticamente ciegos como las hormigas, sean capaces de encontrar los caminos más cortos desde sus colonias a los sitios de comida y regresar al punto de salida? La respuesta a esta pregunta fundamenta la teoría descrita a continuación.

Deneubourg (1983 y 1989) resume algunas investigaciones biológicas en las cuales se han encontrado que las hormigas reales individualmente diseminan un rastro de feromonas sobre el camino que recorren. De esta manera comunican información para que las otras decidan qué camino tomar. Mientras que una hormiga aisladamente sigue un movimiento completamente aleatorio, una hormiga que se encuentra con el rastro que otra ha dejado, puede detectarlo y decidir seguirlo con una alta probabilidad. Este proceso ha sido denominado como comportamiento autocatalíctico o retroalimentación positiva. De este modo la probabilidad que una hormiga escoja un determinado camino, aumenta de acuerdo al número de hormigas que previamente lo han escogido. La Figura 2 es útil para ilustrar este comportamiento.

El problema es encontrar la ruta más corta desde un punto A donde se encuentra ubicada la colonia de hormigas, hasta una fuente de alimento ubicada en el punto E, y volver a la colonia. Claramente en la ausencia de obstáculos, representado en la Figura 2a, el camino más corto es una línea recta entre los dos puntos, no obstante al presentarse un obstáculo, las hormigas deben encontrar el camino más corto para rodearlo y retomar su viaje al destino final.

Con la presencia de un obstáculo, el camino más corto entre el punto A y el E, está dado por la ruta ACE. La primera hormiga que se mueve en línea recta desde A hasta E, al encontrarse con el obstáculo en el punto B, tendrá la misma posibilidad de elegir cualesquiera de los dos caminos (BCD o BHD) dada la ausencia de rastros previos, tal y como se presenta en la Figura 2b. La hormiga que escoja bordear el camino BCD llegará más pronto al punto D y correspondientemente al E, la hormiga regresará por la ruta ED, y al momento de encontrarse con el obstáculo nuevamente, encontrará un fuerte rastro en la dirección DCB, generado por la mitad de las hormigas que han seleccionado el camino DCBA y por las hormigas que previamente han tomado el camino BCDE (el camino es de longitud menor, haciendo que el rastro se acumule en mayor proporción). En la Figura 2c se muestra que el aumento en el número de hormigas que por unidad de tiempo recorren el camino BCD será mayor que las que recorren BHD, aumentando de esta forma el rastro de feromonas por el camino mas corto.

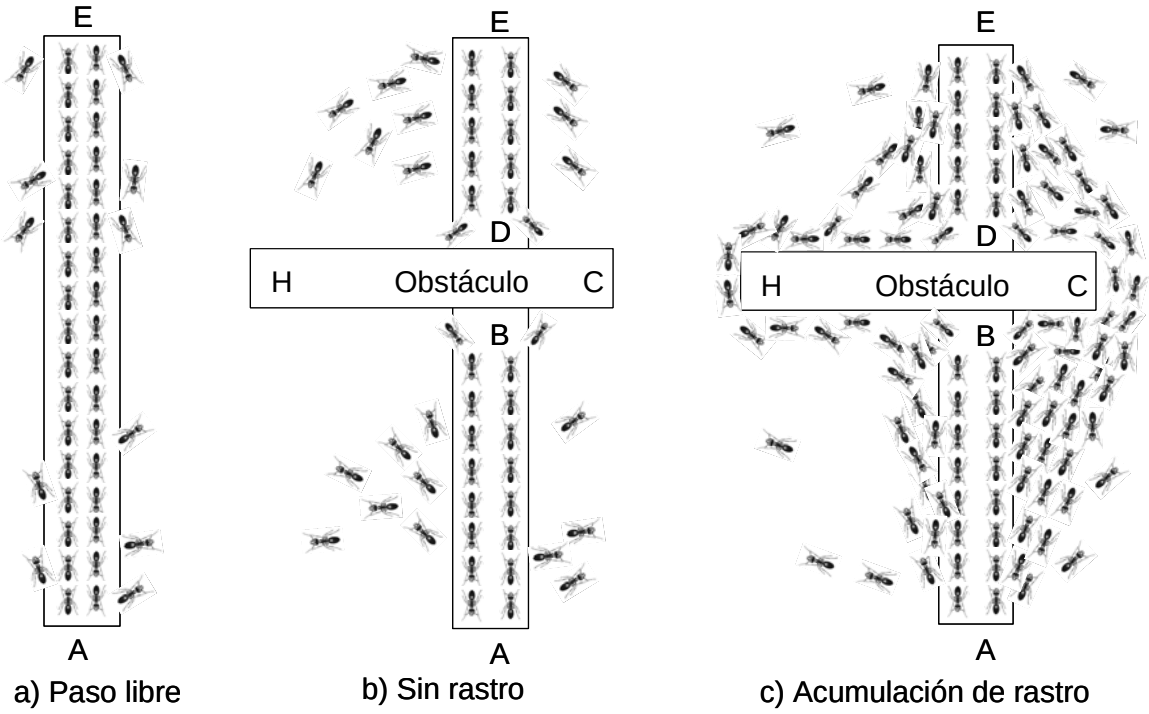


Figura 3 - El rastro de feromonas

Los caminos más cortos estarán asociados con los mayores niveles de feromonas, que dada la retroalimentación positiva de los individuos, hará que estos rastros crezcan tan rápidamente que las decisiones tomadas por cada individuo sean cada vez más sesgadas hacia estas rutas óptimas.

Nótese que si las hormigas se mueven con la misma velocidad y depositan feromonas a la misma tasa, entonces toman mayor tiempo en recorrer los caminos más largos, y contrariamente los rastros de feromonas se acumulan más rápido en las rutas más cortas. Por lo tanto, la denominada retroalimentación positiva hace que la búsqueda de las rutas más cortas sea el producto de la interacción entre la forma del obstáculo y el comportamiento de las hormigas.

Las hormigas artificiales que serán utilizadas en los algoritmos a desarrollarse tienen diferencias con respecto a las hormigas de los sistemas reales:

- Las hormigas artificiales tienen memoria del camino recorrido: recuerdan la ruta que han seguido y la distancia que han viajado.
- Las hormigas artificiales no son completamente ciegas: pueden calcular las distancias que las separan de un nuevo destino y el rastro de feromonas en el mismo.
- Las hormigas artificiales viven en ambientes de tiempos discretos: pueden pasar de un punto a otro instantáneamente.

2.4.2 El problema del Agente Viajero y los Sistemas de Hormigas

A continuación se presenta la implementación de los "Sistemas de Hormigas" a un típico problema combinatorio de optimización. En términos generales el problema del agente viajero (*Traveling Salesman Problem – TSP*), ha sido formulado para encontrar la ruta o circuito de longitud más corta que puede recorrer un individuo, de tal forma que visite sin repetir todas las ciudades de un conjunto dado. Típicamente el problema abordado en estos casos, corresponde a un circuito en el que todas las ciudades están interconectadas. Para representar este problema, se utiliza un grafo completamente conectado donde los nodos que lo componen típicamente se asocian a las ciudades, mientras que los arcos a las distancias entre ciudades.

Para resolver este problema aplicando la teoría de los "Sistemas de Hormigas", Dorigo y Gambardella (1997), una hormiga artificial se constituye en un agente que se mueve a través de cada nodo del grafo, y que utilizando una función probabilística que incorpora los rastros acumulados de feromonas presentes en los caminos al igual que un valor heurístico como función de la longitud de cada arco, selecciona la próxima ciudad que visitará. Bajo estas consideraciones, la hormiga artificial estará más sesgada a moverse desde su punto de origen, a ciudades cuyos arcos presenten mayores rastros de feromonas y que están más cercanos.

Hay tres ideas principales que son extraídas del comportamiento real de las hormigas y que son transferidas a las denominadas hormigas artificiales:

- La preferencia por los caminos con altos niveles de feromonas.

- El gran aumento de la cantidad de feromonas en lo caminos más cortos.
- La comunicación de los rastros entre las hormigas.

Dorigo y Gambardella (1997) plantea la siguiente formulación del TSP utilizando un sistema de hormigas artificiales, las cuales son capaces de determinar qué tan separadas se encuentran las ciudades aledañas y tienen la propiedad de tener una lista de memoria en la cual almacenan las ciudades previamente visitadas en cada excursión. La formulación propuesta es:

- G (N,E) se define como un grafo bidimensional totalmente interconectado, donde N es el conjunto de todas las ciudades y E es el conjunto de arcos que las interconectan.
- d_{ij} = Distancia euclidiana entre las coordenadas (x_i, y_i) (x_j, y_j), correspondientes a las ciudades *i* y *j*, respectivamente.
- n = Número total de ciudades que componen el grafo.

Al incorporar un sistema de hormigas para solucionar para el problema propuesto, es necesario tener en cuenta las siguientes consideraciones y definiciones.

- Cada hormiga artificial tiene los siguientes comportamientos:
  i. En el tiempo *t*, escoge la próxima ciudad que visitará en $(t+1)$ siguiendo una probabilidad función de las distancias entre las ciudades y la cantidad de feromonas en cada arco.
  ii. Puede visitar una ciudad una sola vez en el transcurso de una excursión, para lo cual se define.
    - $tabu_k(s)$ = Ciudad visitada por la hormiga *k* en el orden *s*.
  iii. Al completar una excursión, deja una sustancia denominada Rastro de Feromonas en cada arco (*i*,*j*) visitado.
- m = Número total de hormigas en el sistema.
- $b_i(t)$ = Número de hormigas en la ciudad *i* en el tiempo *t*, $m = \sum_{i=1}^{n} b_i(t)$, $\forall t$.
- $\tau_{ij}(t)$=Es la intensidad del rastro en el arco (i,j) en el tiempo *t*.

- $(1-\rho)$ = Constante de evaporación del rastro para cualquier arco entre el tiempo *t* y el *(t+n)*.
- $\eta_{ij} = 1/d_{ij}$ se define como la visibilidad en el arco (*i,j*).
- $P_{ij}^{k}(t)$ = Probabilidad de transición de la ciudad *i* a la *j* para la hormiga *k* en el tiempo *t*.

En el lapso comprendido entre el tiempo *t* y el tiempo *(t+1)*, se realizarán un total de m movimientos, asociados al desplazamiento de cada una de las *m* hormigas desde una ciudad a otra. La función probabilística que se utiliza para determinar cual es la ciudad *j* que se visitará en *(t+1)* dado que en el tiempo *t* se está en la ciudad *i*, es definida por la Ec. 19, donde α y β son parámetros de control de la importancia relativa entre el rastro y la distancia respectivamente.

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{(\tau_{ij}(t))^{\alpha}(\eta_{ij})^{\beta}}{\sum_{p \in Permisible} (\tau_{ip}(t))^{\alpha}(\eta_{ip})^{\beta}} & \forall\ j \in \text{permitido para la hormiga k} \\ 0 & \text{de lo contrario} \end{cases} \qquad \text{Ec. 19}$$

Al concluir estos movimientos, la lista tabú de cada una de las hormigas se actualiza con el objeto que no visite nuevamente a esa ciudad y de definir el conjunto de ciudades permisibles o no visitadas.

Es claro que después de *n* movimientos desarrollados por cada una de las hormigas, estas ya habrán visitado el total de *n* ciudades, concluyendo una iteración del heurístico. Una vez concluido un ciclo, se incorporan los resultados del mismo en las variables de decisión que serán tenidas en cuenta en la siguiente iteración.

$$\tau_{ij}(t+n) = \rho * \tau_{ij}(t) + \Delta\tau_{ij} \qquad \text{Ec. 20}$$

La actualización de información se realiza sobre el rastro de feromonas para cada uno de los arcos, siguiendo la Ec.20. El rastro de feromonas para el arco (*i,j)* en el tiempo *(t+n),* está compuesto por un residuo del rastro presente en el tiempo *t* y un Δ o variación en el rastro producto de los resultados obtenidos en dicha iteración.

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{\,k} \qquad \text{Ec. 21}$$

Donde, $\Delta\tau_{ij}^{\,k}$ es la cantidad por unidad de longitud de rastro de feromona dejado en el arco *(i,j)* por la hormiga *k* en el ciclo comprendido entre *t* y *(t+n)*, y esta definido en la Ec.22. Para este cálculo se introduce una constante Q de valor positivo y la variable L_k, definida como la longitud de la excursión recorrido por la hormiga *k* en el último ciclo.

$$\Delta\tau_{ij}^{\,k} = \begin{cases} Q / L_k & \text{Si la hormiga k usó el arco ij en su tour entre } t \text{ y } (t+n). \\ 0 & \text{De lo contrario} \end{cases} \qquad \text{Ec. 22}$$

Dorigo (Dorigo et al., 1996) muestra los resultados de experimentos asociados a la solución de un problema con 30 ciudades interconectadas que han denominado Oliver30 en (Whitley et al., 1996). Utilizando un número máximo NCMAX de 500 iteraciones, y modificando las combinaciones de los parámetros α y β, han determinado experimentalmente el número óptimo *m* de hormigas a ser utilizadas (como función de *n*) al igual que los valores de α y β más idóneos.

3 Metodología

En la primera parte del capítulo se describen en detalle las características y funcionamiento de las máquinas de recogido y depósito que son utilizadas al interior de la Fábrica Modelo.

Posteriormente se presenta una descripción del problema de minimización del tiempo de recogido para estas máquinas, tanto la formulación como los supuestos del modelo. Se presenta un problema pequeño que ayuda a clarificar la validez del modelo.

En la parte final se describe el proceso de implementación de los "Sistemas de Hormigas" para la realización del heurístico propuesto. Se presentan dos enfoques realizados y se resuelve el problema pequeño que fue propuesto al presentar la formulación matemática.

3.1 Descripción de las Máquinas Recogido y Depósito Fuji IP III

Las dos máquinas de recogido y depósito que son utilizadas al interior de la Fábrica Modelo, se pueden describir como máquinas de tecnología SMT, con dos brazos de movimientos independientes, que ejecutan instalaciones secuenciales coordinadamente.

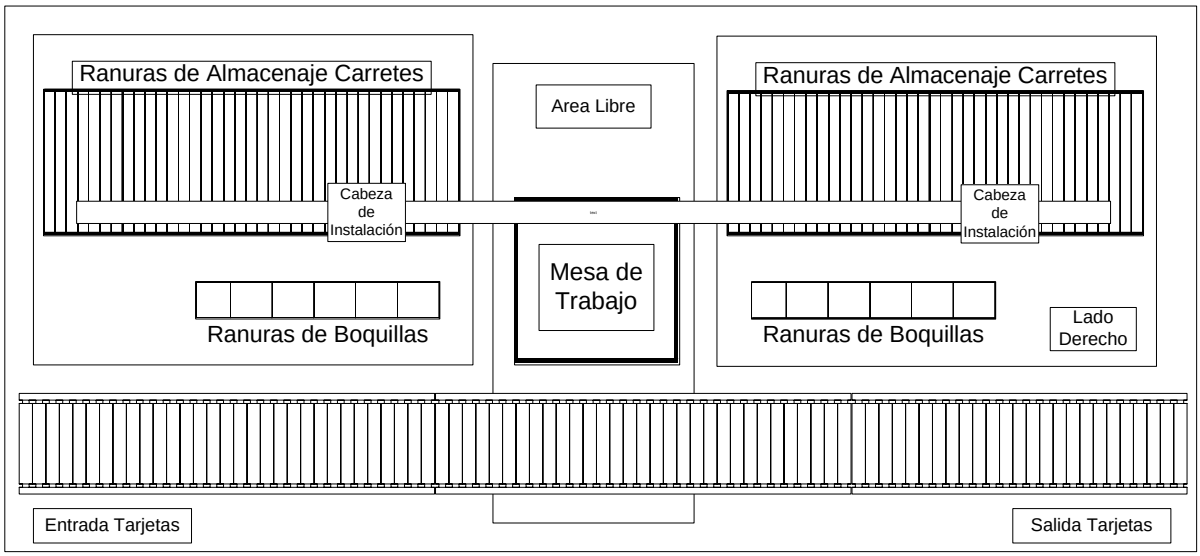


Figura 4 – Vista de tope Fuji IP III

La Figura 4 sirve como referencia para mostrar la ubicación de las partes principales de la máquina. Para entender claramente el funcionamiento de dicha máquina, es necesario describir las partes principales que la componen:

- **Mesa de trabajo**: Sobre la cual se ubica la tarjeta que va a ser ensamblada. La tabla puede maniobrar con una tarjeta a la vez, y los procesos de alimentación desde y descarga hasta la correa principal, se hace verificando que la instalación de todos los componentes se haya completado, al igual que la disponibilidad de espacio en la correa para recibir o entregar una nueva tarjeta. El movimiento permisible de esta tabla se hace sobre el eje Y, y además de asegurar la tarjeta para el proceso, se desplaza para ubicar la tarjeta en un posición especifica y que la cabeza de alimentación puede realizar la instalación de un componente.

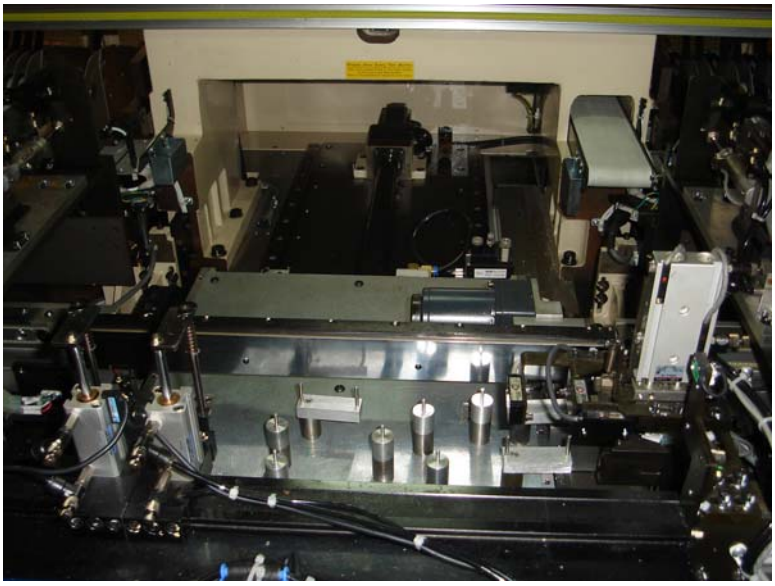


Figura 5 - Mesa de trabajo - Fuji IP III

- **2 Brazos de instalación:** Con movimientos independientes y ubicados sobre el eje X, en la parte superior de la máquina, uno al extremo derecho y otro en el izquierdo de la mesa de trabajo. Los movimientos permisibles se hacen sobre el eje X al desplazase sobre una varilla a la cual están sujetados y su función consiste en transportar las cabezas de instalación hasta los carretes de almacenamiento y/o herramientas de

instalación, y llevarlas de nuevo sobre a la tabla para que realicen la operación de instalación.

- **2 Cabezas de instalación:** Cada una fija a un brazo de instalación. El movimiento de las cabezas es sobre el eje Z, es decir un movimiento vertical. Una vez los brazos de instalación las llevan hasta una ubicación determinada sobre los carretes de alimentación, descienden hasta la cinta donde se ubica el componente y se realiza el proceso de recogido, inmediatamente vuelven a subir hasta la altura del brazo de instalación, para que éste las lleve hasta un punto situado encima de la mesa de trabajo, en ese momento desciende sobre la tarjeta y se ejecuta el proceso de depósito, para volver nuevamente a subir a la posición del brazo y comenzar otra secuencia de recogido y depósito.

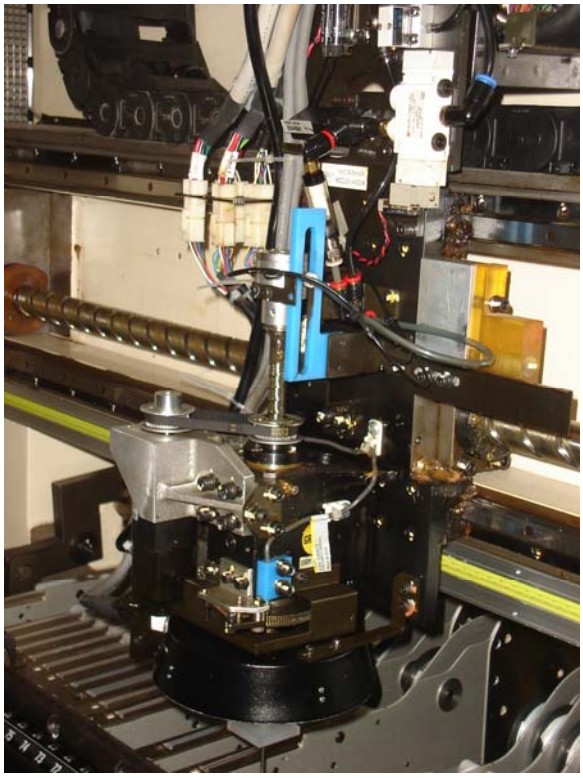


Figura 6 - Cabezas de instalación _ Fuji IP III

- **Las ranuras de almacenaje:** Distribuidas sobre el eje X a cada uno de los extremos de la mesa de trabajo, están ubicadas paralelamente y justamente debajo del eje sobre

el cual se desplazan los brazos. Dado que estas ranuras se encuentran distribuidas en dos grupos, cada brazo puede acceder solamente al grupo ubicado en su lado. En estas ranuras de alimentación se fijan los carretes con los componentes que serán instalados en la tarjeta y tienen un espesor de 8mm. Los carretes pueden presentar diferentes anchos de acuerdo al componente que almacenan (un único componente por carrete). Si todos los carretes de alimentación correspondieran a un grosor de 8mm se dispondría de un total de 37 carretes de alimentación a cada extremo de la mesa de trabajo. Cuando el grosor de un carrete es mayor al estándar, se le asigna más de un comportamiento. Una vez que un componente es retirado de un carrete por la cabeza de instalación, el carrete automáticamente se mueve a la siguiente posición disponible en la cual se encuentra un componente, similar a un proceso de desenrollar.

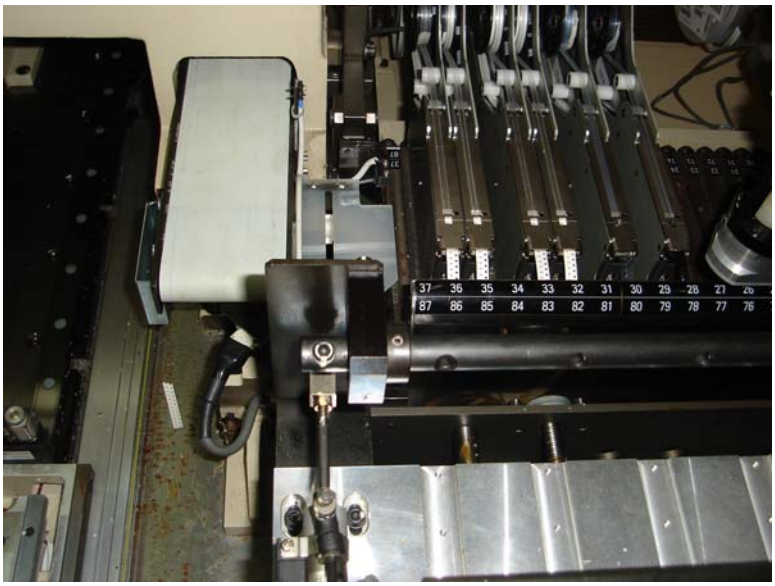


Figura 7 - Carretes de almacenamiento – Fuji IP III

- **Las herramientas de instalación o boquillas:** Corresponden a herramientas ubicadas en los extremos de las cabezas de instalación para que puedan realizar el proceso de recogido y depósito de un componente en particular. Dado que los componentes a instalar tienen características geométricas distintas, se requiere de una pieza específica para retirarlos de la cinta y posteriormente instalarlos en la tarjeta. Típicamente, una vez que finaliza el proceso de instalación de un tipo de componente,

los brazos de instalación desplazan las cabezas hasta los sitios donde se almacenan las boquillas (al frente y a la misma altura de los carretes de alimentación) para cambiar la herramienta e instalar la que será utilizada en el siguiente tipo de componente. Similar a las ranuras de almacenaje, se encuentran ubicados sobre el eje x, y se pueden tener hasta 6 diferente tipos de boquillas a cada lado de la mesa de trabajo, es decir, un total de 12 boquillas por cada máquina.

Figura 8 - Boquilla de instalación –Fuji IP III

En términos generales, el mecanismo de operación de las máquinas recogido y depósito Fuji IP III funciona de acuerdo a los siguientes macro procesos:

- Se asignan los tipos de componentes que serán instalados en cada una de las dos máquinas recogido y depósito. El software que controla las máquinas puede ser la asignación, aunque da la opción de hacer una asignación mixta (algunos pueden ser definidos previamente) o una total por parte del operador. En la actualidad dicha asignación se hace de acuerdo al tamaño del componente, en la primera máquina se instalan los componentes pequeños y en la segunda los componentes de mayor tamaño.

- Se instalan los carretes de componentes en las ranuras de almacenamiento al interior de cada una de las máquinas, dados los requerimientos de la tarjeta a ser ensamblaje. Igual al proceso de asignación de componentes a máquinas, el software permite las tres opciones previamente mencionadas.
- Una vez que la máquina ha sido configurada, software establece la secuencia en la cual se ubicaran cada uno de los componentes dado el despliegue de la tarjeta que esta siendo procesada, y la máquina comienza el proceso de instalación. Esta secuencia es establecida con el propósito de minimizar el tiempo para completar una tarjeta.
- El brazo de instalación se acerca hasta el almacenamiento de herramientas para instalar al boquilla requerida para el siguiente componente de la secuencia.
- El brazo de instalación lleva a la cabeza hasta un sitio específico en los carretes de alimentación, para que descienda y realice la operación de recogido. La cabeza se eleva hasta el brazo.
- El brazo comienza a moverse sobre el eje x hasta una posición dada, justamente encima de la mesa de trabajo donde esta la tarjeta. Simultáneo a esta actividad de recogido, la mesa de trabajo se desplaza en la dirección del eje y hasta una ubicación determinada, de tal modo que el brazo de instalación se ubique justamente arriba del punto donde se instalará el componente seleccionado.
- La cabeza de instalación desciende hasta la tarjeta y fija (solda) el componente. Una vez finalizado esta operación de depósito, asciende hasta el brazo para recoger el siguiente componente.
- En caso de cambiar el tipo de componente, el brazo se desplaza para cambiar la boquilla en caso de ser necesario, de lo contrario va hasta la ubicación del siguiente componente en la secuencia.
- Mientras uno de los brazos realiza la operación de recogido, el otro realiza la operación de depósito sincrónicamente (igual a la descrita previamente).

- Al finalizar la instalación de todos los componentes, la tarjeta es transportada a la correa de transporte, y se ingresa una nueva tarjeta para ser procesada.

Luego de revisar literatura asociada a este campo, aún cuando existen múltiples modelos para varios tipos de tecnología de recogido y depósito, no se ha encontrado alguno que refleje las características mencionadas para las máquinas Fuji IP III.

La motivación de esta tesis de maestría, justamente es el de desarrollar un heurístico basado en los "Sistemas de Hormigas" que ayude a mejorar el tiempo total de producción, enfocándose en la minimización del tiempo de acceso a los componentes (tomados de los carretes de almacenamiento) para el ensamblaje de un producto, dada una producción deseada, al determinar el número de carretes que se deben instalar, y asignar tanto las ranuras como las herramientas de instalación a cada lado de la máquina.

Esta buena configuración es el punto de partida para resolver el problema de secuenciación de actividades, el cual puede ser abordado con cualquiera de los enfoques tradicionales del área, e inclusive puede ser pensado el resolverlo al adaptar los "Sistemas de Hormigas", como trabajo futuro.

3.2 Minimización del Tiempo de Recogido para una Máquina Fuji IP III

En literatura revisada, no se ha encontrado una metodología, heurístico o formulación matemática que refleje el funcionamiento de las máquinas Fuji IP III de la Fábrica Modelo, que pueda ser usada de referencia para aplicar la teoría de los Sistemas de Hormigas.

El caso más cercano fue presentado en el Capítulo 2 y corresponde al de una máquina de instalación dual, haciéndose una explicación de las limitaciones de los dos modelos descritos.

Tomando como referencia las ideas fundamentales de los modelos revisados, se presenta a continuación una formulación para la minimización del tiempo de recogido para las máquinas Fuji IP III.

3.2.1 Consideraciones de la Formulación

Si bien es cierto que la forma tradicional en la cual se ha abordado el problema de minimizar el tiempo de recogido, ha sido el de resolver de forma secuencial el problema de "Asignación de Componentes" y posteriormente el de "Partición", la formulación que se ha desarrollado en esta tesis corresponde a un modelo global de minimización, en donde el número de tarjetas que son completadas corresponde a una restricción cuyo valor puede ser modificado.

Al presentar el modelo propuesto por Amhadi (Amhadi et al., 1990) para la "Asignación de Componentes", se hicieron algunos comentarios con respecto a su estructura y sobre la relevancia de tratarlo como un problema independiente. A continuación se hace una descripción más detallada de los cursos de acción que fueron sugeridos en dicho Capítulo.

El primero de los problemas se asocia a la relevancia de resolver este problema de una forma independiente. Si bien es cierto que es un problema típico de maximización, el cual puede ser trabajado como un problema entero, es posible desarrollar un método alternativo para encontrar una buena solución más fácilmente.

Se propone para tal efecto el implementar una rutina iterativa, en la cual se parte de una solución inicial y se comienza a aumentar la capacidad de producción, al aumentar el número de carretes a los tipos de componentes que la están limitando.

Un supuesto clave es que el número de componentes que se extraen desde un carrete tipo *i*, es igual al total de componentes de este tipo que son utilizados para completar una tarjeta, dividido entre el número de carretes del mismo que son instalados. Por tanto, el número de componentes que se extraen desde cada carrete de tipo *i* para completar una tarjeta es el mismo. Esto garantiza que en cada ensamblaje se consuman a la misma tasa, es decir, que todos los carretes del tipo *i* se terminan al mismo tiempo.

El número de tarjetas que pueden ser procesadas dada la disponibilidad del componente *i* es calculado como la división del número de componentes que se pueden almacenar en un

carrete de tipo *i* entre el número de componentes que se toman desde este carrete para completar una tarjeta.

Un seudo código al procedimiento propuesto se presenta a continuación:

- La solución inicial es el número de tarjetas que se pueden completar al asignar un único carrete a cada tipo de componente. Es necesario validar que las limitaciones de ranuras estándar pueda ser satisfecha con esta solución (se tiene en cuenta el número de ranuras estándar que ocupa cada carrete).
- Verificar cual es el tipo de componente que limita la producción y aumentar el número de carretes de este componente a instalarse, verificando que no se incumple la limitación del número de ranuras. El número de carretes que se van a asignar debe ser un divisor del número de componentes requeridos de este tipo para completar una tarjeta.
- Realizar este proceso de forma iterativa mientras no se incumpla la restricción del total de ranuras estándar disponibles.

Es importante señalar, que esta solución tiene la misma inconsistencia que fue planteada por Ahmadi (e.g., Ahmadi et al., 1990), en cuanto es posible que se obtenga una solución que no sea realmente factible al no poderse acomodar en los dos bancos de almacenamiento. En este trabajo de tesis, este problema es solucionado al incluir una rutina de verificación.

Continuando con la utilidad de resolver esto como un problema independiente, en términos prácticos, al momento de hacer una corrida de producción, se conoce el número de tarjetas que son requeridas y por tanto este puede ser un parámetro conocido de entrada para un modelo global.

Bastaría con ingresar este número de tarjetas mínimas como una restricción en el modelo de la asignación de carretes y hacer una verificación de su factibilidad. Si no es posible ubicar el número de carretes requeridos para completar dicha producción, es necesario disminuir el valor de este parámetro, hasta encontrar una solución factible.

Es claro que dada una producción mínima requerida, se calcula el número de carretes necesarios para cada tipo y se puede verificar la factibilidad o no de poderlos ubicar en el total de ranuras disponibles utilizando la pequeña rutina explicada previamente.

En resumen el modelo global planteado en esta tesis integra los dos problemas en uno solo, utilizando las siguientes ideas generales:

- El número de tarjetas requeridas para una corrida de producción es un valor conocido en la mayoría de los ambientes industriales y debe corresponder a un programa establecido de la planta.
- Se puede calcular el número de carretes requeridos para completar una producción dada y se verifica la factibilidad de instalar estos carretes en el total de ranuras.
- En caso de ser factible se busca una solución que minimice el tiempo de las operaciones de recogido, de lo contrario se debe disminuir el número de tarjetas requeridas.

3.2.2 Formulación Matemática

A continuación se presenta la formulación matemática para resolver el modelo global de minimización del tiempo de recogido para la máquina Fuji IP III, donde el número de tarjetas completadas se constituye en un parámetro que puede ser modificado y que determina la factibilidad de la solución.

Es necesario definir los siguientes conjuntos de índices.

- $i \in I,\ I = \{1, 2, 3, ..., nc\}$ - Asociado a los diferentes tipos de componentes.
- $j \in J,\ \mathrm{J} = \{1, 2, 3, ..., nn\}$ - Asociado a los diferentes tipos de nozzles.
- $k \in K,\ \mathrm{K} = \{1(Izquierda), 2(Derecha)\}$ - Asociado a los lados disponibles de almacenamiento.
- $l \in L,\ \mathrm{L} = \{1, 2, 3, ..., nr\}$ - Asociado al número máximo de posiciones que se podrían asignar.

Los parámetros de entrada del modelo son:

- b_i = Número de componentes de tipo i requeridos para completar una tarjeta.
- w_i = Número de ranuras estándar que requiere un carrete tipo i.
- v_i = Número de componentes tipo i que pueden ser almacenados en un carrete.
- W = Total de ranuras estándar disponibles en cada lado k.
- Z = Total de espacios disponibles para boquillas en cada lado k.
- $c_{ij} = \begin{cases} 1 \text{ Si un componente de tipo } i \text{ puede ser instalado utilizando una boquilla tipo } j. \\ 0 \text{ De lo contrario.} \end{cases}$
- nt = Distancia de penalización asociada al cambio de una boquilla.
- N = Número total de tarjetas mínimo que se deben completar en una corrida.

Es necesario definir a qué se refiere el índice de posición. La posición l es el lugar relativo que se le asigna a cada carrete dentro del espacio físico de las ranuras. El número máximo de posiciones que podrían ser asignadas es igual a $2W$, donde la posición 1 se asocia a la ubicación más cercana y la posición W a la más lejana con respecto al lado izquierdo, mientras que las posiciones $W+1$ y $2W$ corresponden a las ubicaciones más cercana y más lejana respectivamente, con respecto al lado derecho.

Las variables de decisión del modelo son:

- x_{il} = Número de componentes de tipo i colocados desde un carrete ubicado en la posición l para el ensamblaje de una tarjeta.
- $y_{il} = \begin{cases} 1 \text{ Si la posición } l \text{ es asignada a un carrete de tipo } i. \\ 0 \text{ De lo contrario.} \end{cases}$
- d_l = Tiempo requerido para desplazarse hasta la posición l desde un punto de referencia en el banco de almacenamiento correspondiente (unas posiciones se asocian al lado izquierdo, otras al lado derecho).

- $t_{jk} = \begin{cases} 1 \text{ Si la boquilla tipo } j \text{ esta disponible en lado } k (k = 1 \text{ izquierda, } k = 2 \text{ derecha}). \\ 0 \text{ De lo contrario.} \end{cases}$

- a_i = Número de carretes de tipo i a ser instalados.

Un punto a mencionar es el por qué las distancias son variables de decisión del modelo. Dado que el interés es el asignar una posición a cada carrete dentro del espacio físico disponible, la distancia recorrida para alcanzar un componente desde un carrete específico, se asocia no solo a la posición, sino al número de ranuras que son utilizadas por los carretes que son precedentes en ubicación con respecto al punto de referencia (posiciones más cercanas). Es decir, la distancia desde un punto de referencia en la mesa de trabajo a un carrete ubicado en la posición *l*, depende del número de ranuras estándar que son utilizadas por los carretes que han sido asignados en posiciones más cercanas. La Figura 9 es útil para aclarar el concepto de posición asignada a un carrete y del por qué la distancia es una variable de decisión.

La Figura 9a, presenta un esquema de las ranuras estándar disponibles en la máquina que se han numerado teniendo en cuenta que tan cercanas se encuentran a cada punto de referencia, donde 1 es lo más cercano y *W* lo más lejano a cada lado.

La Figura 9b presenta una posible distribución para un ejemplo, en el cual se pretenden organizar carretes con diferentes tamaños para tres tipos de componentes (A, B y C) en la totalidad del espacio disponible. Un carrete de tipo A ocupa 2 ranuras estándar ($w_A=2$), mientras que los carretes de B y C solamente ocupan una. ($w_B = w_c=1$).

La Figura 9c presenta la posición relativa que se le ha asignado a cada uno de los carretes de cada tipo de componente. Nótese que el número máximo de posiciones que se podrían asignar a cada lado es *W*, pero cuando algunos carretes ocupan más de una ranura estándar, este total de posiciones es menor (en el ejemplo *E* y *S* son menores a *W*). Además, es importante mencionar que al tener en cuenta la restricción del número de ranuras disponibles, ésta debe dividirse en dos: una para limitar el número de ranuras asignadas a la izquierda y otra para a la derecha. Aún cuando se pueda cumplir que la suma de ranuras requeridas no

sobrepase la capacidad total, es posible que dicha asignación no pueda ser organizada en los dos lados.

Por último, en la Figura 9d se asocia la distancia con respecto al punto de referencia para algunas de las posiciones. Si tomamos como referencia el carrete tipo B asignado en la posición 3, la distancia que se debe recorrer para acceder desde el punto de referencia en el lado izquierdo es de 4 unidades estándar, dado que la posición 1 se encuentra un carrete tipo A que ocupa las dos primeras ranuras estándar y en la posición 2 se encuentra un carrete tipo C que ocupa una ranura. En el caso que la posición 1 hubiera sido asignada a un carrete tipo C, la distancia hasta la posición 3 no seria de 4 unidades, sino solamente de 3.

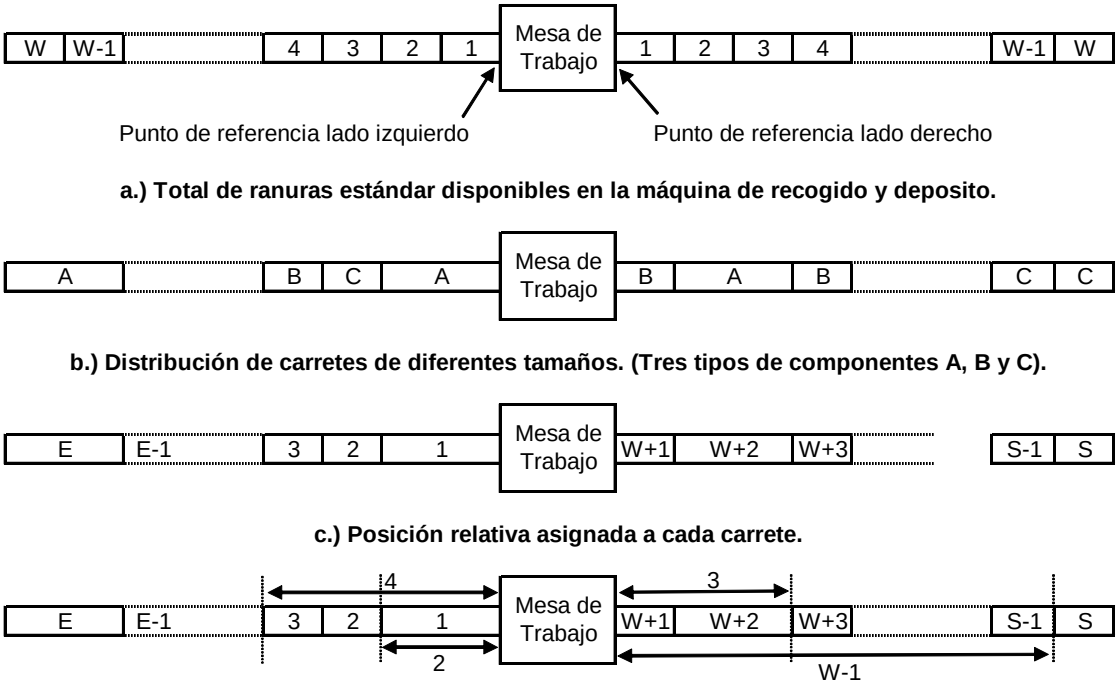


Figura 9 - Distancias a ranuras y a posiciones.

El modelo matemático propuesto para disminuir el tiempo de las operaciones de recogido dado un número mínimo de tarjetas que deben ser completadas, para una máquina de tecnología SMT con las características de la Fuji IP III, se presenta a continuación.

La función objetivo (Ec. 23) está compuesta por dos términos: el primero de ellos corresponde a la suma de las distancias para acceder aisladamente a cada componente requerido para completar una tarjeta, mientras que el segundo a una penalización en términos de distancia que se asocia al tiempo invertido en el cambio de boquillas necesarios para hacer dichas instalaciones.

Encontrar $\{x_{il}, y_{il}, a_i, d_l, t_{jk}\}$ de modo que

$$Min \quad (\sum_{l}\sum_{i} x_{il} y_{il} d_l + \sum_{j}\sum_{k} t_{jk} nt), \qquad \text{Ec. 23}$$

El propósito de esta función es organizar los carretes en cada lado de la máquina, de tal forma que las distancias recorridas para alcanzar todos los componentes y realizar el cambio de las boquillas sea el menor posible.

Aún cuando el proceso de instalación se desarrolla en forma paralela entre los dos brazos (mientras uno realiza un recogido el otro realiza un depósito), el problema de encontrar una óptima sincronización en dichos movimientos corresponde al tercero de los problemas en la optimización de la tecnología de SMT, el cual no es considerado en esta tesis. No obstante, el asignar una posición a los carretes de tal forma que se minimice la distancia que se debe recorrer para alcanzar todos los componentes requeridos para una tarjeta, es el punto de partida óptimo para hacer una programación de *n* trabajos en dos máquinas de forma sincrónica.

La primera inquietud parte del por qué utilizar la distancia hasta la posición asignada sobre un punto de referencia a cada lado, y no el factor de tiempo utilizado en dicho desplazamiento. La razón básica es que se puede asumir que los brazos de instalación se mueven a una velocidad constante, por lo tanto la distancia recorrida hasta cierta ubicación desde un punto de referencia es un parámetro proporcional al tiempo empleado en dicha acción, y a la distancia es más fácil de calcular.

La segunda observación es que el parámetro *nt* se define como una penalidad en términos de distancia, dado el desplazamiento requerido para el cambio de una boquilla. Entre más grande sea este valor, el modelo tratará de buscar soluciones que incluya el menor número de

cambios, mientras que con una penalización pequeña, sería más permisible en este sentido y se centraría en disminuir los tiempos de acceso a los componentes. Luego de verificar el proceso de producción actual en la Fábrica Modelo y consultar con los encargados de las máquinas, se ha definido que un valor aceptado de penalización (*nt*) debe ser aproximadamente igual a *2W*.

A continuación se presentan las restricciones asociadas.

$$x_{il} N \le v_i, \qquad \forall i \in I, l \in L. \qquad \text{Ec. 24}$$

La Ec. 24 garantiza que el número de unidades que se extraen desde cada carrete para cumplir con el total de tarjetas requeridas (*N*) no sobrepase el número de unidades que son almacenadas. Esta restricción es el punto de unión para los problemas 1 y 2. El valor de *N* puede ser incrementado hasta tanto el problema encuentre una solución óptima.

$$\sum_{l=1}^{2W} x_{il} = b_i, \qquad \forall i \in I. \qquad \text{Ec. 25}$$

$$\sum_{l}^{2W} y_{il} = a_i, \qquad \forall i \in I. \qquad \text{Ec. 26}$$

$$\sum_{i=1}^{nc} y_{il} <= 1, \qquad \forall l \in L. \qquad \text{Ec. 27}$$

La Ec. 25 garantiza que el total de componentes de tipo *i* que son tomados desde todas las posiciones para completar una tarjeta sea igual a los requerimientos por tarjeta de este tipo de componente. El total de carretes de cada tipo que son requeridos para cumplir con la producción deseada es calculado en la Ec. 26. La Ec. 27 garantiza que una posición no pueda ser asignada a más de un carrete.

$$x_{il} \le v_i y_{il}, \qquad \forall i \in I, l \in L. \qquad \text{Ec. 28}$$

$$y_{il} \le x_{il}, \qquad \forall i \in I, l \in L. \qquad \text{Ec. 29}$$

Las ecuaciones 28 y 29 sirven para unir la asignación de una posición a un carrete tipo *i* con la utilización del mismo para completar una tarjeta. Es decir, si una posición es asignada a un carrete, se deben extraer componentes desde el mismo, y si se han de extraer

componentes desde un carrete en cierta posición, se debe asegurar que dicha posición se asigne a tal componente. La Ec. 30 sirve para garantizar que el número de componentes que se extraen desde cada carrete de tipo *i* sea el mismo y es equivalente al número de componentes necesarios para una tarjeta divido entre el número de carretes a instalarse.

$$x_{il} a_i \le b_i, \qquad \forall i \in I, l \in L. \qquad \text{Ec. 30}$$

$$d_r = \sum_{l=1}^{r} \sum_{i=1}^{nc} w_i y_{il}, \qquad 1 \le r \le W. \qquad \text{Ec. 31}$$

$$d_r = \sum_{l=W+1}^{r} \sum_{i=1}^{nc} w_i y_{il}, \qquad W+1 \le r \le 2W. \qquad \text{Ec. 32}$$

Las ecuaciones 31 y 32 ayudan a determinar el valor de la variable de decisión d_l como función del tipo de carrete asignado en dicha posición y de los carretes que se han de instalar en posiciones más cercanas a la *l* con respecto al punto de referencia.

La restricción que el número de ranuras utilizadas en cada banco de instalación no sobrepasen la capacidad son las ecuaciones 33 y 34.

$$\sum_{i=1}^{nc} \sum_{l=1}^{W} w_i y_{il} \le W, \qquad \text{Ec. 33}$$

$$\sum_{i=1}^{nc} \sum_{l=W+1}^{2W} w_i y_{il} \le W, \qquad \text{Ec. 34}$$

Con el fin de garantizar que una vez se ha asignado un carrete cada uno de los lados, se ubiquen al menos una boquilla que pueda ser utilizada para su manipulación se utilizan las ecuaciones 35 y 36. Por último la restricción 37 sirve para garantizar que no se utilicen más boquillas de las que pueden ser almacenadas en cada lado.

$$\sum_{l=1}^{W} y_{il} \le b_i \sum_{j=1}^{nn} c_{ij} t_{j1}, \qquad \forall i \in I. \qquad \text{Ec. 35}$$

$$\sum_{l=W+1}^{2W} y_{il} \le b_i \sum_{j=1}^{nn} c_{ij} t_{j2}, \qquad \forall i \in I.$$

Ec. 36

$$\sum_{j=1}^{nn} t_{jk} \le Z, \qquad \forall k \in K. \qquad \text{Ec. 37}$$

Donde se las variables de decisión se definen como:

$$x_{il} \in Z^{+}, \qquad \forall i \in I, l \in L. \qquad \text{Ec. 38}$$

$$y_{il} \in \{0,1\}, \qquad \forall i \in I, l \in L. \qquad \text{Ec. 39}$$

$$t_{jk} \in \{0,1\}, \qquad \forall j \in J, k \in K. \qquad \text{Ec. 40}$$

El modelo resultante corresponde a uno con una función objetivo no lineal, con variables enteras y binarias, el cual tiene características similares a los modelos descritos por Ahmadi (1990). Su naturaleza combinatoria lo hace ser un problema difícil de resolver y el encontrar una solución óptima es muy costoso en cuanto al tiempo de computadora se refiere.

3.2.3 Un Modelo Pequeño

Con el propósito de comprobar que la formulación matemática propuesta refleja las características de la máquina de recogido y depósito Fuji IP III y determinar el tiempo que le toma a un software convencional de optimización (Lingo 8.0) encontrar la solución optima, se ha resuelto el problema para una tarjeta pequeña con características similares a las que actualmente se producen en la Fábrica Modelo.

Se ha tomado como referencia la información de una tarjeta real y se ha concebido una con 7 tipos de componentes y un requerimiento de producción de 200 tarjetas con una misma configuración.

Las características asociadas a los requerimientos de una tarjeta de cada tipo de componente (b_i), el número de unidades almacenadas en cada carrete (v_i) y el número de ranuras estándar que ocupa cada uno (w_i), son resumidas en la Tabla 1.

Tabla 1 - Características por Tipo de Componente

Tipo de Componente (i)	Unidades por Carrete (v_i)	Unidades por Tarjeta (b_i)	Ranuras Estándar (w_i)
1	3,000	12	3
2	3,000	18	1
3	5,000	54	1
4	3,000	24	2

La máquina tiene un total de 7 ranuras estándar (W) y 5 lugares para el almacenamiento de boquillas (Z) a cada lado de la tabla de instalación. El valor de la penalización por el cambio de las herramientas de instalación (nt) es igual a 15 (nótese que este valor es mayor a $2W$).

Se tienen un total de 5 diferentes tipos de boquillas disponibles y la matriz de correspondencias con los tipos de componentes C se resume en la Tabla 2.

Tabla 2 – Matriz C

Tipo de Componente	Tipo de *"Nozzle"* 1	2	3	4	5
1	1	0	1	1	0
2	0	0	1	1	0
3	1	1	1	0	0
4	1	0	0	0	0

Siguiendo lo propuesto en la Sección anterior, el modelo de formulación para este modelo pequeño se presenta a continuación.

Encontrar $\left\{x_{il}, y_{il}, a_i, d_l, t_{jk}\right\}$ de modo que

$$Min \quad (\sum_{l=1}^{14}\sum_{i=1}^{4} x_{il} y_{il} d_l + \sum_{j=1}^{5}\sum_{k=1}^{2} 15 t_{jk}), \qquad \text{Ec. 41}$$

Sujeto a:

$$200 x_{il} \le v_i, \qquad \forall i \in \{1,2,3,4\}, l \in \{1,2,...14\}. \qquad \text{Ec. 42}$$

$$\sum_{l=1}^{14} x_{il} = b_i, \qquad \forall i \in \{1,2,3,4\}. \qquad \text{Ec. 43}$$

$$\sum_{l}^{14} y_{il} = a_i, \qquad \forall i \in \{1,2,3,4\}. \qquad \text{Ec. 44}$$

$$\sum_{i=1}^{4} y_{il} <= 1, \qquad \forall l \in \{1,2,...14\}. \qquad \text{Ec. 45}$$

$$x_{il} \le v_i y_{il}, \qquad \forall i \in \{1,2,3,4\}, l \in \{1,2,...14\}. \qquad \text{Ec. 46}$$

$$y_{il} \le x_{il}, \qquad \forall i \in \{1,2,3,4\}, l \in \{1,2,...14\}. \qquad \text{Ec. 47}$$

$$x_{il} a_i \le b_i, \qquad \forall i \in \{1,2,3,4\}, l \in \{1,2,...14\}. \qquad \text{Ec. 48}$$

$$d_r = \sum_{l=1}^{r} \sum_{i=1}^{4} w_i y_{il}, \qquad 1 \le r \le 7. \qquad \text{Ec. 49}$$

$$d_r = \sum_{l=1}^{r} \sum_{i=1}^{4} w_i y_{il}, \qquad 7 \le r \le 14. \qquad \text{Ec. 50}$$

$$\sum_{i=1}^{4} \sum_{l=1}^{7} w_i y_{il} \le 7, \qquad \text{Ec. 51}$$

$$\sum_{i=1}^{4} \sum_{l=8}^{14} w_i y_{il} \le 7, \qquad \text{Ec. 52}$$

$$\sum_{l=1}^{7} y_{il} \le b_i \sum_{j=1}^{5} c_{ij} t_{j1}, \qquad \forall i \in \{1,2,3,4\}. \qquad \text{Ec. 53}$$

$$\sum_{l=8}^{14} y_{il} \le b_i \sum_{j=1}^{5} c_{ij} t_{j2}, \qquad \forall i \in \{1,2,3,4\}. \qquad \text{Ec. 54}$$

$$\sum_{j=1}^{5} t_{jk} \le 5, \qquad \forall k \in \{1,2\}. \qquad \text{Ec. 55}$$

$$x_{il} \in Z^+, \qquad \forall i \in \{1,2,3,4\}, l \in \{1,2,...14\}. \qquad \text{Ec. 56}$$

$$y_{il} \in \{0,1\}, \qquad \forall i \in \{1,2,3,4\}, l \in \{1,2,...14\}. \qquad \text{Ec. 57}$$

$$t_{jk} \in \{0,1\}, \qquad \forall j \in \{1,2,...5\}, k \in \{1,2\}. \qquad \text{Ec. 58}$$

La función objetivo para este modelo pequeño se presenta en la Ecuación 41. El primer término se asocia a la distancia recorrida para alcanzar los componentes requeridos de los 4 diferentes tipos para completar una tarjeta, mientras que el segundo se asocia al tiempo empleado en el cambio de boquillas. Las restricciones que se derivan de la Ecuación 42

garantizan que se completan al menos 200 tarjetas, mientras que las planteadas por la Ecuación 43 hacen que el número de componentes de tipo *i* que son tomados desde todas las posiciones para completar una tarjeta sea igual a los requerimientos por tarjeta de este tipo de componente. El total de carretes de cada tipo que son requeridos para cumplir con la producción deseada es calculado en la Ecuación 44. La Ecuación 45 garantiza que una posición (hay un máximo de posibles 14 posiciones) no pueda ser asignada a más de un carrete.

Las Ecuaciones 46 y 47 sirven para atar la asignación de una posición a un carrete tipo *i* con la utilización del mismo para completar una tarjeta. Es decir, si una posición es asignada a un carrete, se deben extraer componentes desde el mismo, y si se han de extraer componentes desde un carrete en cierta posición, se debe asegurar que dicha posición se asigne a tal componente. La Ecuación 48 sirve para garantizar que el número de componentes que se extraen desde cada carrete de tipo *i* sea el mismo y es equivalente al número de componentes necesarios para una tarjeta divido entre el número de carretes a instalarse.

Las Ecuaciones 49 y 50 ayudan a determinar el valor de la variable de decisión d_l. La restricción que el número de ranuras utilizadas en cada banco de instalación no sobrepasen la capacidad (7) son las Ecuaciones 51 y 52.

Con el fin de garantizar que una vez se ha asignado un carrete cada uno de los lados, se ubique al menos una boquilla que pueda ser utilizada para su manipulación se utiliza las Ecuaciones 53 y 54. Por último la Ecuación 55 garantiza que no se utilicen más boquilla de las que pueden ser almacenadas en cada lado (5).

Teniendo en cuenta la composición de la función objetivo y la relevancia que tiene el valor de penalización que se le asigne al cambio de boquilla, se ha decidido resolver dos problemas asociados.

El Caso 1 es el más sencillo, y busca minimizar el tiempo de las operaciones de recogido sin tener en cuenta la utilización de boquillas, mientras que en el Caso 2 se realiza la asignación de boquillas y carretes simultáneamente.

Para resolver el Caso 1, basta con reemplazar la función objetivo (Ec.41) por la Ecuación 59, al igual que eliminar las ecuaciones 53, 54, 55 y 58, en tanto que el Caso 2 comprende el modelo completo previamente explicado

$$Min \quad \sum_{l=1}^{14} \sum_{i=1}^{4} x_{il} y_{il} d_l, \qquad \text{Ec. 59}$$

El modelo planteado en Lingo 8.0 para el Caso 1, al igual que el reporte de resultados que proporciona el software se presenta en el Apéndice B.1; en cuanto al Caso 2 se refiere, estos se presentan en el Apéndice B.2.

Después de un tiempo de ejecución de 70 minutos, aunque Lingo 8.0 encontró soluciones factibles para los dos problemas, no encontró la solución óptima para ninguno de los dos casos enumerados. Es importante señalar que Lingo utiliza el procedimiento de *"Branch and Bound"* para resolver estos problemas enteros. Por la naturaleza de este procedimiento de optimización, en cada iteración se reporta el valor mínimo que podría tomar la función objetivo aún cuando este pueda no ser factible, y a medida que va descartando ramas (posibilidades) este valor puede ir incrementándose (es un problema de minimización). El procedimiento se detiene cuando la mejor solución factible encontrada es igual a ese valor mínimo que se podría alcanzar. Lingo nos reporta este valor mínimo y se denomina Límite Inferior. Es posible que en el largo plazo, el programa encuentre que esta mejor solución factible encontrada después de una hora, sea la solución óptima.

La Tabla 3 es útil para presentar los resultados obtenidos en cuenta a los función objetivo se refiere para los dos casos.

Tabla 3 - Resultados Lingo 8.0

Caso	Tiempo [minutos]	Solución Optima	Mejor Solución Factible	Límite Inferior
1	70	No encontrada	309	240.582
2	70	No encontrada	360	280.43

Para el Caso 1 se presenta en la Tabla 4 la asignación asociada a la mejor factible encontrada por el programa.

Tabla 4 - Resultados Caso 1

Posición	Tipo de Componente	Unidades por Tarjeta	Distancia
1	3	18	1
2	3	18	2
3	2	9	3
5	4	12	5
9	3	18	1
10	2	9	2
11	4	12	4
14	1	12	7
	Distancia Total Recorrida		309

Tabla 5 - Resultados Caso 2

Posición	Tipo de Componente	Unidades por Tarjeta	Distancia
1	3	18	1
2	3	18	2
3	1	12	5
8	2	9	1
11	3	18	2
12	2	9	3
13	4	12	5
14	4	12	7
	Distancia Total Recorrida		330

"Nozzle"	**Lado Izquierdo**	**Lado Derecho**	**Total**
1	1	1	2
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
		Cambios de *"Nozzles"*	2
		Valor Penalización	30
		Función Objetivo	360

En tanto que para el caso 2, los resultados se presentan en la Tabla 5. Es importante señalar que esta solución factible presentada por Lingo incumple con el problema de asignación de boquillas.

3.3 Aplicación de los Sistemas de Hormigas a la Minimización del Tiempo de Recogido para la Máquina Fuji IP III

Ya se han definido claramente las características de la máquina de instalación estudiada, al igual que se ha planteado un modelo de optimización con el propósito de minimizar el tiempo de recogido, resolviendo dos de los tres problemas que son estudiados en la optimización de la tecnología de SMT.

Ahora corresponde el explicar la forma en la que esta formulación ha sido adaptada a la familia de heurísticos de los "Sistemas de Hormigas" y posteriormente verificar la bondad de usar una "buena" solución, frente al tiempo y recursos que se deben invertir para encontrar la solución óptima utilizando algún software especializado en optimización.

En el Capítulo 2, se señala la filosofía que se ha seguido al momento de adaptar la teoría de los "Sistemas de Hormigas" a un problema tan común como lo es el del agente viajero. Apoyado en la descripción detallada de varias adaptaciones realizadas a problemas combinatorios que hace Dorigo (e.g., Dorigo et al., 2004), se pueden listar las siguientes como características generales para la aplicación de los Sistemas de Hormigas:

- Una hormiga es una solución única y completa para un problema. Aún cuando en la mayoría de los casos se recomienda la construcción de soluciones que sean factibles, la idea de estos heurísticos, permite la construcción de algunas que no lo sean y que posteriormente pueden ser descartadas.
- Para la generación de una hormiga deben existir valores heurísticos, que den la posibilidad que durante la construcción de una solución se puedan tomar diferentes decisiones. La existencia de estos parámetros proporciona aleatoriedad al modelo y permite la generación de múltiples soluciones, sean estas factibles o no.
- No existe una regla general para la selección de los valores heurísticos, pero se requiere que estos involucren características deseadas de una buena solución y ayudan a sesgar las decisiones (posteriormente se explica su selección). La selección de buenos valores mejora dramáticamente el comportamiento del heurístico.

- Los parámetros propios de los Sistemas de Hormigas como lo son el número de réplicas, el número de colonias por réplicas y el total de hormigas por colonia, al igual que lo valores de importancia relativa para los valores heurísticos y determínisticos y la conservación del rastro de feromonas, no pueden ser definidos con certeza antes de la implementación. Una vez se ha desarrollado una rutina de aplicación, se requiere que se hagan pruebas para encontrar los valores que dadas las características del problema proporcionen la "mejor" solución factible. Entiéndase que se procura encontrar una muy buena solución y no de encontrar la solución óptima.
- Permite la integración con rutinas determínisticas. Es posible que durante la construcción de una hormiga (solución), para la siguiente decisión se pueda encontrar la mejor alternativa de forma exacta y no necesariamente se requiera la utilización de una rutina heurística.

3.3.1 Extensión de los Sistemas de Hormigas

El enfoque presentando en el Capítulo 2 corresponde a la versión tradicional de los Sistemas de Hormigas. Sin embargo, Dorigo (Dorigo et al., 2004) plantea una extensión al introducir los Sistemas de Colonias de Hormigas, los cuales alcanzan mejores resultados y nacen de cambios menores sobre los Sistemas de Hormigas.

Estos Sistemas de Colonias de Hormigas difieren del enfoque tradicional en tres puntos fundamentales: se hace una búsqueda más intensiva de soluciones basada en la experiencia acumulada por otras hormigas, la evaporación y depósito de rastros de feromonas se hace teniendo como base las mejores soluciones encontradas, y a medida que se construye una solución los rastros de feromonas se van disminuyendo para fomentar la búsqueda de nuevas soluciones.

Tomando como referencia algunas de estas modificaciones se ha desarrollado la implementación que se plantea en este proyecto de tesis.

3.3.2 Consideraciones del Modelo

Las consideraciones sobre las cuales se ha desarrollado esta implementación son:

- La noción del tiempo ha sido modificada por distancia recorrida, dadas las consideraciones expresadas al formular el modelo de optimización
- Se asume que las cabezas de instalación no tienen movimientos simultáneos, de tal forma que se minimiza el peor de los casos, es decir, en la ausencia de sincronización.
- Una vez que se ha instalado una boquilla a una cabeza, se instalaran todos los componentes posibles con dicha herramienta, para que sea utilizada solamente una vez.
- La penalización asociada a los cambios de boquillas es grande. Se ha establecido como una distancia mayor a la recorrida para ir y volver hasta la ranura más lejana en cada banco de instalación ($2W$). Se asume que esta penalización es una constante, independiente del orden de las boquillas dentro de los almacenes de herramientas en cada lado.
- La selección de una boquilla es la primera decisión, sobre la cual se hace una asignación de posición a los carretes de los componentes que pueden ser instalados con dicha herramienta.
- El máximo de posiciones que podrían ser asignados a los componentes, es igual al total de ranuras estándar disponibles ($2W$). Dada la existencia de diferentes tamaños de carretes, es posible que no sea factible la asignación de todas las posiciones, aún cuando éstas sean disponibles en la implementación.
- El heurístico asignará una posición a cada carrete de cada tipo de componente, permitiendo que la posición asignada pueda ser mayor a la máxima posición (con respecto al mismo punto de referencia) que haya sido asignada hasta ese momento en la solución. De tal forma, que es posible que para cierta instancia, se hayan asignado las posiciones $n-1$ y $n+1$, mientras que la posición n no se ha asignado.

- Una vez que se ha completado una solución, es posible que posiciones intermedias no hayan sido asignados, por lo cual se utilizará una rutina que corrija estos huecos de asignación.

3.3.3 Parámetros Generales de Entrada

Los parámetros de entrada para la implementación son:

- N = Número mínimo de tarjetas que deben ser completadas.
- b = Vector con los requerimientos por tipo de componente para una tarjeta.
- v = Vector con la capacidad de almacenaje de cada tipo de carrete.
- w = Vector con los requerimientos del número de ranuras estándar por cada tipo de carrete.
- C = Matriz binaria en la cual se refleja la correspondencia de cada boquilla para la instalación de cada tipo de componente.
- L = Total de ranuras estándar disponibles en la máquina (número máximo de posiciones que podrían ser asignadas). Es un número par.
- W = Total de ranuras estándar disponibles en cada lado de la máquina ($L/2$).
- Z = Número máximo de boquillas que pueden almacenarse en cada lado de la máquina.
- nt = Penalización asociada al cambio de boquillas (mayor que $2W$).
- K = Número de lados disponibles en la máquina.

3.3.4 Parámetros Asociados a los Sistemas de Hormigas

En cuanto a los parámetros propios de la implementación de los Sistemas de Hormigas, se deben definir los siguientes:

- Rep = Número de réplicas del heurístico.
- Col = Número de colonias generadas en cada réplica.

- *Hor* = Número de hormigas que componen cada colonia.
- α = Parámetro de importancia del rastro de feromonas.
- β = Parámetro de importancia a la información heurística.
- *Ro* = Coeficiente de evaporación del rastro de feromonas entre cada colonia.
- q = Valor asociado a la probabilidad de mejorar las buenas soluciones encontradas, frente a la búsqueda de nuevas soluciones.

3.3.5 Variables de Decisión

Las variables de decisión involucradas en el modelo y que corresponden a los resultados de cada hormiga se describen a continuación.

- $y(r,c,h,i,l) = \begin{cases} 1 \text{ Si la hormiga } h \text{ de la colonia } c \text{ en la réplica } r, \text{ asigna la posición } l \\ \quad \text{a un carrete con componentes tipo } i. \\ 0 \text{ De lo contrario.} \end{cases}$
- $x(r,c,h,i,l) = \begin{cases} \text{n Si la hormiga } h \text{ de la colonia } c \text{ en la réplica } r, \text{ debe seleccionar} \\ \quad \text{desde la posición } l \text{ un total de } n \text{ componentes tipo } i. \\ 0 \text{ De lo contrario.} \end{cases}$
- $t(r,c,h,i,l) = \begin{cases} 1 \text{ Si la hormiga } h \text{ de la colonia } c \text{ en la réplica } r, \text{ asigna la boquilla } j \\ \quad \text{al lado } k \text{ de la máquina.} \\ 0 \text{ De lo contrario.} \end{cases}$

3.3.6 Construcción de Soluciones

Cada hormiga representa una solución completa (no necesariamente debe ser factible) al problema en cuestión y parte de una información inicial conocida.

Adicionalmente a los parámetros generales del problema y los asociados a los "Sistemas de Hormigas", se debe calcular el número de carretes por cada tipo de componente que son requeridos para completar con la producción requerida, siguiendo la rutina mencionada en la Sección 3.2.1.

Conociendo el número mínimo de carretes requeridos, los parámetros heurísticos y de rastros de feromonas asociados a cada decisión, la construcción de cada hormiga sigue el siguiente procedimiento iterativo:

- Seleccionar un tipo de boquilla que aún no haya sido utilizado en la solución.
- Determinar el conjunto de tipos de componentes que pueden ser instalados con la boquilla seleccionada y que no se les ha asignado posición.
- Asignar una posición disponible a cada carrete de cada tipo de componente que pertenece al conjunto ya mencionado.
- Actualizar las variables y, x y t.
- Actualizar las boquillas sin utilizar, los componentes sin instalar y las posiciones sin asignar.

Esta rutina se repite hasta que a todos los carretes de todos los componentes requeridos para completar una tarjeta se les haya asignado una posición.

Posteriormente se verifica que dadas las posiciones asignadas a los carretes, no se incumpla con la capacidad de ranuras estándar a cada lado de la máquina, se calcula los vectores de distancias y se calcula el valor de la función objetivo.

3.3.7 Toma de Decisiones

En la construcción de una hormiga, se deben tomar tres tipos de decisiones:

- Seleccionar una boquilla a utilizar.
- Seleccionar un componente a instalar.
- Asignar una posición a los carretes del tipo de componentes seleccionado.

La selección de una opción para cualquiera de las tres decisiones, se hace utilizando los fundamentos de los “Sistemas de Hormigas” y los “Sistemas de Colonias de Hormigas” que han sido mencionados previamente.

Con el propósito de explicar el mecanismo de selección, se ejemplifica a continuación la primera de las decisiones, es decir, la selección de una boquilla.

Se debe conocer el valor heurístico y el rastro de feromonas asociado a cada una de las opciones disponibles de boquillas, parámetros claves de decisión.

El valor heurístico y el rastro de feromonas, corresponden a dos *"rankings"* entre opciones. Mayores valores se asignan a las mejores decisiones, es decir, entre mayor sea el valor para una opción con respecto a las otras, mayor será la posibilidad de tomarla. Sin embargo, existe una explicación fundamental para justificar el manejo de dos *"rankings"* paralelamente.

El valor heurístico corresponde a una clasificación a priori (dadas las características del problema) que se puede hacer del posible beneficio de seleccionar una opción con respecto a las otras disponibles y que no varía en ningún momento del heurístico.

Por ejemplo, dado que se busca minimizar los cambios de herramientas, una característica que puede ayudar a clasificar las boquillas es el número de distintos tipos de componentes que pueden instalarse con cada uno, de tal forma, que la herramienta más atractiva sea aquella con el mayor valor heurístico. Este valor puede ser un valor absoluto o una proporción.

En tanto al rastro de feromonas se refiere, este es una clasificación que nace de la experiencia que se ha recogido en la construcción de las soluciones pasadas. Su valor no está asociado a que tan bueno puede ser o no una opción dadas las características del problema, sino que esta asociado a si la selección de cada opción ha llevado a producir buenas soluciones en el pasado. Entre más veces haya sido seleccionada una opción en las mejores soluciones encontradas, mayor será el valor de su rastro de feromonas.

Los parámetros α y β corresponden a una ponderación que se le da al rastro de feromonas y al valor heurístico respectivamente (posteriormente se justifica su selección).

Se presume que existe un conjunto J de opciones de boquillas a ser seleccionadas, y correspondiente un vector de valores heurísticos (h) y de rastros de feromonas (f) asociados a

cada componente del conjunto *J*. Es decir, para la componente *r* del conjunto *J*, su valor heurístico está dado por la componente *r* del vector *h* y el valor del rastro de feromonas es la componente *r* del vector *f*.

La componente *s* escogida dentro del conjunto de opciones *J* se selecciona siguiendo el procedimiento:

- Se genera un número aleatorio *q0* de una distribución U(0,1).
- Si *q0* es menor que el parámetro *q* (importancia relativa entre mejora de buenas soluciones encontradas y la búsqueda de nuevas) se escoge la opción más atractiva dada la información actual.

$$s=\begin{cases}\arg\max_{a\in J}\dfrac{(f_a^{\alpha})(h_a^{\beta})}{\sum_{b\in J}(f_b^{\alpha})(h_b^{\beta})} & \text{Si } q0 \le q. \\ S & \text{De lo contrario.}\end{cases} \qquad \text{Ec. 60}$$

- Si *q0* es mayor al valor de *q*, entonces se genera una nueva búsqueda y se selecciona al boquilla S siguiendo la función de distribución calculada con la Ec. 61.

$$p_a(t)=\frac{(f_a)^{\alpha}(h_a)^{\beta}}{\sum_{b\in J}(f_b)^{\alpha}(h_b)^{\beta}} \qquad \forall\, a \in J \qquad \text{Ec. 61}$$

3.3.8 Valores Heurísticos y Rastros de Feromonas

A continuación se presenta una descripción de los valores heurísticos y los rastros de feromonas para cada una de las decisiones.

Para la primera decisión, el valor heurístico es utilizado para clasificar las boquillas. Conociendo las características del problema y considerando que la penalización por el cambio de boquillas es grande, entre mayor sea el número de diferentes tipos de componentes que se pueden instalar con esa única herramienta, más deseable será su utilización. Para tal propósito el valor heurístico asociado a la selección de la boquilla (*hsn) j* es calculado como:

$$hsn_j = \frac{\sum_{i=1}^{nc} C_{ij}}{\max\{\sum_{i=1}^{nc} C_{if},\ f \in J\}}, \qquad \forall j \in J. \qquad \text{Ec. 62}$$

En cuanto al rastro de feromonas inicial (*fsn0)* de la boquilla *j* se refiere, este será igual para todos los tipos de boquillas y se calcula con la expresión:

$$fsn0_j = \frac{1}{Cardinalidad(J)}, \qquad \forall j \in J. \qquad \text{Ec. 63}$$

Para la segunda decisión (selección de un componente a instalar), el valor heurístico crea una prioridad para la asignación de órdenes. Es decir, si apenas se esta construyendo la solución, es deseable que se de la oportunidad de poder asignarle los mejores órdenes a los carretes con el tipo de componente más atractivos. Se considera que los componentes más atractivos para asignarle los mejores órdenes son aquellos que más son utilizados y justamente un buen valor a ser utilizado es el número de componentes que aporta cada tipo de componente. El valor heurístico para la selección del componente (*hsc) i* es:

$$hsc_i = b_i, \qquad \forall i \in I. \qquad \text{Ec. 64}$$

Dado que no se desea ingresar un sesgo inicial diferente al número de componentes utilizados de cada tipo, el rastro inicial de feromonas (*fsc0*) es igual para todos los tipos de componentes.

$$fsc0_i = \frac{1}{Cardinalidad(I)}, \qquad \forall i \in I. \qquad \text{Ec. 65}$$

Para la asignación de una posición a un tipo de componente, el valor heurístico (*hsr)* esta asociado simplemente al inverso del orden de importancia, es decir, entre más cercano se encuentre a la tabla de instalación este valor tenderá a uno.

$$hsr_l = \begin{cases} \frac{1}{l}, & 1 \le l \le W. \\ \frac{1}{l - W}, & W + 1 \le l \le 2W \end{cases} \qquad \text{Ec. 66}$$

En cuanto al rastro de feromonas se refiere, éste no está atado solamente al orden *l*, se asocia a cada una de las parejas (*i,l*) o (componente, orden). Este rastro refleja la preferencia que un carrete con componentes tipo *i*, sea asignado en el orden *l*. El rastro inicial (*fsr)* es calculado de tal forma qué se clasifique que tan deseado es el orden *l* si todos los componentes del tipo *i* deben ser instalados en dicho orden

$$fsr0_{i,l} = \begin{cases} \dfrac{1}{b_i l} & , \quad 1 \le l \le W, \forall i \in I. \\ \dfrac{1}{b_i (l - W)} & , \quad W + 1 \le l \le 2W, \forall i \in I. \end{cases} \qquad \text{Ec. 67}$$

3.3.9 Actualización de los Rastro de Feromonas

La actualización de feromonas es el mecanismo de recolección de información de las hormigas o soluciones que ya han sido generadas, constituyéndose en un punto crítico para la búsqueda de buenas soluciones utilizando los Sistemas de Hormigas.

Siguiendo algunas de las ideas de Dorigo (Dorigo et al., 2004), la actualización del rastro de feromonas sigue cuatro fundamentos básicos:

- El rastro de feromonas es constante para todas las hormigas de una misma colonia, es decir, no se hace una actualización al interior de una colonia.
- La tasa de evaporación del rastro de feromonas entre cada colonia es constante y se aplica a todos las componentes.
- Una vez que se ha generado una colonia, se incrementara el rastro de feromonas solamente en las componentes asociadas a las 5 mejores soluciones generadas.
- El aporte que hace cada una de estas 5 hormigas a la actualización del rastro, es proporcional a una clasificación entre ellas (entre mejor sea la solución, mayor será su aporte).

Dentro del proceso de implementación se ha tomado la decisión que los rastros asociados a la selección de una boquilla y de un orden para un tipo de componente son los que serán actualizados. En cuanto al rastro para la selección de un componente se mantiene constante para todas las colonias.

3.3.9.1 Rastro para Selección de una Boquilla

El procedimiento implementado para encontrar un valor de Δfsn es el siguiente.

- Se encuentran las 5 mejores soluciones factibles de la colonia.
- El valor máximo del rastro de feromonas para la selección de boquillas antes de la actualización esta dado por *maxfsn*.
- El mejor valor para la función objetivo encontrado es *fopt*, y el valor de la función para la hormiga *ant* esta dado por f_{ant}.
- Para cada boquilla, se verifica para cada una de las 5 hormigas si este fue utilizado o no. En caso de haberse utilizado la boquilla *j* por parte de la hormiga *ant* se actualiza la variable Δfsn_j de la forma:

$$\Delta fsn_j = \sum_{ant=1}^{5} \frac{fopt}{f_{ant}} \quad , \qquad \forall j \in J. \qquad \text{Ec. 68}$$

- Se estandariza dicho valor con respecto al *mfsn*.

$$\Delta fsn_j = \frac{\Delta fsn_j mfsn}{\arg\max_{e \in J} \Delta fsn_e} \quad , \qquad \forall j \in J. \qquad \text{Ec. 69}$$

- Se hace la actualización del rastro de feromonas.

$$fsn_j = fsn_j(1-\rho) + \Delta fsn_j \quad , \qquad \forall j \in J. \qquad \text{Ec. 70}$$

3.3.9.2 Rastro para Selección de una Posición para un Tipo de Componente

El procedimiento implementado para encontrar un valor de Δfsr es el siguiente.

- Se encuentran las 5 mejores soluciones factibles de la colonia.
- El valor máximo del rastro de feromonas para la selección de boquillas antes de la actualización esta dado por *mfsr*.
- El menor valor para la función objetivo encontrado es *fopt*, y el valor de la función para la hormiga *ant* esta dado por f_{ant}.
- Para cada combinación de componente *i* y orden *l*, se verifica para cada una de las 5 hormigas si esta fue utilizado o no. En caso de haberse asignado el orden *l* a un

carrete con componentes tipo *i* por parte de la hormiga *ant* se actualiza la variable Δfsr_{il} de la forma:

$$\Delta fsr_{il} = \sum_{ant=1}^{5} \frac{fopt}{f_{ant}} \quad , \qquad \forall i \in I, l \in L. \qquad \text{Ec. 71}$$

- Se estandariza dicho valor con respecto al *maxfsn*.

$$\Delta fsr_{il} = \frac{\Delta fsr_{il} mfsr}{\arg\max_{e \in I, f \in l} \Delta fsr_{ef}} \quad , \quad \forall i \in I, l \in L. \qquad \text{Ec. 72}$$

- Se hace la actualización del rastro de feromonas.

$$fsr_{il} = fsr_{il}(1-\rho) + \Delta fsr_{il} \quad , \quad \forall i \in I, l \in L. \qquad \text{Ec. 73}$$

3.4 Descripción de los Heurísticos

Continuando con la idea de presentar dos versiones del problema de optimización, dada la composición de la función objetivo y la relevancia que tiene el valor de penalización que se le asigne al cambio de boquilla, se han desarrollado dos implementaciones de los "Sistemas de Hormigas".

El Caso 1 busca minimizar el tiempo de las operaciones de recogido sin tener en cuenta la utilización de boquillas y el Caso 2 contempla la asignación de boquillas y carretes simultáneamente.

3.4.1 Caso 1 – Modelo sin Boquillas

A continuación se presenta un seudo código del algoritmo que fue desarrollado para el Caso 1. El programa completo fue desarrollado en MATLAB y se presenta en el Apéndice #1.

Definir los datos de entrada.

b, v, w, L, W, Z, N, C, nt, K — *Sección 3.3.3.*

Definir los parámetros para los Sistemas de Hormigas.

Rep, Col, Hor, Alpha, Beta, Ro, q. — *Sección 3.3.4.*

Definir variables del modelo.

x, y — *Sección 3.3.5.*

Inicializar los valores heurísticos y los rastros de feromonas.

hsc *Ec. 64.*

fsc0 *Ec. 65.*

hsr *Ec. 66.*

fsr0 *Ec. 67.*

Determinar el número de carretes necesarios de cada tipo, verificación de factibilidad al comprobar si las ranuras disponibles son suficientes y calcular el total de tarjetas que se pueden completar.

Comienza ciclo de réplicas.

Inicialización de los rastros de feromonas.

Comienza ciclo de generación colonias.

Comienza ciclo de generación de hormigas.

Se inicializa un vector con los componentes sin asignar.

Se inicializa un vector con los órdenes sin asignar.

Mientras el vector de componentes no este vació.

Seleccionar un componente i. *Sección 3.3.7.*

Actualizar el vector de componentes sin instalar.

Seleccionar un orden l a cada uno carretes de los requeridos del componente tipo i. *Sección 3.3.7.*

Actualizar el vector de posiciones disponibles.

Mejorar la solución al asegurarse que los órdenes asignados sean consecutivos.

Se actualizan las variables x, y, d dada la configuración encontrada.

Se calcula el valor de la función objetivo.

Calcular las mejores 5 hormigas generadas.

Actualizar el rastro para selección de orden. *Sección 3.3.9.1*

Aún cuando este seudo código refleja un caso general, puede ser estudiado bajo dos perspectivas. La primera de ellas es cuando los carretes tienen un único tamaño y requieren solamente de una ranura estándar, de tal forma que *w* es un vector de unos. El segundo refleja

mejor la realidad y es cuando los carretes tienen diferentes tamaños. Dado que la formulación pretende asignar un orden en una secuencia y éste no se asocia directamente al número de la ranura en la cual se colocará el carrete, vale la pena como trabajo futuro, realizar un estudio separado de estas dos variantes y verificar si los resultados obtenidos en cuenta a la mejor configuración son diferentes (La diferencia entre los dos radica en los coeficientes de la función objetivo).

Con el propósito de hacer más claro el heurístico propuesto y tomando como referencia la tarjeta caracterizada en la sección 3.2.3, se presentan a continuación los pasos iniciales del heurístico.

Definir los datos de entrada.

b = [12, 18, 54, 24].

v = [3000, 3000, 5000, 3000].

w = [3, 1, 1, 2].

L = 14.

W = 7.

Z = 5.

N = 200.

nt = 15.

K = 2.

C = [1 0 1 1 0
0 0 1 1 0
1 0 0 0 0
0 1 1 1 1]

Definir los parámetros para los Sistemas de Hormigas.

Rep = 1

Col = 20

Hor = 40

Alpha = 1.5

Beta = 0.5

Ro = 0.25

$q = 0.25$

Inicializar los valores heurísticos y los rastros de feromonas.

hsc = [12, 18, 54, 24]

$fsc0$ = [0.25, 0.25, 0.25, 0.25]

hsr = [1, 0.5, 0.33, 0.25, 0.2, 0.16, 0.14, 1, 0.5, 0.33, 0.25, 0.2, 0.16, 0.14]

$fsr0^T$ = [

0.0833	0.0556	0.0185	0.0417
0.0417	0.0278	0.0093	0.0208
0.0278	0.0185	0.0062	0.0139
0.0208	0.0139	0.0046	0.0104
0.0167	0.0111	0.0037	0.0083
0.0139	0.0093	0.0031	0.0069
0.0119	0.0079	0.0026	0.0060
0.0833	0.0556	0.0185	0.0417
0.0417	0.0278	0.0093	0.0208
0.0278	0.0185	0.0062	0.0139
0.0208	0.0139	0.0046	0.0104
0.0167	0.0111	0.0037	0.0083
0.0139	0.0093	0.0031	0.0069
0.0119	0.0079	0.0026	0.0060

]

Determinar el número de carretes necesarios de cada tipo, verificación de factibilidad al comprobar si las ranuras disponibles son suficientes y calcular el total de tarjetas que se pueden completar.

El problema es factible.

El número de tarjetas que pueden ser completadas es 250.

El vector con el número de carretes requeridos por cada tipo de componentes es:

a = [1, 2, 3, 2]

Comienza ciclo de réplicas.

Se comienza la primera réplica

r = 1

Inicialización de los rastros de feromonas.

fsc = fsc0

fsr = fsr0

Comienza ciclo de generación colonias.

Se comienza la primera colonia

Co = 1

Comienza ciclo de generación de hormigas.

Se genera la primera hormiga

Ant = 1

Se inicializa un vector con los componentes sin asignar.

components = [1, 2, 3, 4]

Se inicializa un vector con los posiciones sin asignar.

posicion = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]

Mientras el vector de componentes no este vació.

Seleccionar un componente i .

q0 = 0.803 (U(0,1)).

q0 < q

Se selecciona un componente siguiendo la Ec. 60.

El componente seleccionado (*i*) es 3.

a(3) = 3.

Actualizar el vector de componentes sin instalar.

components = [1, 2, 4]

Seleccionar una posición para los tres carretes del componente tipo 3.

Para el primer carrete.

q0 = 0.1153 (U(0,1)).

q0 < q

Se asigna posición siguiendo la Ec. 60.

La posición asignada (*l*) es 1.

Para el segundo carrete.

q0 = 0.4794 (U(0,1)).

q0 > q

Se asigna posición siguiendo la Ec. 61.

La posición asignada (*l*) es 2.

Para el tercer carrete.

q0 = 0.2684 (U(0,1)).

q0 > q

Se asigna posición siguiendo la Ec. 61.

La posición asignada (*l*) es 8.

Actualizar el vector de posiciones disponibles.

posicion = [3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14]

Se continúa hasta que se le haya asignado un orden a todos los carretes requeridos.

La asignación realizada por esta primera hormiga es:

Tipo de Carrete	Orden Asignado	Unidades por Tarjeta
3	1	18
3	2	18
4	3	12
4	4	12
2	7	9
3	8	18
2	11	9
1	12	12

Mejorar la solución al asegurarse que los órdenes asignados sean consecutivos.

Se actualizan las variables x, y, d dada la configuración encontrada.

Se calcula el valor de la función objetivo.

Se presenta el resultado final para la primera hormiga.

Tipo de Carrete	Orden Asignado	Unidades por Tarjeta	*w(i)*	Distancia
3	1	18	1	1
3	2	18	1	2
4	3	12	2	4
4	4	12	2	6
2	5	9	1	7
3	8	18	1	1
2	9	9	1	2
1	10	12	3	5
		Función Objetivo		**333**

Calcular las mejores 5 hormigas generadas.

Se encuentran las 5 mejores soluciones del total de 40 generadas.

Réplica	Colonia	Hormiga	Valor Función Objetivo
1	1	35	315
1	1	1	333
1	1	13	333
1	1	7	336
1	1	10	336

Actualizar el rastro para selección de orden.

Se calcula un valor de Δfsr_{il} para cada pareja (i,l) siguiendo las Ecuaciones 71 y 72. La matriz resultante es igual a:

$$\Delta fsr^T = \begin{bmatrix} 0.000 & 0.000 & 0.078 & 0.000 \\ 0.078 & 0.083 & 0.078 & 0.000 \\ 0.078 & 0.078 & 0.000 & 0.079 \\ 0.000 & 0.078 & 0.000 & 0.078 \\ 0.000 & 0.079 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.078 & 0.000 \\ 0.000 & 0.079 & 0.078 & 0.078 \\ 0.079 & 0.078 & 0.000 & 0.078 \\ 0.083 & 0.078 & 0.000 & 0.078 \\ 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 \end{bmatrix}$$

Utilizando la Ec.73 se actualiza el rastro de feromonas y se continúa con la siguiente colonia.

Los mejores resultados encontrados por cada colonia se presentan a continuación:

Réplica	Colonia	Hormiga	Valor Función Objetivo
1	1	35	315
1	2	5	309
1	3	4	309
1	4	3	309
1	5	4	309
1	6	2	309
1	7	3	309
1	8	5	309
1	9	1	309
1	10	1	309
1	11	1	309
1	12	1	309
1	13	1	309
1	14	1	309
1	15	1	309
1	16	1	309
1	17	1	309
1	18	1	309
1	19	1	309
1	20	1	309

Es importante recalcar que este valor 309 fue encontrado en un tiempo de ejecución de 4 segundos, comparado con los 70 minutos que le tomo a Lingo 8.0 en encontrar una solución factible, con el mismo valor en la función objetivo.

3.4.2 Caso 2 – Modelo con Boquillas

La existencia de una limitación en el espacio de almacenamiento de las herramientas de instalación, al igual que el tiempo asociado a los cambios de las mismas, hace necesario el incorporar estas características en una buena solución.

Previamente se había mencionado que la penalización por el cambio de boquillas es grande, lo que hace que el enfoque en la implementación es diferente: la primera decisión que se toma es el seleccionar una herramienta atractiva, para después asignar todos los

carretes con tipos de componentes que puedan ser instalados con está, continuando hasta que todos hayan sido asignados.

Movido por la misma motivación, también se puede dividir en los dos casos particulares: cuando los carretes ocupan el mismo número de ranuras y cuando tienen diferentes tamaños.

El algoritmo fue implementado en MATLAB y se encuentra disponible en el Apéndice #2.

Definir los datos de entrada.

b, v, w, L, W, Z, N, C, nt, K *Sección 3.3.3.*

Definir los parámetros para los Sistemas de Hormigas.

Rep, Col, Hor, Alpha, Beta, Ro, q. *Sección 3.3.4.*

Definir variables del modelo.

x, y, t *Sección 3.3.5.*

Inicializar los valores heurísticos y los rastros de feromonas.

hsn *Ec. 62.*

fsn0 *Ec. 63.*

hsc *Ec. 64.*

fsc0 *Ec. 65.*

hsr *Ec. 66.*

fsr0 *Ec. 67.*

Determinar el número de carretes necesarios de cada tipo, verificación de factibilidad al comprobar si las ranuras disponibles son suficientes y calcular el total de tarjetas que se pueden completar.

Comienza ciclo de réplicas.

Inicialización de los rastros de feromonas.

Comienza ciclo de generación colonias.

Comienza ciclo de generación de hormigas.

Se inicializa un vector con las boquillas que no se han utilizado.

Se inicializa un vector con los componentes sin asignar.

Se inicializa un vector con los órdenes sin asignar.

Mientras el vector de componentes no este vació.

Selección de la boquilla j a utilizar. *Sección 3.3.7.*

Seleccionar un componente i que pueda ser instalado con este tipo de herramienta. *Sección 3.3.7.*

Actualizar el vector de componentes sin instalar.

Seleccionar un orden l a cada uno carretes de los requeridos del componente tipo i. *Sección 3.3.7.*

Actualizar el vector de órdenes disponibles.

Mejorar la solución al asegurarse que los órdenes asignados sean consecutivos.

Se actualizan las variables x, y, t, d, dada la configuración encontrada.

Se calcula el valor de la función objetivo.

Calcular las mejores 5 hormigas generadas.

Actualizar el rastro para selección de boquilla. *Sección 3.3.9.1*

Actualizar el rastro para selección de orden. *Sección 3.3.9.2*

Tomando como referencia la tarjeta caracterizada en la sección 3.2.3, se presentan a continuación los pasos iniciales del heurístico previamente descrito y que soluciona el problema completo que es estudiado en este trabajo de tesis

Definir los datos de entrada.

b = [12, 18, 54, 24].

v = [3000, 3000, 5000, 3000].

w = [3, 1, 1, 2].

L = 14.

W = 7.

Z = 5.

N = 200.

nt = 15.

K = 2.

C = [1 0 1 1 0

0 0 1 1 0
1 0 0 0 0
0 1 1 1 1]

Definir los parámetros para los Sistemas de Hormigas.

Rep = 1

Col = 20

Hor = 40

Alpha = 1.5

Beta = 0.5

Ro = 0.25

q = 0.25

Inicializar los valores heurísticos y los rastros de feromonas.

hsc = [12, 18, 54, 24]

fsc0 = [0.25, 0.25, 0.25, 0.25]

hsr = [1, 0.5, 0.33, 0.25, 0.2, 0.16, 0.14, 1, 0.5, 0.33, 0.25, 0.2, 0.16, 0.14]

$fsr0^T$ = [0.083 0.056 0.019 0.042
0.042 0.028 0.009 0.021
0.028 0.019 0.006 0.014
0.021 0.014 0.005 0.010
0.017 0.011 0.004 0.008
0.014 0.009 0.003 0.007
0.012 0.008 0.003 0.006
0.083 0.056 0.019 0.042
0.042 0.028 0.009 0.021
0.028 0.019 0.006 0.014
0.021 0.014 0.005 0.010
0.017 0.011 0.004 0.008
0.014 0.009 0.003 0.007
0.012 0.008 0.003 0.006]

hsn = [0.66, 0.33, 1, 1, 0.33]

fsn0 = [0.2, 0.2, 0.2, 0.2, 0.2]

Determinar el número de carretes necesarios de cada tipo, verificación de factibilidad al comprobar si las ranuras disponibles son suficientes y calcular el total de tarjetas que se pueden completar.

El problema es factible.

El número de tarjetas que pueden ser completadas es 250.

El vector con el número de carretes requeridos por cada tipo de componentes es:

$a = [1, 2, 3, 2]$

Comienza ciclo de réplicas.

Se comienza la primera réplica

r = 1

Inicialización de los rastros de feromonas.

$fsn = fsn0$

$fsc = fsc0$

$fsr = fsr0$

Comienza ciclo de generación colonias.

Se comienza la primera colonia

Co = 1

Comienza ciclo de generación de hormigas.

Se genera la primera hormiga

Ant = 1

Se inicializa un vector con las boquillas sin utilizar.

nozzles = [1, 2, 3, 4, 5]

Se inicializa un vector con los componentes sin asignar.

components = [1, 2, 3, 4]

Se inicializa un vector con los posiciones sin asignar.

posicion = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]

Mientras el vector de componentes no este vació.

Selección de la boquilla j a utilizar.

$q0 = 0.9372$ (U(0,1)).

Dado que el aleatorio es mayor que q se selecciona una boquilla siguiendo la Ec. 61.

La boquilla seleccionada (j) = 4.

El conjunto de componentes que pueden instalarse con esta herramienta esta dado por component_nozzle

component_nozzle = [1, 2, 4]

Se actualiza el conjunto de boquillas sin utilizar.

nozzles = [1, 2, 3, 5]

Seleccionar un componente i que pueda ser instalado con este tipo de herramienta.

q0 = 0.3894 (U(0,1)).

Dado que el aleatorio es mayor que *q* se selecciona un componente siguiendo la Ec. 61.

El componente seleccionado (*i*) = 1.

a(1) = 1.

Actualizar el vector de componentes sin instalar con esta boquilla

Component_nozzle = [2, 4]

Seleccionar un orden l a cada uno carretes de los requeridos del componente tipo i.

Para el único carrete.

q0 = 0.6735 (U(0,1)).

Dado que el aleatorio es mayor que *q* se selecciona el mejor orden siguiente la Ec. 61.

El orden asignado es (*l*) = 3.

Actualizar el vector de órdenes disponibles.

orden = [1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]

Se continúa hasta que se le haya asignado un orden a todos los carretes de los componentes que se puedan instalar con la herramienta *j*.

Actualizar el vector de componentes sin instalar.

components = [3]

Si faltan componentes por instalar se escoge una nueva herramienta y se asigna orden a todos los componentes asociados a ella, hasta que no queden

componentes. Es posible que no se requiera la utilización de todas las boquillas.

La asignación realizada por esta primera hormiga es:

Tipo de Carrete	Orden Asignado	Unidades por Tarjeta
4	1	12
3	2	18
1	3	12
4	4	12
2	8	9
3	9	18
2	10	9
3	13	18

Mejorar la solución al asegurarse que los órdenes asignados sean consecutivos. Se actualizan las variables x, y, t, d, dada la configuración encontrada. Se calcula el valor de la función objetivo.

Se presenta el resultado final para la primera hormiga

Posición	Tipo de Componente	Unidades por Tarjeta	Distancia
1	4	12	2
2	3	18	3
3	1	12	6
4	4	12	8
8	2	9	1
9	3	18	2
10	2	9	3
11	3	18	4
	Distancia Total Recorrida		390

"Nozzle"	Lado Izquierdo	Lado Derecho	Total
1	1	1	2
2	0	0	0
3	0	0	0
4	1	1	2
5	0	0	0
		Cambios de *"Nozzles"*	4
		Valor Penalización	60
		Función Objetivo	450

Con un total de 4 cambios de boquillas, asociado a tener boquillas 1 y 4 a cada lado de la máquina. El valor de la función objetivo por tanto es:

Función Objetivo = 390 + 60 = 450

Calcular las mejores 5 hormigas generadas.

Réplica	Colonia	Hormiga	Valor Función Objetivo
1	1	11	396
1	1	8	399
1	1	38	423
1	1	13	426
1	1	26	441

Actualizar el rastro para selección de boquillas.

Se calcula un valor de Δfsn_j para cada boquilla siguiendo las Ecuaciones 68 y 69. El vector resultante es:

Δfsn = [0.58, 0, 0.58, 0, 0]

Utilizando la Ec.70 se actualiza el rastro de feromonas y se continúa con la siguiente colonia.

Actualizar el rastro para selección de orden.

Se calcula un valor de Δfsr_{il} para cada pareja (*i*,*l*) siguiendo las Ecuaciones 71 y 72. La matriz resultante es igual a:

$$\Delta fsr^T = \begin{bmatrix} 0.000 & 2.082 & 2.082 & 0.000 \\ 0.000 & 0.000 & 2.082 & 2.082 \\ 2.082 & 0.000 & 0.000 & 0.000 \\ 0.000 & 2.098 & 0.000 & 2.115 \\ 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 2.082 & 2.082 & 0.000 \\ 0.000 & 2.098 & 2.082 & 0.000 \\ 0.000 & 2.082 & 0.000 & 2.082 \\ 0.000 & 0.000 & 2.082 & 2.082 \\ 0.000 & 2.082 & 0.000 & 2.082 \\ 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 \end{bmatrix}$$

Utilizando la Ec.73 se actualiza el rastro de feromonas y se continúa con la siguiente colonia.

Los mejores resultados encontrados por cada colonia se presentan a continuación:

Réplica	Colonia	Hormiga	Valor Función Objetivo
1	1	11	396
1	2	39	390
1	3	13	390
1	4	33	375
1	5	39	378
1	6	32	390
1	7	11	381
1	8	30	402
1	9	27	369
1	10	17	375
1	11	12	369
1	12	10	390
1	13	8	375
1	14	5	381
1	15	37	384
1	16	11	378
1	17	12	381
1	18	12	381
1	19	4	375
1	20	32	381

Es importante recalcar que la mejor solución de 369 fue encontrado en un tiempo de ejecución de 6 segundos, comparado con la hora que le tomo a Lingo 8.0 en encontrar una solución no factible, con un valor de función objetivo de 360 (2.5% menor, pero no falible).

4 Resultados

Una vez descritos los dos heurísticos que han sido desarrollados para encontrar una secuencia óptima de configuración para la máquina Fuji IP III, es necesario identificar los parámetros asociados a los "Sistemas de Hormigas" que influyen en la variable de respuesta, al igual que el determinar las combinaciones de estos que se recomiendan utilizar al producir tarjetas con características similares a las que actualmente se ensamblan en la Fábrica Modelo. La primera parte de esta Capítulo se centra en mostrar estos resultados.

4.1 Resultados

4.1.1 Implementación del Heurístico en una Tarjeta de la Fábrica Modelo

La tarjeta modelo que fue utilizada en el Capítulo anterior es muy pequeña y los resultados obtenidos no puede ser tomados como un punto real de la efectividad del heurístico. Es necesario validar el modelo con datos más complejos que se ajusten a la realidad. Utilizando los datos de una de las tres tarjetas que actualmente se producen en la Fábrica Modelo, se pretende encontrar buenos valores para los parámetros del heurístico, de tal forma que se puedan garantizar buenas soluciones a problemas con características similares.

Por razones de confidencialidad los nombres reales de los tipos de componentes al igual que de las boquillas utilizadas no son presentados en este proyecto de tesis, no obstante la omisión de esta información no tiene relevancia en cuanto a los resultados que se presentan.

La tarjeta tiene un total de 34 diferentes tipos de componentes y se va a asumir un requerimiento de producción de 200 tarjetas bajo una misma configuración. El número de unidades requeridas de cada componente (b_i), el número de unidades almacenadas en cada carrete (v_i) y el número de ranuras estándar que ocupa cada uno (w_i), se presentan en la Tabla 6. La máquina tiene un total de 37 ranuras estándar (W) y 6 lugares para el almacenamiento de boquillas (Z) a cada lado de la mesa de instalación. El valor de la penalización por el

cambio de las herramientas de instalación (*nt*) es igual a 80 (Nótese que este valor es mayor a 2W).

Tabla 6 - Tarjeta de la Fábrica Modelo

Tipo de Componente	Unidades por Carrete	Unidades por Tarjeta	Ranuras Estándar
1	3,000	12	1
2	3,000	18	1
3	4,000	6	1
4	5,000	54	1
5	5,000	6	1
6	4,000	24	1
7	5,000	6	1
8	3,000	24	1
9	5,000	6	1
10	5,000	6	1
11	1,000	6	2
12	2,500	6	1
13	4,000	6	1
14	4,000	6	1
15	4,000	6	1
16	4,000	6	1
17	4,000	6	1
18	3,000	6	1
19	4,000	6	1
20	5,000	6	1
21	5,000	6	2
22	3,000	30	1
23	5,000	54	1
24	2,500	6	2
25	3,300	6	2
26	2,500	6	2
27	2,500	6	2
28	5,000	18	1
29	500	6	2
30	2,500	12	2
31	3,000	6	2
32	2,000	6	2
33	1,000	6	3
34	960	6	3

Se tienen un total de 5 diferentes tipos de boquillas disponibles y la matriz *C* de correspondencias con los diferentes tipos de componentes se presenta en la Tabla 7.

Tabla 7 - Matriz *C* de Tarjeta Fábrica Modelo

Tipo de Componente	Tipo de *"Nozzle"* 1	2	3	4	5
1	1	0	1	1	0
2	0	0	1	1	0
3	1	1	1	0	0
4	1	0	0	0	0
5	1	0	0	1	1
6	0	1	0	0	0
7	0	0	1	1	0
8	0	1	1	1	1
9	0	1	0	0	1
10	1	1	1	0	0
11	0	1	0	0	0
12	1	0	0	0	1
13	1	0	1	0	0
14	0	0	0	1	0
15	1	1	0	1	0
16	1	1	0	0	0
17	0	1	1	1	0
18	0	1	0	0	1
19	1	1	0	0	0
20	0	1	1	1	1
21	0	1	1	1	1
22	0	0	1	0	1
23	0	1	1	0	0
24	0	1	1	1	1
25	0	0	1	0	1
26	1	1	0	0	1
27	0	0	0	1	0
28	0	1	0	1	0
29	0	1	1	0	0
30	1	1	1	1	1
31	1	0	0	0	1
32	0	1	0	0	1
33	1	1	0	0	1
34	0	0	1	0	1

4.1.2 Diseño de los Experimentos

El heurístico desarrollado tiene 7 parámetros de entrada, donde no necesariamente todos ellos inciden directamente en su comportamiento, de acuerdo a lo expresado por Dorigo, Maniezzo y Colorni (2006).

Las características de los problemas, puede hacer posible que ciertos parámetros no tengan influencia en la variables de respuesta. Por ejemplo, si se esta trabajando con un problema muy pequeño, es posible que al tener un número muy grande hormigas, colonias o réplicas (se generan muchas hormigas y prácticamente se enumeran todas las soluciones factibles), sea suficiente para generar buenas soluciones sin importar los valores seleccionados; pero esto es algo que se desconoce para el problema en cuestión.

El propósito de plantear los siguientes experimentos es el determinar cuales de estos parámetros y cuales de sus interacciones influyen en los resultados obtenidos por el heurístico, con esta información estimar ecuaciones que predigan el comportamiento de las variables de respuesta como función de los parámetros significativos, de tal forma que puedan ser utilizadas para determinar los parámetros que optimizan la respuesta.

Conjuntamente con una buena inicialización de los rastros de feromonas y los valores heurísticos, la adecuada selección de estos parámetros es clave al momento de generar mejores soluciones a medida que aumenta el número de colonias generadas, con lo cual aumenta la probabilidad que la mejor solución factible encontrada, no este muy distante de la solución óptima del problema.

Considerando la característica heurística de los algoritmos y la generación de múltiples soluciones factibles en cada corrida del experimento, el utilizar solamente la mejor solución factible como punto de referencia para la selección de parámetros de entrada no es lo más recomendable. Es posible que la mejor solución para una condición experimental (combinación de parámetros de entrada) corresponda a un punto no representativo del comportamiento de las soluciones factibles generadas por dicha condición. Es decir, que por la aleatoriedad propia del heurístico, unas pocas soluciones sean muy buenas y que representen un porcentaje muy bajo con respecto al total de la población de soluciones factibles generadas.

El objetivo es el seleccionar parámetros de entrada que puedan ayudar a generar soluciones consistentemente, donde la probabilidad de encontrar buenas configuraciones sea lo más alta posible. La bondad de la utilización de múltiples respuestas asociadas a una

distribución empírica, radica en la posibilidad de estimar parámetros que no solamente buscan minimizar la mejor solución encontrada, sino todos los datos que se generan en cada condición experimental y con la cual se construye dicha distribución.

No solamente se estudia como variable de respuesta la mejor solución factible, sino que se han organizado ascendentemente los valores encontrados para la función objetivo en cada condición experimental, con el fin de construir una función de distribución empírica y estimar las percentilas 90, 75 y 50. Tomando como referencia el mejor valor encontrado como la percentila 100, se ha de encontrar valores límites para los cuales el 10%, 25% y 50% de los resultados factibles sean menores (percentilas 90, 75 y 50). La cercanía de estos puntos a la mejor solución encontrada es un indicio de consistencia de las soluciones, donde la probabilidad de encontrar buenas soluciones es alta.

Pero la selección de estos parámetros no solamente influye en el resultado final que se pueda obtener, sino también lo hace en el tiempo que invierte el heurístico en encontrar una buena solución y en la variabilidad de los resultados obtenidos. No hay que olvidar que la motivación principal para esta implementación, es proporcionar en un tiempo razonable buenas soluciones a este problema, que dada sus características combinatorias, el encontrar la solución óptima toma demasiado tiempo. Por tal motivo es agrega también como variable de respuesta el tiempo de ejecución.

En resumen, la selección de buenos parámetros debe ir encaminada a la generación de los mejores valores para la función objetivo y en disminuir el tiempo de ejecución del procedimiento.

Los 7 parámetros asociados al heurístico en cuestión son:

- Número de Réplicas.
- Número de Colonias por réplica.
- Número de Hormigas por colonia.
- El valor Alpha de importancia del rastro de feromonas.

- El valor Beta de importancia del valor heurístico.
- El valor Ro asociado a la tasa de evaporación del rastro de feromonas.
- El valor q asocia a la mejora de buenas soluciones frente a la búsqueda de nuevas.

Se ha decidido utilizar un diseño factorial 2^7 (Montgomery, 2006), donde los niveles altos y bajos asociados a cada factor han sido seleccionados siguiendo dos consideraciones: la primera asociada al conjunto de posibles valores que pueden tomar, mientras que la segunda corresponde a la experiencia que se ha obtenido al manipular el heurístico tomando como referencia el valor de la función objetivo.

Se han de considerar 6 variables de respuesta para cada condición experimental: mejor valor (percentila 100), las percentilas 90, 75 y 50, el logaritmo natural de la desviación de las soluciones factibles encontradas y el tiempo de ejecución en segundos.

Para el factor *Réplicas* se ha definido como nivel bajo 1 y como nivel alto 5. El comportamiento de las mejores respuestas para cada colonia es muy similar entre réplicas: todos parecen ajustarse a una misma curva descendente, por lo que a primera vista se puede pensar que las réplicas no tienen un efecto en el valor de la función objetivo (percentil 100). No obstante, para involucrar su efecto en el modelo, se considera que 1 y 5 son buenos valores.

Se ha seleccionado 10 y 20 como los niveles bajo y alto para el factor *Colonias*. La experiencia ha ayudado a identificar que cuando se generan varias colonias, se presenta una disminución pronunciada en el valor de la función objetivo hasta que se genera en promedio la colonia 10. La mejora en la función desde esta colonia hasta la 20 aproximadamente, es menor. La mejora en la función al generar más de 20 colonias, desde el punto de vista gráfico parece no ser significativa.

En cuanto a la selección del número de *Hormigas*, el parámetro de decisión ha sido aún más intuitivo. La selección del nivel bajo en 40 nace de una observación gráfica, en la cual se verifica que tal número de hormigas introducen la suficiente variabilidad en el modelo

estudiado, es decir, el rango de soluciones factibles que se pueden encontrar es bastante amplio. El nivel alto de 80 es el resultado de duplicar el nivel bajo.

Para la selección de los niveles para *Alpha*, *Beta*, *Ro* y *q*, la decisión se fundamenta en las características particulares de cada uno.

Alpha y *Beta* son parámetros de importancia para el rastro de feromonas y el valor heurístico, el valor bajo se ha fijado en 0.5 y el nivel alto en 1.5 para los dos. En cuanto a *Ro* y *q*, estos corresponden a proporciones y los valores bajo y alto son 0.25 y 0.75 respectivamente.

Si se quisiera completar una réplica completa del diseño factorial, se requerirían un total de 128 experimentos, un número grande en cuanto a recursos de computadoras y tiempo de ejecución requiere. Haciendo uso de la herramienta de los diseños fraccionarios (Montgomery, 2006), se ha de completar una réplica de ¼ del diseño factorial 2^7 (el número de condiciones experimentales disminuye a 32).

Es necesario tener en cuenta que al utilizar una fracción, el aporte de factores y sus interacciones estarán fundidos entre ellos: existirán grupos de factores e interacciones que estarán representadas en un único efecto, por lo cual el número de factores e interacciones que se pueden estimar es menor al total de posibles combinaciones.

Tomando en cuenta que los factores que se pueden estimar dependen de los generadores que se utilicen para diseñar el experimento factorial (Montgomery, 2006), se ha seleccionado aquellos que permiten diferenciar los factores que intuitivamente deben influir en la variable de respuesta.

El diseño del experimento permite que se pueda estimar el efecto de los siguientes factores principales e interacciones de segundo orden:

- Replicas, Colonias, Hormigas, Alpha, Beta, Ro, q.
- Replicas – Colonias, Replicas – Hormigas, Replicas – Alpha, Replicas – Beta, Replicas – Ro, Replicas – q, Colonias – Hormigas, Colonias – Alpha, Colonias – Beta,

Colonias – Ro, Colonias – q, Hormigas – Alpha, Hormigas – Beta, Hormigas – Ro, Hormigas – q, Alpha – Beta, Alpha – Ro, Alpha – q.

4.1.3 Resultados de los Experimentos

Con el propósito de determinar cuales de los factores y sus interacciones son representativas para explicar el comportamiento de la función objetivo, se implementó el procedimiento de *"Stepwise"* (Montgomery, 2006). Es un procedimiento iterativo que con la incorporación y extracción sistemática de factores e interacciones al modelo, ayuda a determinar aquellos que explican el comportamiento de la variable de respuesta.

Utilizando Minitab 14, se implementó dicho procedimiento para cada una de las 6 variables de respuesta en cada uno de los dos casos planeados (sin boquillas y con boquillas).

Partiendo de los resultados obtenidos por *"Stepwise"*, para cada variable de respuesta se estimaron modelos de regresión para determinar ecuaciones para cada variable como función de factores e interacciones. Se ha tomado como referencia un error Tipo I de 5%.

4.1.4 Resultados Caso 1 – Sin Boquillas

A continuación se presentan las gráficas de las funciones de distribución de probabilidad calculas para cada una de las 32 condiciones experimentales, donde se logra apreciar la variabilidad en los resultados de acuerdo a los parámetros que se seleccionen. Las gráficas se han construido de tal forma que se presentan grupos de 8 condiciones experimentales (se tiene en cuenta el orden estándar dentro del experimento) que tienen en común los valores Alpha y Beta. Es importante mencionar las convenciones expresadas en las gráficas (R= Réplicas, C = Colonias y H= Hormigas).

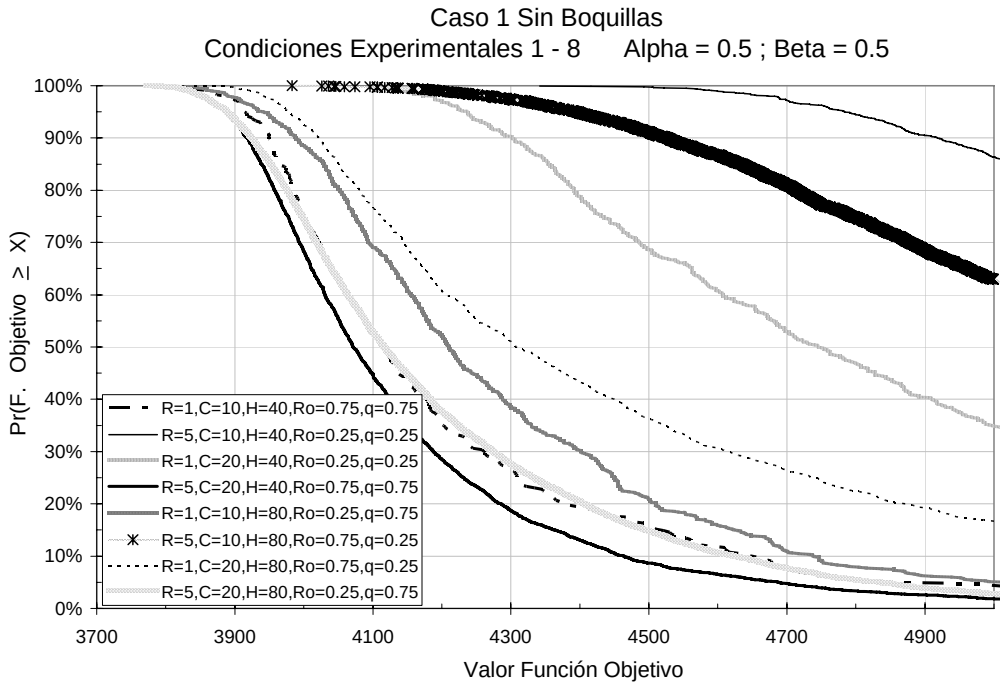


Figura 10 - Experimentos 1 a 8

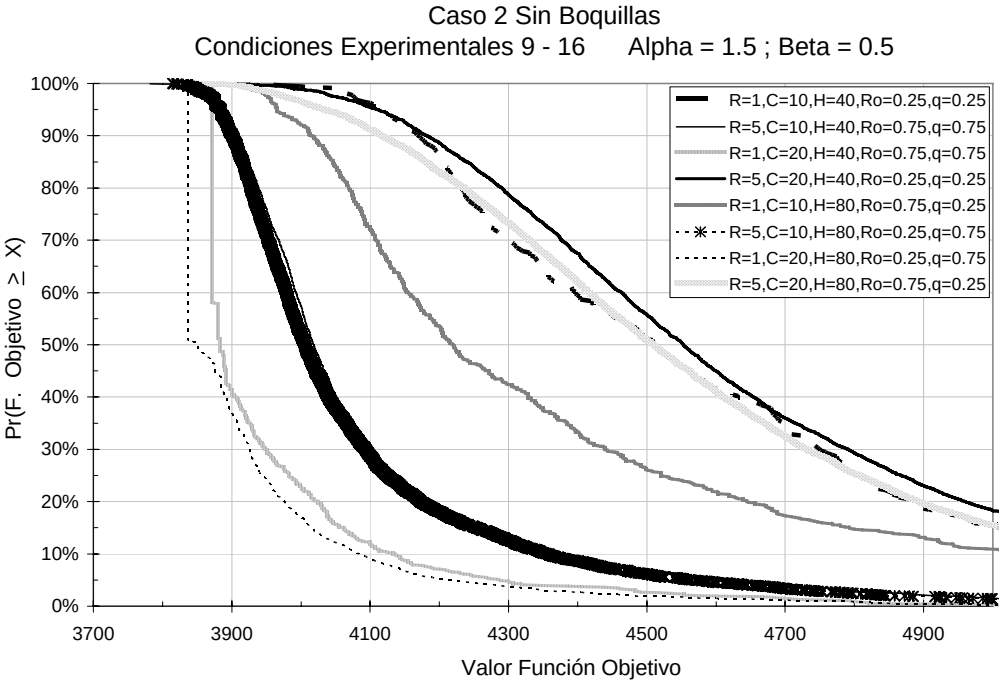


Figura 11 – Experimentos 9 a 16

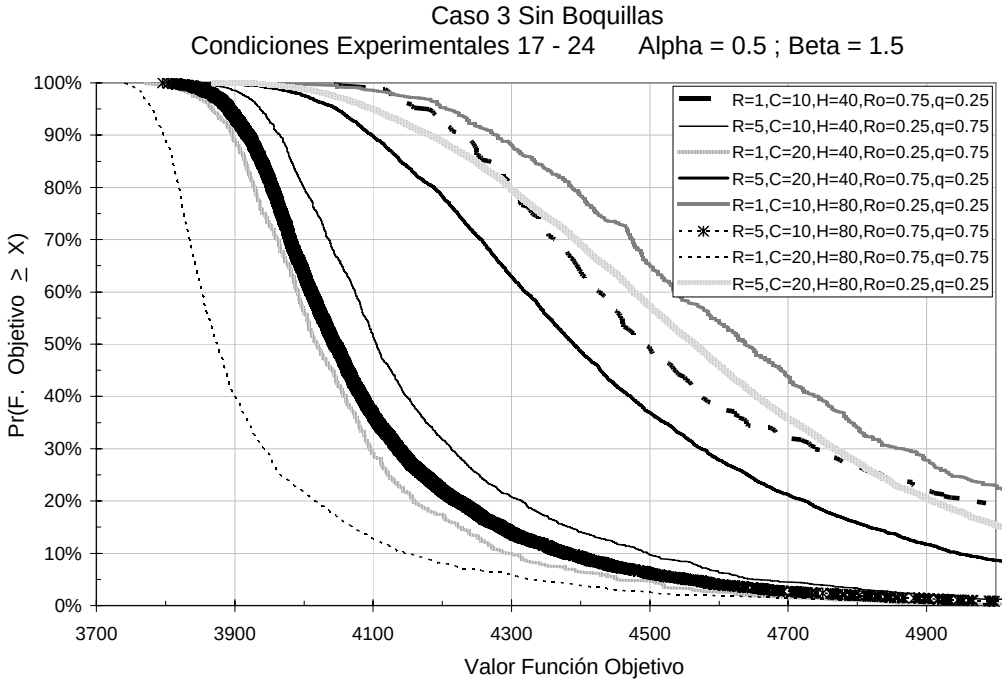


Figura 12 - Experimentos 17 a 24

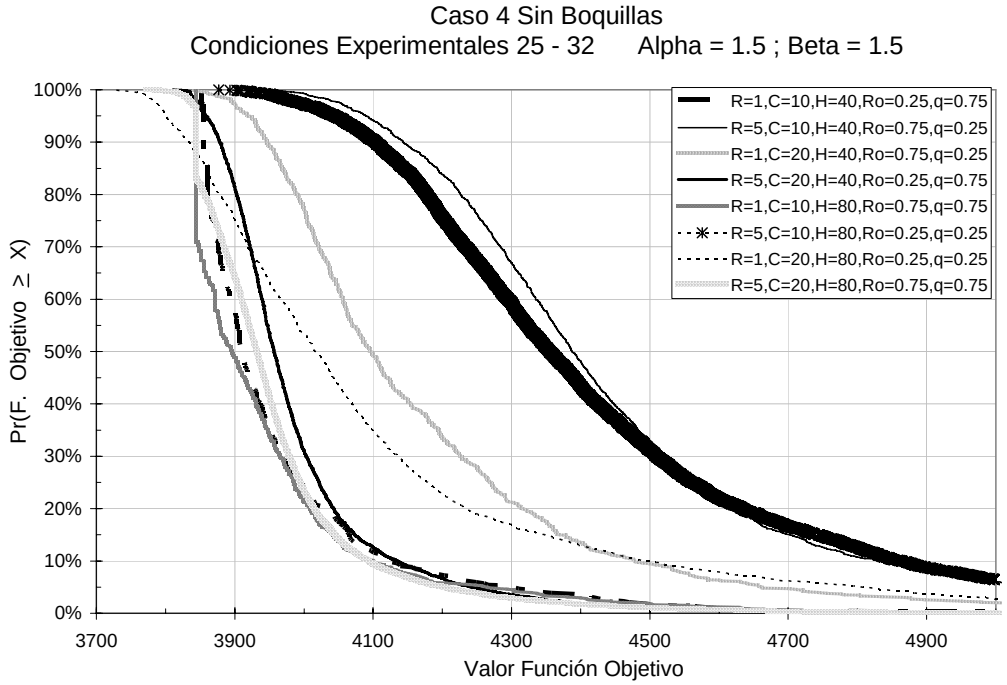


Figura 13 - Experimentos 25 a 32

En la Figura 14 se presentan las mejores 4 condiciones experimentales, del total de 32 realizadas. Si se tuviera que elegir alguna de ellas como la mejor, esta sería la combinación: 1 réplica, 20 colonias, 80 hormigas, un alpha de 0.5, beta de 1.5 y Ro y q de 0.75, correspondiente a la condición experimental 23.

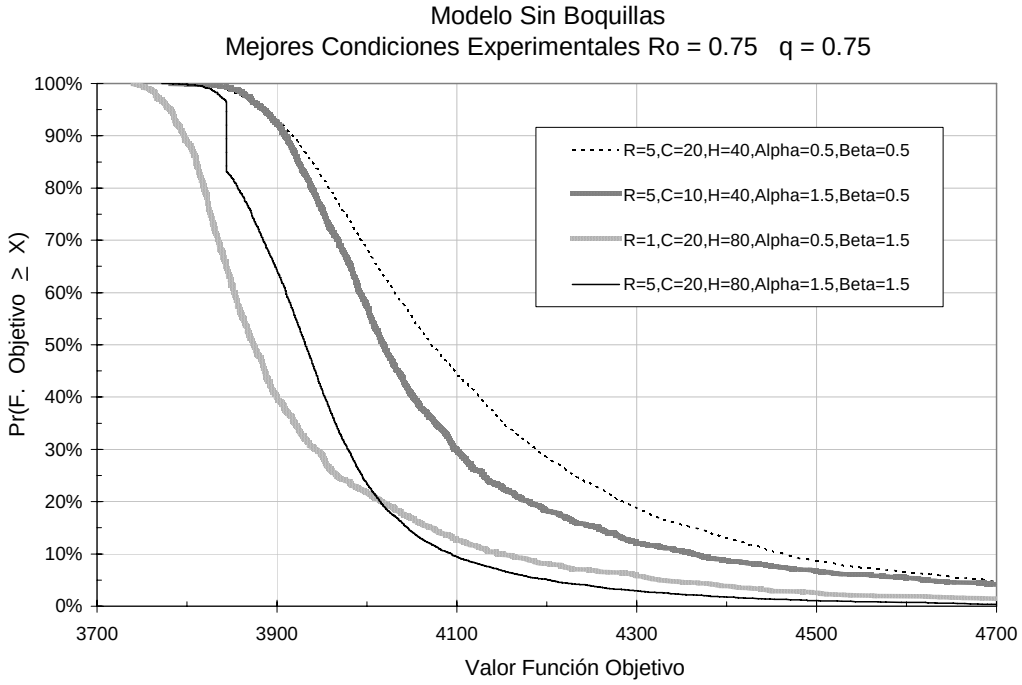


Figura 14 - Mejores Condiciones Sin Boquillas

El resumen de los resultados obtenidos para las 6 variables de respuesta en las 32 condiciones experimentales se presenta en la Tabla 8.

Para el tiempo de ejecución los resultados de regresión obtenidos en Minitab 14 fueron los siguientes.

```
The regression equation is
Tiempo (y) = 7406 - 1485 Replicas - 340 Colonias - 86.0 Hormigas - 1820 Alpha
         - 3680 Ro + 68.7 Replicas*Colonias + 17.5 Replicas*Hormigas
         + 4.75 Colonias*Hormigas + 3527 Alpha*Ro

S = 134.281   R-Sq = 99.7%   R-Sq(adj) = 99.6%

Unusual Observations
Obs  Replicas  Tiempo     Fit  SE Fit  Residual  St Resid
  8      5.00  6831.0  6377.4    75.1     453.6      4.07R
 24      5.00  6140.6  6377.4    75.1    -236.8     -2.13R
```

Para el mejor valor de la función objetivo (percentil 100) los resultados de regresión obtenidos al usar fueron:

```
The regression equation is
100%(y) = 4640 - 6.67 Colonias - 4.74 Hormigas - 207 Alpha - 62.9 Beta - 910 q
       + 5.89 Hormigas*q + 318 Alpha*q

S = 71.5855   R-Sq = 72.2%   R-Sq(adj) = 64.1%

Unusual Observations
Obs  Colonias    100%     Fit  SE Fit  Residual  St Resid
  2      10.0  4342.0  4120.4    35.8     221.6      3.57R
 20      20.0  3842.0  3990.8    35.8    -148.8     -2.40R
```

Para el percentil 90 los resultados de regresión obtenidos son:

```
The regression equation is
90%(y) = 6044 + 90.1 Replicas - 44.4 Colonias - 9.68 Hormigas - 775 Alpha
      - 245 Beta - 798 Ro - 1527 q - 24.2 Replicas*Beta - 78.7 Replicas*q
      + 14.9 Colonias*Alpha + 33.2 Colonias*q + 2.16 Hormigas*Beta
      + 7.17 Hormigas*Ro + 4.45 Hormigas*q + 79.5 Alpha*Beta + 233 Alpha*Ro
      + 404 Alpha*q

S = 47.2094   R-Sq = 98.1%   R-Sq(adj) = 95.9%

Unusual Observations
Obs  Replicas      90%      Fit  SE Fit  Residual  St Resid
  2      5.00  4918.00  4829.44   35.41     88.56      2.84R
 12      5.00  4182.00  4265.06   35.41    -83.06     -2.66R
```

Para el percentil 75 los resultados de regresión obtenidos son:

```
The regression equation is
75%(y) = 6472 + 128 Replicas - 53.2 Colonias - 10.7 Hormigas - 967 Alpha - 271
Beta
      - 982 Ro - 1756 q - 36.2 Replicas*Beta - 107 Replicas*q
      + 19.6 Colonias*Alpha + 37.7 Colonias*q + 2.31 Hormigas*Beta
      + 8.96 Hormigas*Ro + 4.48 Hormigas*q + 102 Alpha*Beta + 283 Alpha*Ro
      + 456 Alpha*q

S = 55.3458   R-Sq = 98.4%   R-Sq(adj) = 96.6%

Unusual Observations
Obs  Replicas      75%      Fit  SE Fit  Residual  St Resid
  2      5.00  5210.00  5115.38   41.51     94.62      2.58R
 12      5.00  4334.00  4435.13   41.51   -101.13     -2.76R
```

Tabla 8 - Resultados para el Caso 1 - Sin Boquillas

StdOrder	RunOrder	CenterPt	Blocks	Replicas	Colonias	Hormigas	Alpha	Beta	Ro	q	Tiempo	100%	90%	75%	50%	Ln (Desv)
1	6	1	1	1	10	40	0.5	0.5	0.75	0.75	24.5	3825	3950	3999	4116	5.782
2	22	1	1	5	10	40	0.5	0.5	0.25	0.25	780.7	4342	4918	5210	5598	6.304
3	24	1	1	1	20	40	0.5	0.5	0.25	0.25	139.7	4119	4302	4431	4746	6.385
4	9	1	1	5	20	40	0.5	0.5	0.75	0.75	1477.9	3798	3918	3977	4071	5.602
5	23	1	1	1	10	80	0.5	0.5	0.25	0.75	48.6	3834	3989	4074	4211	5.811
6	5	1	1	5	10	80	0.5	0.5	0.75	0.25	1586.5	3983	4526	4794	5210	6.428
7	17	1	1	1	20	80	0.5	0.5	0.75	0.25	276.9	3841	4022	4111	4308	6.397
8	19	1	1	5	20	80	0.5	0.5	0.25	0.75	6831	3771	3924	3998	4117	5.729
9	11	1	1	1	10	40	1.5	0.5	0.25	0.25	25.3	3960	4171	4262	4510	6.097
10	10	1	1	5	10	40	1.5	0.5	0.75	0.75	430.2	3782	3909	3953	4019	5.501
11	2	1	1	1	20	40	1.5	0.5	0.75	0.75	89.5	3871	3871	3871	3883	5.168
12	8	1	1	5	20	40	1.5	0.5	0.25	0.25	1588	3890	4182	4334	4553	6.162
13	14	1	1	1	10	80	1.5	0.5	0.75	0.25	79.5	3873	4016	4089	4220	6.222
14	27	1	1	5	10	80	1.5	0.5	0.25	0.75	1616.8	3814	3900	3940	4005	5.495
15	31	1	1	1	20	80	1.5	0.5	0.25	0.75	277.7	3836	3836	3836	3848	5.132
16	28	1	1	5	20	80	1.5	0.5	0.75	0.25	6142.5	3834	4117	4283	4510	6.162
17	29	1	1	1	10	40	0.5	1.5	0.75	0.25	43.1	4043	4222	4329	4493	5.931
18	12	1	1	5	10	40	0.5	1.5	0.25	0.75	425.4	3798	3966	4019	4105	5.477
19	1	1	1	1	20	40	0.5	1.5	0.25	0.75	81.3	3771	3896	3942	4018	5.257
20	15	1	1	5	20	40	0.5	1.5	0.75	0.25	1602.1	3842	4098	4222	4390	5.884
21	20	1	1	1	10	80	0.5	1.5	0.25	0.25	79.8	3883	4278	4424	4639	6.057
22	16	1	1	5	10	80	0.5	1.5	0.75	0.75	1574	3796	3919	3968	4045	5.400
23	30	1	1	1	20	80	0.5	1.5	0.75	0.75	276.3	3740	3797	3826	3874	5.285
24	18	1	1	5	20	80	0.5	1.5	0.25	0.25	6140.6	3870	4183	4343	4565	5.964
25	13	1	1	1	10	40	1.5	1.5	0.25	0.75	27.1	3852	3855	3868	3910	5.125
26	4	1	1	5	10	40	1.5	1.5	0.75	0.25	432.2	3897	4143	4257	4387	5.705
27	3	1	1	1	20	40	1.5	1.5	0.75	0.25	82.2	3841	3946	4005	4097	5.561
28	7	1	1	5	20	40	1.5	1.5	0.25	0.75	1605.1	3811	3878	3913	3957	4.930
29	21	1	1	1	10	80	1.5	1.5	0.75	0.75	80.9	3844	3844	3844	3896	5.061
30	26	1	1	5	10	80	1.5	1.5	0.25	0.25	1603.7	3877	4104	4204	4355	5.782
31	32	1	1	1	20	80	1.5	1.5	0.25	0.25	281.4	3729	3834	3901	4017	5.752
32	25	1	1	5	20	80	1.5	1.5	0.75	0.75	6130.4	3772	3844	3872	3932	4.919
									Mínimo		24.5	3729	3797	3826	3848	4.919
									Máximo		6831	4342	4918	5210	5598	6.428
									Average		1308.778	3866.8	4042.4	4128.1	4268.9	5.702
									Desviación		2019.551	119.41	232.96	298.62	393.75	0.447

Para el percentil 50 los resultados de regresión son.

```
The regression equation is
50%(y) = 7442 + 157 Replicas - 71.6 Colonias - 14.3 Hormigas - 1208 Alpha
      - 581 Beta - 1350 Ro - 2146 q - 45.3 Replicas*Beta - 128 Replicas*q
      + 22.8 Colonias*Alpha + 10.9 Colonias*Beta + 40.5 Colonias*q
      + 3.23 Hormigas*Beta + 13.6 Hormigas*Ro + 4.99 Hormigas*q + 156
Alpha*Beta
      + 320 Alpha*Ro + 548 Alpha*q

S = 54.9777   R-Sq = 99.2%   R-Sq(adj) = 98.1%

Unusual Observations
Obs  Replicas      50%      Fit  SE Fit  Residual  St Resid
 12      5.00  4553.00  4660.84   42.36   -107.84     -3.08R
```

Para el logaritmo de la desviación de los datos, los resultados de regresión son los siguientes.

```
The regression equation is
Ln (Desv)(y) = 7.03 - 0.0354 Replicas - 0.231 Alpha - 0.393 Beta - 0.398 Ro
            - 0.404 q + 0.00143 Replicas*Colonias + 0.0205 Replicas*Alpha
            - 0.0326 Colonias*q + 0.00570 Hormigas*Ro - 0.00368 Hormigas*q
            - 0.276 Alpha*q

S = 0.0464715   R-Sq = 99.3%   R-Sq(adj) = 98.9%

Unusual Observations
Obs  Replicas  Ln (Desv)      Fit   SE Fit  Residual  St Resid
  8      5.00    5.72900  5.63295  0.02965   0.09605      2.68R
```

El propósito de la estimación de estas ecuaciones es el identificar mejores condiciones experimentales a las 32 que fueron previamente corridas. Se han escogido valores límites para los parámetros del modelo, lo cual no garantiza que la mejor condición se haya corrido, puede estar dentro del espacio definido por los niveles bajos y altos de cada parámetro, y aún cuando esta se encontrará en dicha frontera, es necesario recordar que no se han considerado 96 condiciones (el total de combinaciones es 128).

Utilizando el *"Solver"* de Excel, y tomando como referencia las ecuaciones que se han encontrado para cada variable de respuesta, se ha tratado de identificar los niveles para cada parámetro que optimizan cada una de las variables, sujeto a una restricción de tiempo disponible. Tomando como restricción el tiempo de ejecución, el cual se encuentra expresado en términos de una ecuación de regresión previamente señalada, se calculan los valores de

los parámetros (dentro del espacio de búsqueda en el cual se ha definido con los niveles bajo y alto) que minimizan separadamente cada una de las 5 restantes variables.

Tabla 9 - Caso 1 - Búsqueda de mejores condiciones experimentales

Tiempo Máximo	40	80	160	320	640	1280	2560	5120	6300	Mejor
Percentila 100										
Función Objetivo	3722.6	3722.3	3721.3	3719.7	3719.7	3719.7	3719.7	3719.7	3719.7	3719.7
Tiempo Corrida	29.125	55.625	135.125	267.625	267.625	268	268	268	268	268
Replicas	1	1	1	1	1	1	1	1	1	1
Colonias	20	20	20	20	20	20	20	20	20	20
Hormigas	71	72	75	80	80	80	80	80	80	80
Alpha	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Beta	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Ro	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
q	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
Percentila 90										
Función Objetivo	3764.0	3764.0	3694.7	3694.7	3678.1	3672.9	3667.6	3667.6	3667.6	3667.6
Tiempo Corrida	15.625	15.625	16.625	16.625	385.625	975	1564	1564	1564	1564
Replicas	5	5	2	2	3	4	5	5	5	5
Colonias	12	12	20	20	20	20	20	20	20	20
Hormigas	40	40	45	45	40	40	40	40	40	40
Alpha	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Beta	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Ro	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
q	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
Percentila 75										
Función Objetivo	3694.6	3694.6	3694.6	3694.6	3673.8	3667.2	3660.7	3660.7	3660.7	3660.7
Tiempo Corrida	16.625	16.625	16.625	16.625	385.625	975	1564	1564	1564	1564
Replicas	2	2	2	2	3	4	5	5	5	5
Colonias	20	20	20	20	20	20	20	20	20	20
Hormigas	45	45	45	45	40	40	40	40	40	40
Alpha	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Beta	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Ro	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
q	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
Percentila 50										
Función Objetivo	3705.5	3705.5	3705.5	3703.1	3676.2	3669.2	3662.3	3662.3	3662.3	3662.3
Tiempo Corrida	16.625	16.625	16.625	273.425	385.625	975	1564	1564	1564	1564
Replicas	2	2	2	3	3	4	5	5	5	5
Colonias	20	20	20	18	20	20	20	20	20	20
Hormigas	45	45	45	40	40	40	40	40	40	40
Alpha	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Beta	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Ro	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
q	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
Logaritmo de la Desviación										
Función Objetivo	4.778	4.778	4.778	4.778	4.778	4.778	4.778	4.778	4.778	4.778
Tiempo Corrida	32.875	32.875	32.875	32.875	32.875	33	33	33	33	33
Replicas	1	1	1	1	1	1	1	1	1	1
Colonias	20	20	20	20	20	20	20	20	20	20
Hormigas	40	40	40	40	40	40	40	40	40	40
Alpha	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Beta	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Ro	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
q	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75

La Tabla 9, presenta las combinaciones resultantes al minimizar cada una de las variables de respuesta. La primera columna se asocia a la combinación de parámetros requerida para minimizar cada una de las 5 variables de respuesta (Mejor valor, Percentila 90, Percentila 75,

Percentila 50 y el logaritmo de la desviación), sujeto a que el tiempo de ejecución no sea mayor a 40 segundos. Se presentan para cada caso los parámetros, el valor mínimo de la función y el tiempo estimado de ejecución de dicha condición. Se han seleccionado tres condiciones experimentales.

La justificación para la utilización del rango de 40 a 6300 segundos, radica en los tiempos mínimo y máximo que fueron encontrados en las 32 condiciones previamente corridas. La última columna se asocia a los parámetros que minimizan cada función sin tener en cuenta restricción alguna de tiempo de ejecución.

Una vez analizados los resultados, se encuentra que el proceso de optimización ha definido tres condiciones experimentales que pueden proporcionar mejores resultados para las variables de respuesta, y estas son:

- 1 réplica, 20 colonias, 80 hormigas, alpha se fija en 0.5, beta en 1.5, mientras que Ro y q en 0.75. Esta condición ya fue corrida y corresponde al experimento 23.
- 5 réplicas, 20 colonias, 40 hormigas, alpha se fija en 0.5, beta en 1.5, mientras que Ro y q en 0.75.
- 1 réplicas, 20 colonias, 40 hormigas, alpha y beta se fijan en 1.5, mientras que Ro y q en 0.75.

De las dos condiciones que no se han corrido, es importante mencionar que tienen 5 parámetros en común. Los dos parámetros que cambian son el número de réplicas y el parámetro de alpha. Tomando como referencia estos resultados, se corrieron estas dos condiciones experimentales, al igual que dos condiciones adicionales que nacen de la combinación de los dos parámetros que varían (tampoco estaban dentro de las iniciales 32 condiciones).

En la Figura 15 se hace una comparación entre el comportamiento de estas 4 nuevas condiciones, y la mejor de las 32 condiciones inicialmente encontradas.

Las nuevas condiciones tienen buenos comportamientos, no obstante ninguna de ellas parece ser mejor que la condición 23, anteriormente mencionada.

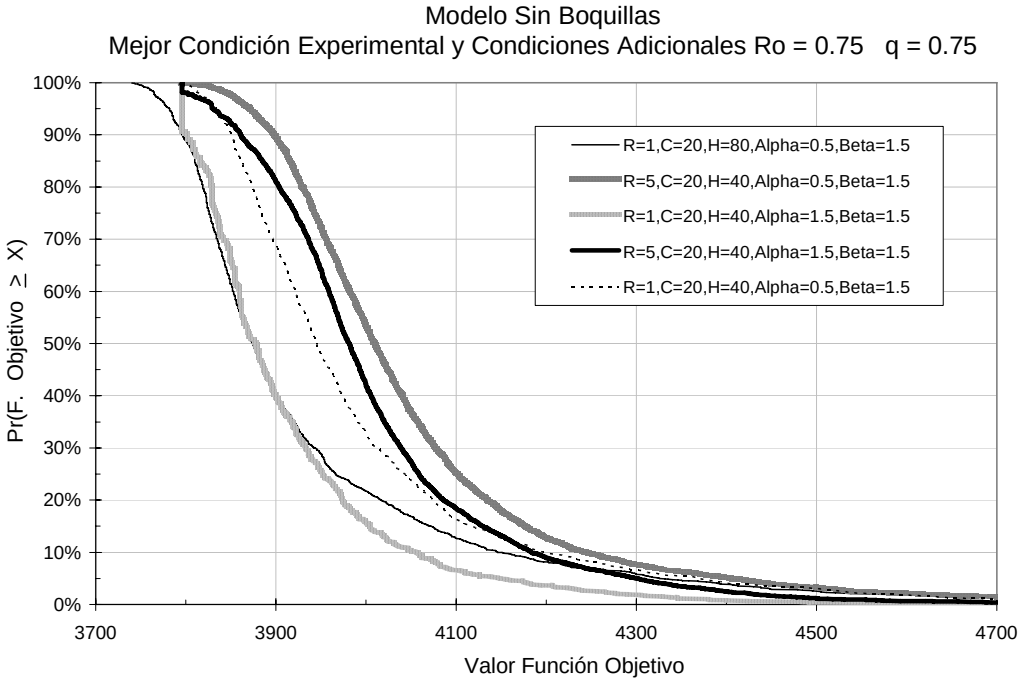


Figura 15 - Condiciones Adicionales

Dadas las anteriores consideraciones, la mejor condición experimental encontrada para el heurístico que no tiene en cuenta la asignación de boquillas es la siguiente:

- 1 Réplica, 20 Colonias, 80 Hormigas, Alpha igual a 0.5, Beta de 1.5 y los parámetros Ro y q en 0.75.

4.1.5 Resultados Caso 2 – Con Boquillas

Se continúa la metodología gráfica para presentar las funciones de distribución de probabilidad para las 32 condiciones experimentales que fueron desarrolladas para el Caso 2.

Los resultados obtenidos para las 6 variables de respuesta para las 32 condiciones experimentales se presentan en la Tabla 10.

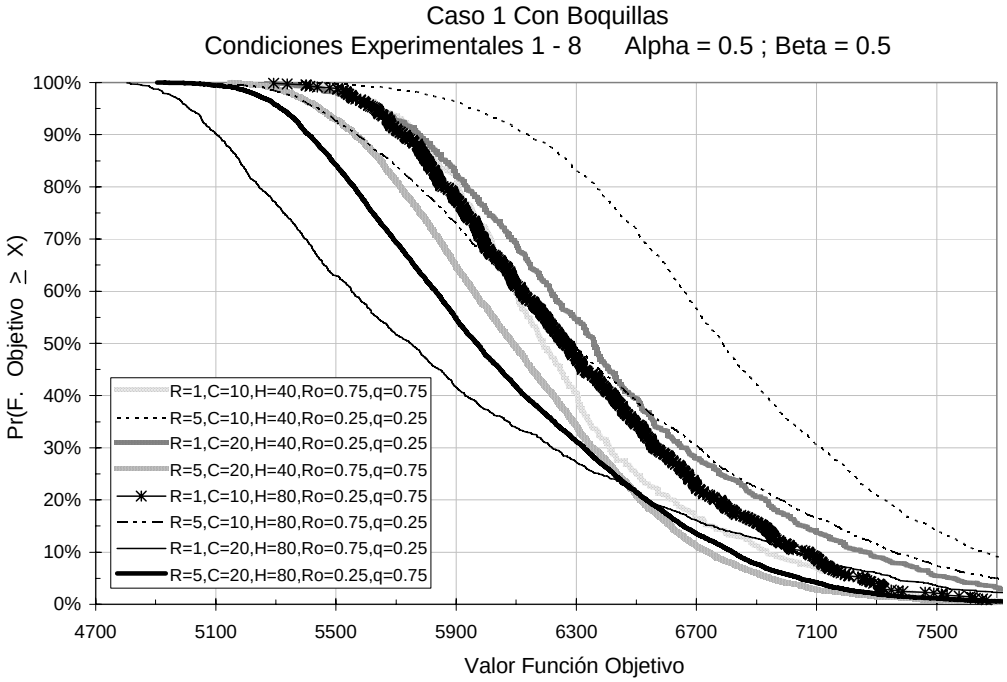


Figura 16 - Experimentos 1 a 8

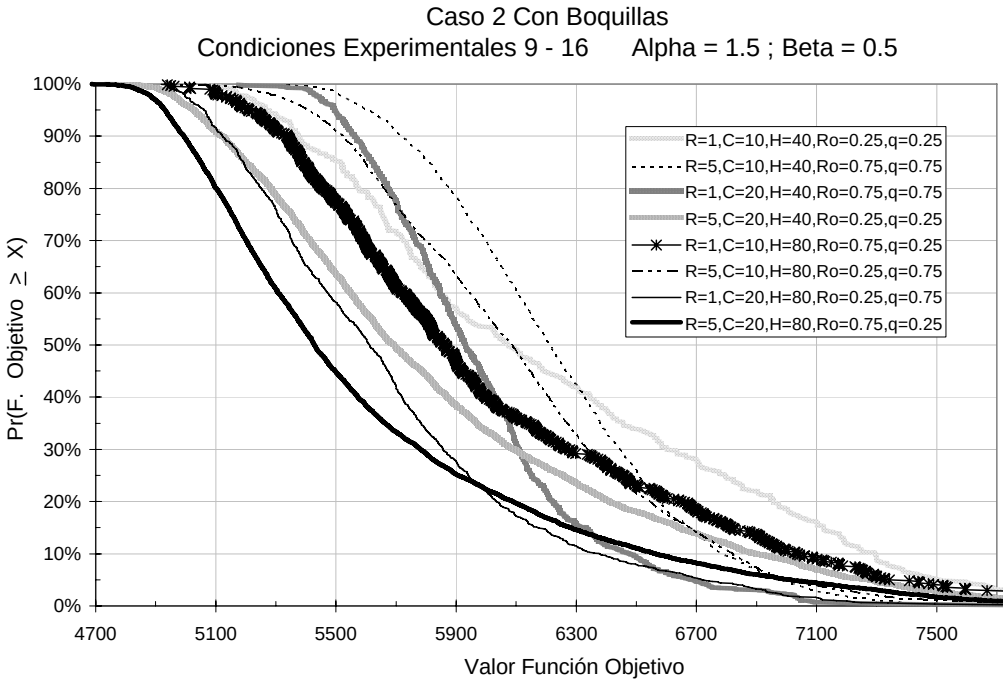


Figura 17 –Experimentos 9 a 16

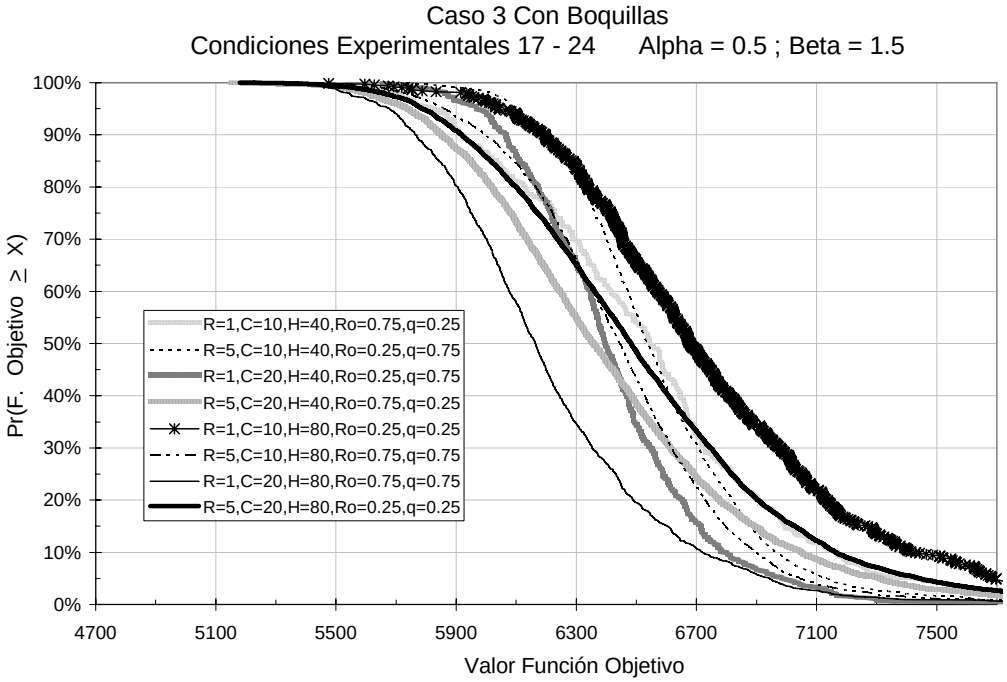


Figura 18 - Experimentos 17 a 24

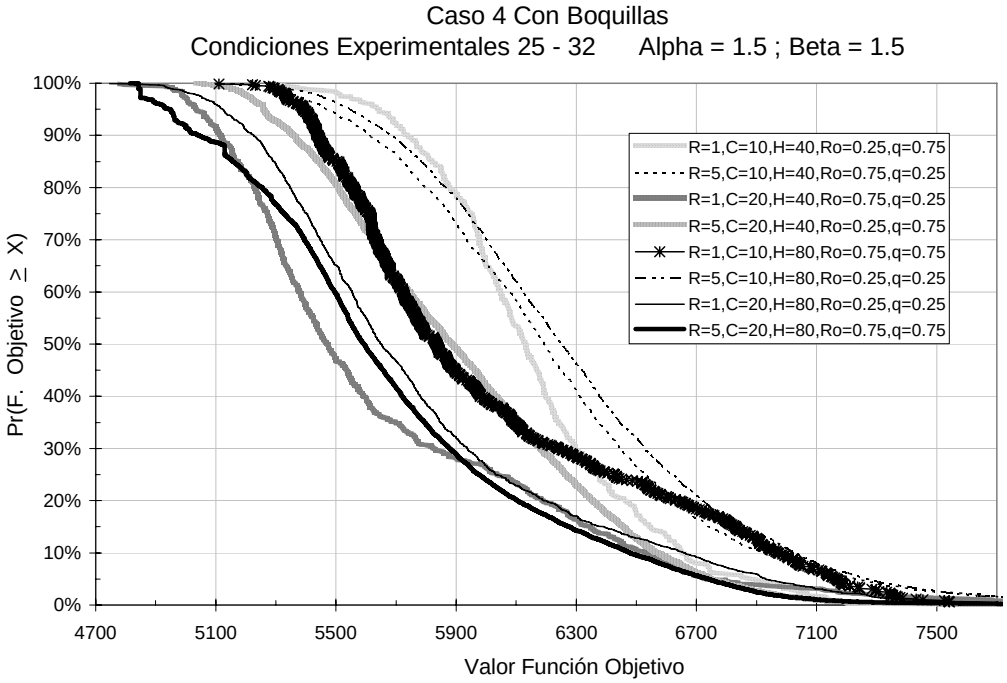
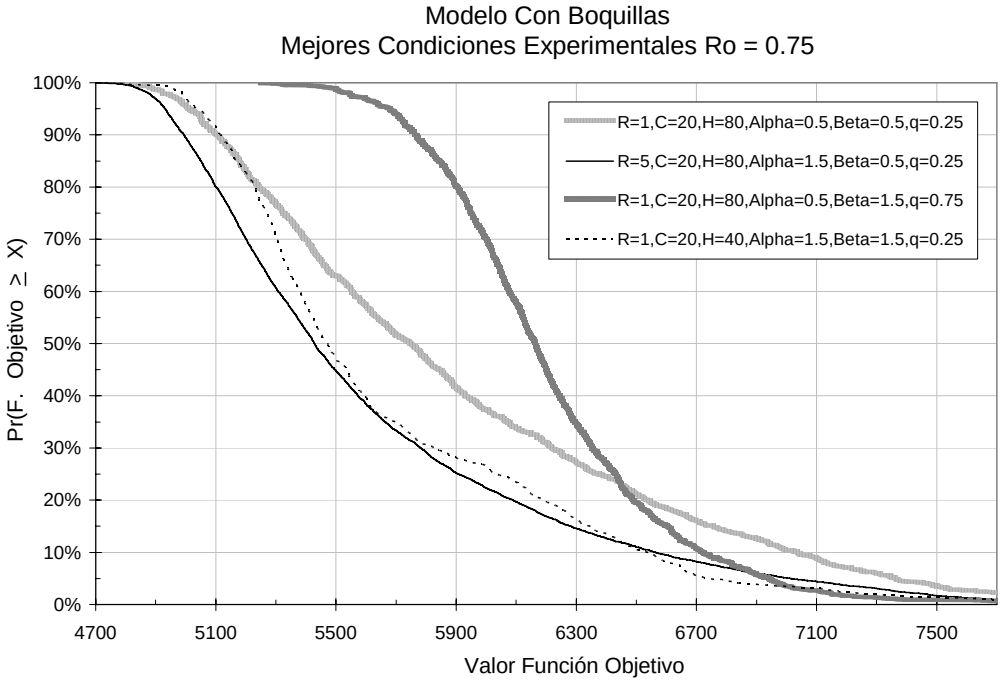


Figura 19 - Experimentos 25 a 32

En la Figura 20 se comparan las mejores 4 condiciones experimentales para este Caso 2.



Gráficamente se define como mejor condición experimental la 16, donde se combinan los siguientes parámetros: 5 réplicas, 20 colonias, 80 hormigas, alpha de 1.5, beta de 0.5, Ro de 0.75 y un q de 0.25

Para la estimación de ecuaciones de regresión que explican el comportamiento de cada variable de respuesta, se utilizó igualmente el procedimiento de *"Stepwise"* y se continúo con regresiones lineales utilizando Minitab 14. Los resultados se presentan a continuación.

Para el tiempo de ejecución los resultados de regresión obtenidos son.

```
The regression equation is
Tiempo = 5520 - 1284 Replicas - 272 Colonias - 63.7 Hormigas - 1241 Alpha
         - 2733 Ro + 76.4 Replicas*Colonias + 13.6 Replicas*Hormigas
         + 3.51 Colonias*Hormigas + 2573 Alpha*Ro

S = 247.463   R-Sq = 99.0%   R-Sq(adj) = 98.5%

Unusual Observations
Obs  Replicas  Tiempo     Fit  SE Fit  Residual  St Resid
  4      5.00  1639.9  2573.8   138.3    -933.9     -4.55R
```

Tabla 10 - Resultados para el Caso 2 - Con Boquillas

StdOrder	RunOrder	CenterPt	Blocks	Replicas	Colonias	Hormigas	Alpha	Beta	Ro	q	Tiempo	100%	90%	75%	50%	Ln (Desv)
1	6	1	1	1	10	40	0.5	0.5	0.75	0.75	38.8	5386	5778	5960	6190	6.140
2	22	1	1	5	10	40	0.5	0.5	0.25	0.25	769.9	5346	6139	6439	6789	6.366
3	24	1	1	1	20	40	0.5	0.5	0.25	0.25	134.4	5268	5776	6010	6360	6.381
4	9	1	1	5	20	40	0.5	0.5	0.75	0.75	1639.9	5145	5562	5780	6088	6.228
5	23	1	1	1	10	80	0.5	0.5	0.25	0.75	54.4	5292	5729	5931	6262	6.228
6	5	1	1	5	10	80	0.5	0.5	0.75	0.25	1915	4941	5570	5866	6284	6.546
7	17	1	1	1	20	80	0.5	0.5	0.75	0.25	289.9	4804	5105	5325	5570	6.619
8	19	1	1	5	20	80	0.5	0.5	0.25	0.75	6343.3	4905	5410	5623	5970	6.306
9	11	1	1	1	10	40	1.5	0.5	0.25	0.25	40	5078	5374	5654	6075	6.613
10	10	1	1	5	10	40	1.5	0.5	0.75	0.75	770.5	5183	5719	5937	6213	6.039
11	2	1	1	1	20	40	1.5	0.5	0.75	0.75	135.5	5176	5561	5717	5932	5.918
12	8	1	1	5	20	40	1.5	0.5	0.25	0.25	2966.8	4737	5123	5352	5694	6.543
13	14	1	1	1	10	80	1.5	0.5	0.75	0.25	91.2	4939	5338	5537	5861	6.561
14	27	1	1	5	10	80	1.5	0.5	0.25	0.75	1613.5	4926	5524	5724	6088	6.238
15	31	1	1	1	20	80	1.5	0.5	0.25	0.75	294.2	4919	5123	5306	5612	6.233
16	28	1	1	5	20	80	1.5	0.5	0.75	0.25	6161.6	4686	4994	5153	5428	6.483
17	29	1	1	1	10	40	0.5	1.5	0.75	0.25	40.6	5510	5920	6225	6548	6.175
18	12	1	1	5	10	40	0.5	1.5	0.25	0.75	771.4	5674	6193	6362	6539	5.801
19	1	1	1	1	20	40	0.5	1.5	0.25	0.75	137	5654	6066	6217	6397	5.742
20	15	1	1	5	20	40	0.5	1.5	0.75	0.25	2972	5151	5852	6085	6357	6.187
21	20	1	1	1	10	80	0.5	1.5	0.25	0.25	91.2	5476	6196	6414	6678	6.224
22	16	1	1	5	10	80	0.5	1.5	0.75	0.75	1633.3	5186	5996	6220	6445	5.923
23	30	1	1	1	20	80	0.5	1.5	0.75	0.75	308.2857	5248	5764	5950	6164	5.981
24	18	1	1	5	20	80	0.5	1.5	0.25	0.25	6319.8	5179	5918	6171	6483	6.230
25	13	1	1	1	10	40	1.5	1.5	0.25	0.75	40.7	5314	5738	5954	6131	5.918
26	4	1	1	5	10	40	1.5	1.5	0.75	0.25	775.2	5118	5612	5878	6193	6.291
27	3	1	1	1	20	40	1.5	1.5	0.75	0.25	139.6	4780	5116	5271	5470	6.389
28	7	1	1	5	20	40	1.5	1.5	0.25	0.75	2967.9	5034	5361	5565	5885	6.158
29	21	1	1	1	10	80	1.5	1.5	0.75	0.75	90.3	5111	5441	5612	5835	6.373
30	26	1	1	5	10	80	1.5	1.5	0.25	0.25	1623.7	5095	5686	5939	6249	6.283
31	32	1	1	1	20	80	1.5	1.5	0.25	0.25	296.3	4746	5227	5402	5651	6.331
32	25	1	1	5	20	80	1.5	1.5	0.75	0.75	6189	4814	5044	5328	5593	6.285
									Mínimo		38.8	4686	4994	5153	5428	5.742
									Máximo		6343.3	5674	6196	6439	6789	6.619
									Average		1489.225	5119.4	5592.3	5809.6	6094.8	6.242
									Desviación		2040.152	260.13	349.25	359.44	359.95	0.227

Para el mejor valor de la función objetivo (percentil 100) los resultados son.

```
The regression equation is
100%(y) = 5963 - 33.0 Colonias - 4.75 Hormigas - 424 Alpha + 350 Beta - 528 Ro
       + 441 q - 50.6 Replicas*q + 24.4 Colonias*q + 2.87 Hormigas*Alpha
       - 6.52 Hormigas*q - 203 Alpha*Beta + 345 Alpha*Ro

S = 57.8286   R-Sq = 97.0%   R-Sq(adj) = 95.1%

Unusual Observations
Obs  Colonias    100%     Fit  SE Fit  Residual  St Resid
  3      20.0  5268.0  5148.4    35.7     119.6      2.63R
```

Para el percentil 90 los resultados de regresión son.

```
The regression equation is
90%(y) = 6247 + 60.0 Replicas - 483 Alpha - 3.53 Replicas*Colonias
      - 0.358 Colonias*Hormigas + 6.60 Hormigas*Beta - 10.4 Hormigas*Ro
      - 211 Alpha*Beta + 356 Alpha*Ro + 157 Alpha*q

S = 113.323   R-Sq = 92.5%   R-Sq(adj) = 89.5%

Unusual Observations
Obs  Replicas     90%     Fit  SE Fit  Residual  St Resid
 25      1.00  5738.0  5532.0    68.7     206.0      2.29R
```

Para el percentil 75 los resultados de regresión son.

```
The regression equation is
75%(Y) = 6346 + 114 Replicas - 335 Alpha - 4.11 Replicas*Colonias - 75.1
Replicas*q
      - 0.293 Colonias*Hormigas - 12.7 Colonias*Alpha + 18.6 Colonias*q
      + 7.02 Hormigas*Beta - 8.52 Hormigas*Ro - 5.63 Hormigas*q - 226
Alpha*Beta
      + 252 Alpha*Ro + 346 Alpha*q

S = 87.2686   R-Sq = 96.6%   R-Sq(adj) = 94.1%
```

Para el percentil 50 los resultados de regresión son.

```
The regression equation is
50%(y) = 6879 + 24.4 Replicas - 885 q - 0.491 Colonias*Hormigas
      - 23.0 Colonias*Alpha + 34.7 Colonias*q + 6.35 Hormigas*Beta
      - 5.84 Hormigas*Ro - 254 Alpha*Beta + 322 Alpha*q

S = 108.388   R-Sq = 93.6%   R-Sq(adj) = 90.9%

Unusual Observations
Obs  Replicas     50%     Fit  SE Fit  Residual  St Resid
 12      5.00  5694.0  5868.5    66.7    -174.5     -2.04R
```

Para el logaritmo de la desviación de los datos, los resultados son.

```
The regression equation is
Ln (Desv)(y) = 6.97 - 0.0594 Replicas - 0.0129 Colonias - 0.400 Beta - 1.11 q
            + 0.00491 Replicas*Colonias - 0.000637 Replicas*Hormigas
            + 0.0506 Replicas*q + 0.00418 Hormigas*Ro + 0.00621 Hormigas*q
            + 0.203 Alpha*Beta - 0.188 Alpha*Ro
```

```
S = 0.0686831   R-Sq = 94.1%   R-Sq(adj) = 90.9%

Unusual Observations
Obs  Replicas  Ln (Desv)     Fit  SE Fit  Residual  St Resid
 29      1.00     6.3730  6.2487  0.0445    0.1243      2.37R
```

Tabla 11 - Caso 2 - Búsqueda de mejores condiciones experimentales

Tiempo Máximo	**40**	**80**	**160**	**320**	**640**	**1280**	**2560**	**5120**	**6300**	**Mejor**
					Percentila 100					
Función Objetivo	4773.8	4769.7	4761.4	4744.8	4738.0	4735.5	4722.8	4697.5	4684.9	4684.9
Tiempo Corrida	40	80	160	320	640	915	2247	4911	6243	6243
Replicas	1	1	1	1	1	1	2	4	5	5
Colonias	20	20	20	20	20	20	20	20	20	20
Hormigas	64	66	70	78	80	80	80	80	80	80
Alpha	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Beta	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Ro	0.26	0.26	0.26	0.26	0.51	0.75	0.75	0.75	0.75	0.75
q	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
					Percentila 90					
Función Objetivo	5131.5	5118.4	5092.3	5040.2	4953.1	4880.2	4869.6	4848.4	4837.8	4837.8
Tiempo Corrida	40	80	160	320	640	915	2247	4911	6243	6243
Replicas	1	1	1	1	1	1	2	4	5	5
Colonias	20	20	20	20	20	20	20	20	20	20
Hormigas	64	66	70	78	80	80	80	80	80	80
Alpha	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Beta	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Ro	0.26	0.26	0.26	0.26	0.51	0.75	0.75	0.75	0.75	0.75
q	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
					Percentila 75					
Función Objetivo	5245.1	5233.2	5209.4	5162.0	5074.7	5000.5	5000.5	5000.5	5000.5	5000.5
Tiempo Corrida	40	80	160	320	640	915	915	915	915	915
Replicas	1	1	1	1	1	1	1	1	1	1
Colonias	20	20	20	20	20	20	20	20	20	20
Hormigas	64	66	70	78	80	80	80	80	80	80
Alpha	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Beta	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Ro	0.26	0.26	0.26	0.26	0.51	0.75	0.75	0.75	0.75	0.75
q	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
					Percentila 50					
Función Objetivo	5542.7	5534.1	5516.4	5479.8	5328.1	5213.9	5213.9	5213.9	5213.9	5213.9
Tiempo Corrida	40	80	160	320	640	915	915	915	915	915
Replicas	1	1	1	1	1	1	1	1	1	1
Colonias	20	20	20	20	20	20	20	20	20	20
Hormigas	41	42	44	51	80	80	80	80	80	80
Alpha	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Beta	1.5	1.5	1.5	1.5	0.5	0.5	0.5	0.5	0.5	0.5
Ro	0.67	0.69	0.72	0.74	0.51	0.75	0.75	0.75	0.75	0.75
q	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
					Logaritmo de la Desviación					
Función Objetivo	5.693	5.691	5.688	5.688	5.688	5.688	5.688	5.688	5.688	5.688
Tiempo Corrida	40	80	145.875	145.875	145.875	146	146	146	146	146
Replicas	1	1	1	1	1	1	1	1	1	1
Colonias	20	20	20	20	20	20	20	20	20	20
Hormigas	40	40	40	40	40	40	40	40	40	40
Alpha	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Beta	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
Ro	0.32	0.30	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
q	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75

Siguiendo el procedimiento realizado para el Caso1 y utilizando el Solver de Excel, en la Tabla 11 se aprecian la combinación de parámetros que minimiza cada variable de respuesta, sujeto a una restricción de tiempo. Se obtuvieron las tres siguientes condiciones:

- 5 réplicas, 20 colonias, 80 hormigas, alpha se fija en 1.5, beta en 0.5, Ro en 0.75 y q en 0.25. Esta condición ya fue corrida y corresponde al experimento 16.
- 1 réplica, 20 colonias, 40 hormigas, alpha se fija en 0.5, beta en 1.5, Ro en 0.25 y q en 0.75. Corresponde a la condición 19.
- 1 réplica, 20 colonias, 80 hormigas, alpha se fija en 1.5, beta en 0.5, Ro en 0.75 y q en 0.75.

La única condición que fue necesaria correr es la tercera. La Figura 20 ayuda ver gráficamente que la mejor condición experimental encontrada inicialmente (26), tiene un comportamiento muy similar a la mejor condición encontrada después del proceso de optimización.

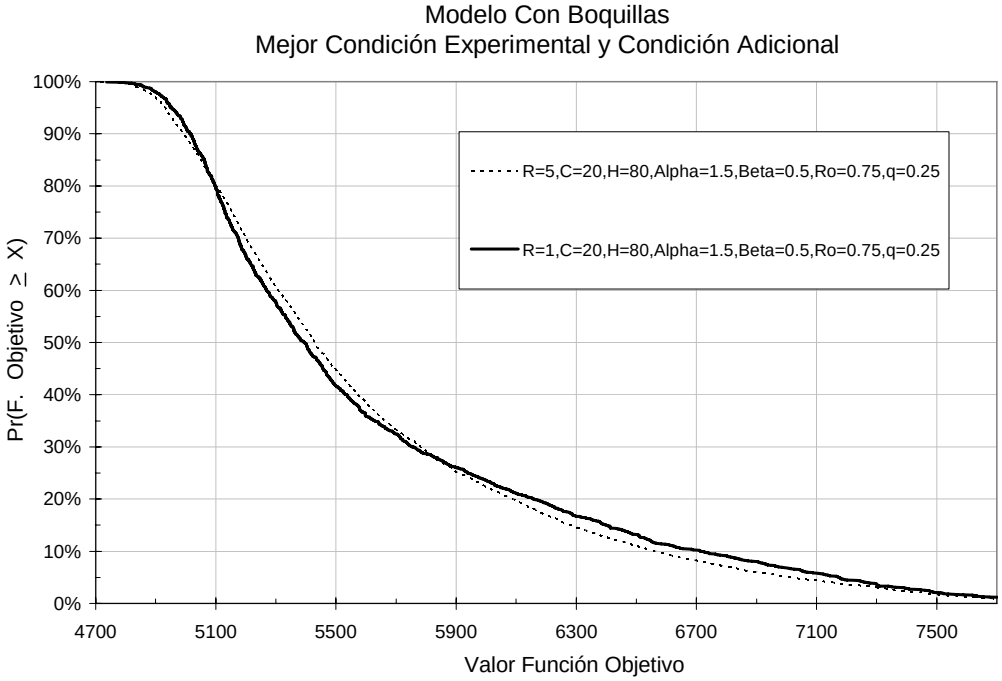


Figura 20 - Condiciones Adicionales

Considerando que estas dos condiciones tienen 6 parámetros en el mismo nivel, donde el único diferente se asocia al número de réplicas, se decide utilizar la condición propuesta por la optimización, dado que el número de réplicas es menor y por tanto también lo es el tiempo de ejecución.

En resumen, la mejor condición experimental encontrada para el heurístico que tiene en cuenta la asignación de boquillas es la siguiente:

- 1 Réplica, 20 Colonias, 80 Hormigas, Alpha igual a 1.5, Beta de 0.5, Ro de 0.75 y q de 0.25.

4.2 Implementación del Heurístico en Tarjetas de la "Fábrica Modelo"

Partiendo de información proporcionada por los encargados de la "Fábrica Modelo" de los tres tipos de tarjetas que se producen en la actualidad y de los parámetros que han sido identificados como los más adecuados a utilizar para el Caso 1 – Sin boquillas, se presenta a continuación las configuraciones propuestas para cada uno de los productos. El Caso 2 – Con boquillas, no es considerado, dado que no se proporcionó los datos asociados a correspondencias entre tipos de productos y tipos de boquillas.

Se considero un requerimiento de producción mínimo de 160 tarjetas.

Por confidencialidad, la primera de las tarjetas ha sido denominada como Dis, y los datos suministrados por la "Fábrica Modelo" se presentan en la Tabla 12. Las mejores soluciones para cada una de las colonias generadas se presentan en la Tabla 13, en tanto que la Tabla 14 presenta la mejor configuración encontrada de tal forma que se minimice el tiempo de recogido. La segunda tarjeta es llamada Char y los datos que la caracteriza, las mejores soluciones encontradas y la mejor configuración encontrada se presentan en las Tablas 15, 16 y 17 respectivamente. La tercera tarjeta ha sido llamada Dri, y la información y resultados relevantes se presentan en las Tablas 18, 19 y 20.

Tabla 12 - Tarjeta tipo Dis.

Producto Dis

Componente	b_i	w_i	v_i	*Componente*	b_i	w_i	v_i
1	6	2	960	17	6	1	5000
2	6	3	960	18	54	1	5000
3	24	1	960	19	54	1	5000
4	12	2	960	20	6	1	5000
5	30	1	960	21	6	1	5000
6	6	1	960	22	6	3	960
7	12	1	960	23	6	1	4000
8	6	1	960	24	6	1	4000
9	6	1	960	25	6	1	4000
10	6	1	960	26	6	1	4000
11	6	1	960	27	6	1	4000
12	6	1	960	28	24	1	4000
13	6	1	960	29	18	1	3000
14	6	1	960	30	6	1	4000
15	6	1	350	31	6	1	4000
16	18	1	5000	32	6	1	3000

Tabla 13 – Mejores soluciones tarjeta tipo Dis.

Réplica	*Colonia*	*Hormiga*	*F. Objetivo*
1	1	47	3592
1	2	6	3650
1	3	63	3670
1	4	66	3641
1	5	66	3622
1	6	32	3548
1	7	5	3580
1	8	49	3539
1	**9**	**10**	**3514**
1	10	28	3552
1	11	5	3562
1	12	18	3570
1	13	46	3534
1	14	79	3562
1	15	17	3552
1	16	67	3520
1	17	18	3516
1	18	41	3542
1	19	41	3546
1	20	15	3518

Tabla 14 - Configuración propuesta tarjeta tipo Dis.

Réplica	*Colonia*	*Hormiga*	*Componente*	*Posición*	*Unidades*	*Distancia*
1	9	10	18	1	27	1
1	9	10	19	2	27	2
1	9	10	29	3	18	3
1	9	10	5	4	6	4
1	9	10	5	5	6	5
1	9	10	5	6	6	6
1	9	10	3	7	6	7
1	9	10	3	8	6	8
1	9	10	7	9	6	9
1	9	10	10	10	6	10
1	9	10	6	11	6	11
1	9	10	30	12	6	12
1	9	10	13	13	6	13
1	9	10	11	14	6	14
1	9	10	12	15	6	15
1	9	10	15	16	2	16
1	9	10	20	17	6	17
1	9	10	26	18	6	18
1	9	10	25	19	6	19
1	9	10	21	20	6	20
1	9	10	4	21	6	22
1	9	10	23	22	6	23
1	9	10	31	23	6	24
1	9	10	24	24	6	25
1	9	10	18	38	27	1
1	9	10	19	39	27	2
1	9	10	5	40	6	3
1	9	10	5	41	6	4
1	9	10	3	42	6	5
1	9	10	3	43	6	6
1	9	10	28	44	24	7
1	9	10	16	45	18	8
1	9	10	27	46	6	9
1	9	10	7	47	6	10
1	9	10	9	48	6	11
1	9	10	8	49	6	12
1	9	10	32	50	6	13
1	9	10	14	51	6	14
1	9	10	4	52	6	16
1	9	10	15	53	2	17
1	9	10	1	54	6	19
1	9	10	2	55	6	22
1	9	10	15	56	2	23
1	9	10	22	57	6	26
1	9	10	17	58	6	27

Tabla 15 - Tarjeta tipo Cha.

Producto Cha

Componente	b_i	w_i	v_i	*Componente*	b_i	w_i	v_i
1	12	2	960	20	6	1	5000
2	12	1	960	21	6	1	5000
3	6	1	960	22	6	1	5000
4	6	1	960	23	36	1	5000
5	6	1	960	24	6	1	5000
6	6	1	960	25	12	1	5000
7	12	2	960	26	6	1	5000
8	6	2	960	27	6	1	5000
9	6	2	960	28	12	1	5000
10	6	2	960	29	6	1	5000
11	6	2	960	30	6	1	5000
12	6	2	960	31	12	1	5000
13	6	1	960	32	6	1	1000
14	6	1	960	33	6	2	1000
15	6	1	960	34	6	1	500
16	6	2	1000	35	42	1	3000
17	18	1	5000	36	6	1	4000
18	6	1	5000	37	6	1	4000
19	12	1	5000	38	12	1	2000

Tabla 16 - Mejores soluciones tarjeta tipo Cha.

Réplica	*Colonia*	*Hormiga*	*F. Objetivo*
1	1	34	4058
1	2	1	4088
1	3	8	4037
1	4	4	3980
1	5	25	4025
1	6	21	3977
1	7	48	3938
1	8	60	3929
1	9	78	3875
1	10	77	3917
1	11	44	3932
1	12	29	3953
1	13	77	3935
1	14	21	3986
1	15	14	3992
1	16	4	3968
1	17	27	3977
1	18	25	3977
1	19	21	3950
1	20	5	3971

Tabla 17 - Configuración propuesta tarjeta tipo Cha.

Réplica	*Colonia*	*Hormiga*	*Componente*	*Posición*	*Unidades*	*Distancia*
1	9	78	35	1	14	1
1	9	78	31	2	12	2
1	9	78	23	3	18	3
1	9	78	23	4	18	4
1	9	78	2	5	6	5
1	9	78	25	6	12	6
1	9	78	38	7	12	7
1	9	78	24	8	6	8
1	9	78	3	9	6	9
1	9	78	4	10	6	10
1	9	78	6	11	6	11
1	9	78	1	12	6	13
1	9	78	1	13	6	15
1	9	78	11	14	6	17
1	9	78	13	15	6	18
1	9	78	15	16	6	19
1	9	78	34	17	3	20
1	9	78	32	18	6	21
1	9	78	16	19	6	23
1	9	78	29	20	6	24
1	9	78	33	21	6	26
1	9	78	7	22	6	28
1	9	78	36	23	6	29
1	9	78	35	38	14	1
1	9	78	35	39	14	2
1	9	78	17	40	18	3
1	9	78	14	41	6	4
1	9	78	2	42	6	5
1	9	78	19	43	12	6
1	9	78	28	44	12	7
1	9	78	7	45	6	9
1	9	78	37	46	6	10
1	9	78	18	47	6	11
1	9	78	20	48	6	12
1	9	78	22	49	6	13
1	9	78	5	50	6	14
1	9	78	12	51	6	16
1	9	78	10	52	6	18
1	9	78	8	53	6	20
1	9	78	34	54	3	21
1	9	78	21	55	6	22
1	9	78	30	56	6	23
1	9	78	27	57	6	24
1	9	78	9	58	6	26
1	9	78	26	59	6	27

Tabla 18 - Tarjeta tipo Dri.

Producto Dri

Componente	b_i	w_i	v_i	*Componente*	b_i	w_i	v_i
1	6	3	960	23	6	1	5000
2	6	1	960	24	6	1	5000
3	6	1	960	25	12	1	5000
4	6	3	960	26	6	1	5000
5	12	1	960	27	6	1	5000
6	12	1	960	28	18	1	5000
7	6	1	960	29	6	1	5000
8	6	1	960	30	6	1	5000
9	6	1	960	31	12	1	5000
10	6	1	960	32	18	1	5000
11	6	1	960	33	24	1	5000
12	6	1	960	34	6	1	5000
13	6	1	960	35	6	1	5000
14	6	1	960	36	6	3	1008
15	6	2	1000	37	6	1	1000
16	6	1	5000	38	6	1	1000
17	6	1	5000	39	6	2	2500
18	12	1	5000	40	12	1	4000
19	18	1	5000	41	24	1	4000
20	6	1	5000	42	12	1	4000
21	6	1	5000	43	12	1	4000
22	6	1	5000	44	24	1	3000

Tabla 19 - Mejores soluciones tarjeta tipo Cha.

Réplica	*Colonia*	*Hormiga*	*F. Objetivo*
1	1	47	4164
1	2	41	4176
1	3	13	4164
1	4	30	4074
1	5	37	4062
1	6	73	4038
1	7	26	4032
1	8	29	4062
1	9	36	4074
1	10	3	4080
1	11	77	4050
1	12	71	4050
1	13	5	4038
1	14	6	4032
1	15	51	4014
1	16	16	4014
1	17	11	4002
1	18	69	3996
1	19	45	3996
1	20	17	3984

Tabla 20 - Configuración propuesta tarjeta tipo Cha.

Réplica	*Colonia*	*Hormiga*	*Componente*	*Posición*	*Unidades*	*Distancia*
1	20	17	41	1	24	1
1	20	17	44	2	12	2
1	20	17	19	3	18	3
1	20	17	32	4	18	4
1	20	17	5	5	6	5
1	20	17	6	6	6	6
1	20	17	18	7	12	7
1	20	17	40	8	12	8
1	20	17	42	9	12	9
1	20	17	2	10	6	10
1	20	17	8	11	6	11
1	20	17	34	12	6	12
1	20	17	14	13	6	13
1	20	17	23	14	6	14
1	20	17	20	15	6	15
1	20	17	24	16	6	16
1	20	17	27	17	6	17
1	20	17	10	18	6	18
1	20	17	15	19	6	20
1	20	17	13	20	6	21
1	20	17	37	21	6	22
1	20	17	1	22	6	25
1	20	17	36	23	6	28
1	20	17	33	38	24	1
1	20	17	44	39	12	2
1	20	17	28	40	18	3
1	20	17	43	41	12	4
1	20	17	5	42	6	5
1	20	17	6	43	6	6
1	20	17	25	44	12	7
1	20	17	31	45	12	8
1	20	17	3	46	6	9
1	20	17	16	47	6	10
1	20	17	7	48	6	11
1	20	17	9	49	6	12
1	20	17	11	50	6	13
1	20	17	12	51	6	14
1	20	17	26	52	6	15
1	20	17	38	53	6	16
1	20	17	22	54	6	17
1	20	17	29	55	6	18
1	20	17	17	56	6	19
1	20	17	35	57	6	20
1	20	17	21	58	6	21
1	20	17	30	59	6	22
1	20	17	39	60	6	24
1	20	17	4	61	6	27

5 Conclusiones y Trabajo Futuro

5.1 Conclusiones

En este proyecto de tesis se utilizan las ideas generales de los meta heurísticos de los "Sistemas de Hormigas" para desarrollar una rutina heurística que ayuda a encontrar la configuración inicial que minimice el tiempo de recogido para una máquina de recogido y depósito Fuji IP III, y que puede ser utilizada para los productos que actualmente se producen en la "Fábrica Modelo".

Considerando la importancia que tiene la selección de los parámetros en la implementación del heurístico, se ha propuesto una metodología para la estimación de los mismos, de tal forma que se garantice la construcción consistente de buenas soluciones, tratando de minimizar el efecto negativo de las fluctuaciones aleatorias.

Otras conclusiones relevantes se presentan a continuación.

- En un mundo altamente competitivo en el que los procesos automatizados dominan los sistemas de producción, la disminución en los tiempos de procesamiento se constituye en un tema fundamental dentro de la optimización de procesos. Es especialmente relevante en industrias que incorporan el uso de Tecnología de Montaje de Superficie dentro de su producción.
- Se ha descrito la teoría asociada a los "Sistemas de Hormigas" como una herramienta alternativa para resolver problemas combinatorios de optimización.
- La minimización del tiempo necesario para completar la instalación de una tarjeta en una máquina de recogido y depósito, se plantea como el resultado de resolver dos problemas secuenciales de optimización, a diferencia del enfoque tradicional en el cual contemplan tres. El primero de ellos corresponde a encontrar una configuración óptima para la ubicación de los carretes de componentes y herramientas de instalación

en el espacio físico de almacenamiento teniendo en cuenta un requerimiento de producción o minimización del tiempo de recogido, mientras que el segundo corresponde al diseño de una secuencia óptima de instalación dada los resultados obtenidos en el primer problema.

- Se planteo una formulación matemática para la máquina de recogido y depósito Fuji IP III para encontrar la configuración óptima que minimiza el tiempo de recogido de los componentes requeridos para completar una tarjeta. Dicha formulación ha sido utilizada para solucionar un problema ficticio de características similares a los productos que actualmente se producen en la "Fábrica Modelo". Los resultados obtenidos han confirmado la validez del modelo de optimización propuesto.
- Tomando como referencia la tarjeta ficticia con la cual fue probado el modelo de formulación propuesto, después de una hora de ejecución el mejor valor factible encontrado por Lingo 8.0 (el procedimiento de *"branco and bound"* no reportó factibilidad) para el Caso 1, fue el mismo que el reportado por el procedimiento heurístico desarrollado, y el cual tuvo un tiempo de ejecución de 4 segundos.
- Al analizar el Caso 2 sobre este mismo modelo pequeño de prueba, el valor reportado como factible por Lingo 8.0 después de una hora de ejecución es menor solamente en un 2.5% con respecto al heurístico propuesto, después de un tiempo de corrida de solamente 6 minutos. Adicionalmente, al verificar la validez de la solución obtenida por Lingo 8.0, se verifico el incumplimiento de un conjunto de restricciones.
- La utilizar el modelo de optimización propuesto para la producción de una tarjeta real de la "Fábrica Modelo", se ha verificado el alto costo de tiempo computacional en el cual se incurre si se requiere la solución óptima. Luego de invertir más de 24 horas de tiempo de ejecución, un software especializado en optimización (Lingo 8.0) no pudo encontrar si quiera una solución factible para el problema.
- Dentro de la ejecución del diseño de experimento propuesto, se verificó la variabilidad que se presenta en los resultados obtenidos por el heurístico a diferentes condiciones experimentales. Hecho que ratifica la sensibilidad del procedimiento a

los parámetros de inicialización, y la necesidad de hacer un análisis particular de cada instancia, con el propósito de estimar los parámetros más convenientes. No solamente tienen efecto en las respuesta los efectos principales, si no que las interacciones entre ellos juegan un papel importante.

- Se desarrolló una estrategia de análisis de resultados para la estimación de los parámetros que deben ser utilizados con el propósito de garantizar buenas soluciones consistentemente a una instancia particular.
- La utilización de una sola variable de respuesta para la estimación de los parámetros de corrida, se constituye en un error al desconocer el aspecto de consistencia deseada de los resultados del heurístico. Dadas las características aleatorias de ciertas decisiones, es posible que la obtención de una muy buena solución para una condición experimental sea el producto de esta aleatoriedad y en realidad no refleje la tendencia de los resultados obtenidos para dicha condición. Bajo estas consideraciones, la metodología propuesta para la estimación de los parámetros óptimos, cumple con el propósito de tener en cuenta las características de las distribuciones empíricas producidas por cada condición experimental y hace la minimización de todos los resultados generados.
- Aplicando el análisis propuesto para dicha estimación, se hace la recomendación de los parámetros a ser utilizados en el heurístico propuesto para encontrar una buena configuración para una de las tarjetas que se hacen actualmente en la “Fábrica Modelo”.
- Se presentaran los resultados para el Caso 1 del heurístico desarrollado para las tres tarjetas que se producen en la “Fábrica Modelo” dados unos requerimientos mínimos de producción, utilizando los parámetros estimados para uno de las tarjetas.

5.2 Trabajo Futuro

Una vez desarrollado este proyecto de tesis, se presentan los siguientes temas que deberían ser considerados para un trabajo futuro, y de esta forma continuar con la línea de investigación desarrollada.

- Dadas las características actuales de la "Fábrica Modelo", es decir, la existencia de dos máquinas Fuji IP III, el contemplar en la formulación la asignación de componentes a máquinas, es el paso obligado para hacer más totalmente operable el heurístico.
- Hacer un análisis detallado del comportamiento de los dos casos propuestos en el heurístico, de tal forma que se pueda determinar el impacto de tener en cuenta o no el problema de asignación de boquillas. En este mismo sentido, valdría la pena el explorar otras opciones para integrar en el análisis esta asignación de boquillas.
- Una vez determinado que el proceso de optimización de la tecnología de SMT se puede dividir en la solución de dos problemas secuenciales, se debe continuar con el utilizar los resultados del primer problema (heurístico propuesto) como datos de entrada la obtención de una secuencia óptima de instalación.
- Considerando las bondades ofrecidas por la familia de meta heurísticos de los "Sistemas de Hormigas" para la resolución de problemas combinatorios de optimización, se crea la expectativa de desarrollar una rutina heurística que pueda resolver el problema de secuenciación óptima de instalaciones y cambios de boquillas.
- La prueba de los resultados obtenidos por los heurísticos desarrollados, frente a las configuraciones que en la actualidad se utilizan en la "Fábrica Modelo".

Referencias

Ahmadi, J., S. Grotzinger, y D. Johnson (1990) “Component allocation and partitioning for a dual delivery placement machine”. *Operations Research* (36): 176- 191.

Ahmadi, R. y H. Mamer (1999) “Routing heuristics for automated pick and place machines”. *European Journal of Operational Research.* Vol. 117(3): 533.

Altinkemer, K., B. Kazaz, M. Koksalan y H. Moskowitz (2000) “Optimization of printed circuit board manufacturing: Integrated modeling and algorithms”. *European Journal of Operational Research*, 124: 409-421.

Ayob, M., P. Cowling y G. Kendall (2002) “Optimization for surface mount placement machines”. *Proceedings of the IEEE ICIT’02, Bangkok, 11-14 December 2002*: 498-503.

Deneubourg, J.L. y S. Goss (1989) *"Collective patterns and decision-making"*. *Ethology, Ecology & Evolution*, Vol.1, 295–311.

Deneubourg, J.L., J.M. Pasteels y J.C. Verhaeghe (1983). "Probabilistic behavior in ants: a strategy of errors?". *Journal of Theoretical Biology*, 105, 259–271.

Dorigo, M. (1992). “Optimization, learning and natural algorithms”. Tesis Ph.D. Politécnico de Milano, Italia.

Dorigo, M. y L.M. Gambardella (1997). “Ant colonies for the traveling salesman problem”. *BioSystems*. 43: 73–81.

Dorigo, M., V. Maniezzo y A. Colorni (1996). “Ant System: Optimization by a colony of cooperating agent”. *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, 26(1): 29-41.

Dorigo M. y T. Stützle (2004) “Ant colony optimization”. Cambridge, MA: MIT Press/Bradford Books.

Montgomery D. (2006) “Design and Analysis of Experiments” 6ta Edición. Arizona State University. 660 páginas.

Tirpak, T.M., P.C. Nelson y A.J. Aswani (2000) “Optimization of revolver head SMT machines using adaptive simulated annealing”. *Electronics Manufacturing Technology Symposium – IEEE* Oct 2003: 214-220.

Whitley, D., T. Starkweather, D.Fuquay (1989) "Scheduling problems and traveling salesman: the genetic edge recombination operator," *Proceedings of the Third International Conference on Genetic Algorithms*.

Apéndice A

Apéndice A.1 Caso 1 – Modelo sin Boquillas

Apéndice A.1.1 Función Principal

```
clc;
clear all;
% DEFINICION DE LOS PARAMETROS DEL MODELO.
% datos.txt [ Número_de_componentes  3 ]
% Se encuentra la información asociada al tipo de tarjeta a producir, al igual que las características de los % tipos de carretes (uno por componente). La primera columna corresponde al número de componentes que % pueden almacenarse en cada carrete. La segunda columna se asocia al número de componentes requeridos por cada tipo para completar una única tarjeta. Mientras que la tercera columna indica el número de ranuras estándar que utiliza cada tipo de carrete.
        load datos1.txt;
% Vector con los requerimientos por tipo de componente en una tarjeta.
        b = datos1(:,2);
% Vector con la capacidad de almacenaje por tipo de carrete.
        v = datos1(:,1);
% Vector con los requerimientos de espacio por tipo de carrete.
        w = datos1(:,3);
% C.txt [ Número_de_componentes   Número_de_Nozzles]
% Matriz binaria en la cual se refleja la correspondencia de cada nozzle para la instalación de cada tipo de componente. Si C(i,j)=1, indica que el componente i puede ser instalado utilizando el nozzle j, por el contrario si C(i,j)=0, indica que el componente i no puede ser instalado utilizando el nozzle j.
        load C.txt;
% El número de tipos de componentes(i) y de nozzles(j) son extraidos.
        [ I J ] = size(C);
% L = Número de ranuras estandar disponibles en la máquina.
        L = 74;
% W = Número de ranuras estandar disponibles en cada lado de la máquina.
        W = L/2;
% or = Vector de la forma [ 1 2 ..... W 1 2 ..... W ] asociado al orden de los carretes a cada lado. Se interpretar como la preferencia por seleccionar algunos de estos indices de orden, donde 1 es el mejor y W corresponde al peor de los ordenes. Notese que los primeros W indices de orden se asocian a un lado, mientras que los restantes W se asocian al otro.
        or = [linspace(1,W,W) linspace(1,W,W)];
% Z = Número máximo de nozzles que se pueden instalar en cada lado es Z.
        Z = 6;
% nt = Tiempo requerido para hacer un cambio de nozzle esta dado por nt (notese que este tiempo en mayor que 2*W).
        nt = 80;
% N = Número mínimo de tarjetas que deben ser completadas con un setup (este corresponde a un parámetro que puede ser modificado).
        N = 100;
% K = Número de lados en la máquina.
        K = 2;
% DEFINICIÓN DE PARAMETROS GENERALES DEL HEURISTICO.
% rep = Número de réplicas del heurístico.
```

```
        Rep = 1;
% col = Número de colonias generadas en cada réplica.
        col = 2;
% hor = Número de hormigas que componen cada colonia.
        hor = 50;
% Alpha = Parámetro de importancia del rastro de feromonas.
        alpha = 1.5;
% Beta = Parámetro de importancia de la información heurística.
        beta = 0.8;
% ro = Valor asociado al coeficiente de evaporación entre cada colonia.
        ro = 0.5;
% q = Valor asociada a la probabilidad de mejorar las mejores soluciones ya encontradas, frente a la búsqueda de nuevas soluciones factibles.
        q = 0.6;
 % DEFINICION DE LAS VARIABLES DEL MODELO.
% y = [rep, col, hor, I, L]; Matriz 5D en la cual y(rep, col,hor, I, L)=1, significa que la hormiga  hor, de la colonia col, en la réplica rep, asigna un carrete de tipo I en el orden L. 0 de lo contrario. Se toma L como dimensión, dado que es el índice máximo de orden que podría ser asignado.
        y = zeros(rep, col, hor,I,L);
% x = [rep, col, hor, I, L]; Matriz 5D en la cual x(rep, col,hor, I, L)=n, significa que la hormiga hor, de la colonia col, en la réplica rep, tomara n componentes del carrete tipo I asignado en el orden L. 0 de lo contrario. Se toma L como dimensión, dado que es el índice máximo de orden que podría ser asignado.
        x = zeros(rep, col, hor,I,L);
% t = [rep, col, hor, J, K]; Matriz 5D en la cual t(rep, col, hor, J, K)=1, significa que la hormiga hor, de la colonia col, en la réplica rep, asigno el nozzle tipo J en el lado K de la máquina.
        t = zeros(rep, col, hor,J,K);
% VALORES HEURISTICOS Y RASTROS INICIALES DE FEROMONAS
        hsc = b;
        fsc0 = (1/I)*ones(I,1);
        hsr = ones(1,L);
        for c = 1:L;
                hsr(1,c) = (hsr(1,c))/(or(1,c));
        end;
        fsr0 = ones(I,L);
        for m = 1:I;
                for n = 1:L;
                        fsr0(m,n) = (fsr0(m,n))/(b(m,1)*or(1,n));
                end;
        end;
%VERIFICACION DEL NUMERO DE TARJETAS REQUERIDAS Y DISPONIBILIDAD DE RANURAS
%Antes de utilizar el modelo es necesario el verificar si el número de tarjetas a ser completadas con una única configuración, puede ser alcanzado dadas las restricciones del número de ranuras estándar disponibles. En caso de ser factible, el vector fila minc [1,I] indicará el número de carretes de tipo I que deben ser instalados en la máquina. Para este proposito se utiliza la función res_tarjetas. flag = valor de factibilidad, Si flag=1 el problema no es factible. 0 de lo contrario.
% N_tarjetas = Indica el número de tarjetas que podrán ser completadas con una única configuración.
% minc = Vector, donde minc(i)=n, se interpreta como que se deben asignar un total de n carretes de tipo i.
        [flag N_tarjetas minc] = res_tarjetas(b, v, w, I, L, N);
        if flag == 1;
                flag;
                fprintf(1,'El problema no es factible. No es posible completar el número\n');
                 fprintf(1,'de tarjetas requeridas, dada la restricción de ranuras estandar\n');
```

```
                return;
        end;
% IMPLEMENTACION DEL HEURISTICO
% asig = [ no_definido 6]; asig = ( r co ant i l x(r, co, ant,i,l))
% Matriz en dos dimensiones, donde cada fila representa la asignación de un orden a un componente, la
columna 1 corresponde a la replica, columna 2 la colonia, columna 3 a la hormiga, columna 4 al componente,
columna 5 al orden y columna 6 al número de unidades a ser accesadas desde dicha ubicación
        asig = [];
% val_dis = [ r*co*ant 4]; val_dis = ( r co ant f )
% Matriz en dos dimensiones, donde cada fila representa la solución encontrada por cada hormiga. La columna
1 es la replica, la columna 2 es colonia, la columna 3 es la hormiga, y la columna 4 el valor de la
% función objetivo.
        val_dis = [];
% mejorhormiga = [ ro*co 4 ]; mejorhormiga = ( r co ant f )
% Matriz en dos dimensiones, donde cada fila representa la mejor solución encontrada por cada colonia. La
columna 1 es la replica, la columna 2 es colonia, la columna 3 es la hormiga, y la columna 4 el valor de la
% función objetivo.
        mejorhormiga = [];
        infactible = 0;
% Procedimiento iterativo hasta completar el total de réplicas.
for r = 1:rep;
   r
   % Inicialización de los rastros de feromonas para cada réplica
   fsc = fsc0;
   fsr = fsr0;
   % Procedimiento iterativo hasta completar el total de colonias.
   for co = 1: col;
      co
      val_dis_co = [];
      % Procedimiento iterativo hasta completar el total de hormigas independientes.
      for ant = 1:hor;
         ant;
         asig_ant =[];
         % components = Vector con los índices asociados a los componentes que no han sido instalados.
         components = linspace(1,I,I);
         [m,n] = size(components);
         % orden = Vector con los índices asociados a los ordenes que no han sido asociados a componentes.
         orden = linspace(1,2*W,2*W);
         %CICLO DE PARA LA INSTALACION DE CADA TIPO DE COMPONENTE.
           %Esta es una rutina cíclica que se ejecutará hasta tanto el vector de índices asociados a %componentes
           "components" este vacio.
         %pr = Contador de referencia para verificar cuantos componentes
         %     se han asignado.
         pr = 0;
         while n >= 1;
            pr = pr+1;
           % aux = Su primera componente indica el componente seleccionado, y continuamente
           % el nuevo vector "components" (se ha retirado dicho componente).
            aux = selec_componente(components,hsc,fsc,alpha,beta,q);
            % i = Indice asociado al componente seleccionado.
            i = aux(1,1);
            % Se actualiza el vector de índices "components" al retirar el componente seleccionado.
```

```
        % n = total de componentes que no se han seleccionado.
        [m,n] = size(aux);
        components = aux(1,2:n);
        [m,n] = size(components);
        % s = número de los "orden" que no se han asignado.
        [r1,s] = size(orden);
        % ASIGNACION DE ORDEN A LOS CARRETES QUE DEBEN INSTALARSE
        % DEL COMPONENTE SELECCIONADO
        % car = número de carretes del componente seleccionado que deben instalarse.
        car = 0;
        car = minc(i);
        % Ciclo de asignación de orden, mientras existan carretes son asignación.
        while car > 0;
          aux = selec_orden(i,hsr,fsr,orden,alpha,beta,q);
          % l = orden que se le asigna al carrete de tipo i.
          l = aux(1,1);
          % Se actualiza el vector de orden.
          [r,s] = size(aux);
          orden = aux(1,2:s);
          const = b(i)/minc(i);
          asig_ant = [asig_ant ; r co ant i l const w(i)];
          car = car - 1;
        end;
      end;
      % Función para tener ordenes consecutivos de asignación y el llenar las matrices "x" y "y". En el
      % mismo sentido esta función es útil para determinar la primera ranura que ocupa cada carrete.
      [asig_ant x y] = mejorarelorden(asig_ant,r,co,ant,x,y,L);
      % La estructura de asig_ant ahora es: asig_ant = [r co ant i l const ranura_mas_distante]
      asig = [ asig; asig_ant ];
      % aux1 = Valor de la función objetivo (distancia).
      % flag = Factor de factibilidad (0=factible, 1=infactible).
      [aux1, flag] = calculardistancia(asig_ant,L);
      % Infactible = Contador del número de soluciones infactibles generadas.
      infactible = infactible + flag;
      % En la matriz val_dis_co se almacena la información general de cada hormiga que se ha generado.
      % La replica, la colonia, el número de hormiga, el valor de la función y si es factible o no.
      val_dis_co = [ val_dis_co; r co ant aux1 flag];
    end;
    % La matriz val_dis se almacena la información de todas las hormigas generadas.
    val_dis = [val_dis; val_dis_co];
    % mejores5hormigas = Devuelve el número de la hormiga, el valor de la función y si es factible o no.
    [ mejores5hormigas ] = encontrar_mejores5hormigas(val_dis_co);
    % mejorhormiga = Corresponde a la mejor solución factible encontrada en cada una de las colonias.
    aux1 = mejores5hormigas(1,:);
    mejorhormiga = [ mejorhormiga; r co aux1(1,1) aux1(1,2) aux1(1,3)];
    % Actualización del rastro de feromonas para la seleccion de ranuras una vez que se ha fijado un tipo
    % de componente a asignar.
    delta_fsr = [];
    maxfsr=max(max(fsr));
    delta_fsr = update_fsr(r,co,hor,mejores5hormigas,y,maxfsr);
    fsr = fsr*(1-ro) + delta_fsr;
end;
```

```
   % Se presenta el vector con las mejores soluciones encontradas en cada colonia.
   mejorhormiga
   mejorsolucion = min(mejorhormiga(:,4))
   % Total y porcentaje de hormigas infactibles que fueron generadas.
   Porcentaje_infactible = 100*(infactible/(rep*col*hor))
end;
```

Apéndice A.1.2 Funciones Utilizadas

```
function [flag N_tarjetas minc] = res_tarjetas(b, v, w, I, L, N)
% flag = valor de factibilidad, Si flag=1 el problema no es factible. 0 de lo contrario.
% N_tarjetas = Indica el número de tarjetas que podrán ser completadas con una única configuración.
% minc = Vector, donde minc(i)=n, se interpreta como que se deben asignar un total de n carretes de tipo i.
% I = Total de componentes que hay que asignar.
% L = Total de ranuras estandar disponibles.
% b = Vector columna con los requirimientos por tipo de componente.
% v = Vector columna con el número de componentes que se pueden almacenar por tipo de carrete.
% w = Vector columna con el número de ranuras estandar utilizadas por cada  tipo de carrete.
flag = 0;
N_tarjetas = 0;
minc = zeros(1,I);
% Si solamente se requiere un componente por board, se debe asignar solamente un carrete.
for i = 1:I;
   aux = 0;
   % Si solamente se requiere un componente por board, entonces se ha asignar un único carrete de dicho tipo.
   if b(i)==1;
      minc(i) = 1;
   else;
      % Aux = Número entero mínimo no factible de carretes de tipo i que deben ser utilizados.
      aux = ceil(N*b(i)/v(i));
      % Con el propósito de encontrar un valor factible para aux, se prosigue a buscar el siguiente valor
      % entero que sea divisor del número de unidades utilizadas para completar una tarjeta.
      while ( mod( b(i),aux) > 0);
         aux = aux+1;
      end;
      minc(i) = aux;
   end;
end;
% Una vez se ha calculado el número de carretes de cada tipo requeridos para completar un mínimo de N
tarjetas, se prosigue a calcular el número de tarjetas realmente pueden ser completadas en dicha  configuración
y el número de ranuras estandar que son requeridas para dicha configuración.
aux = minc.*v';
aux1 = aux./b';
aux1 = floor(aux1);
N_tarjetas= min(aux1);
N_ranurasutilizadas= sum(minc.*w');

% Se verifica si el número de ranuras requeridas supera el número de tarjetas disponibles y se activa la señal de
infactibilidad. De lo contrario se retorna la señal de factibilidad, el número de ranuras estándar utilizadas y el
número de carretes a instalar por cada tipo de componente.
if (N_ranurasutilizadas > L);
   flag = 1;
```

```
   N_tarjetas= 0;
end;
return;

function [components_new]=selec_componente(components,hsc,fsc,alpha,beta,q)
   %components = vector de indices asociados a componentes sin instalarse.
   %hsc = vector de valores heuristicos para la selección de componente.
   %fsc = vector de rastro de feromonas para la selección de componente.
   %hsc = [ I 1 ] (vector columna)
   %fsc = [ I 1 ] (vector columna)
   %alpha = valor de importancia para el rastro de feromonas.
   %beta = valor de importancia para el valor heuristico.
   %q = probabilidad de mejora de buenas soluciones encontradas.
   [m, n] = size(components).
   % q0 = Variable aleatoria para determinar si se busca nuevas soluciones o se mejoran las buenas previamente
encontradas.
   q0 = rand(1);
   aux = [];
   if q0 <= q;
      % Mejorar buenas soluciones encontradas previamente.
      aux = argumax_c(components,hsc,fsc,alpha,beta,n);
   else;
      % Búsqueda de nuevas soluciones
      aux = probabilityc(components,hsc,fsc,alpha,beta,n);
   end;
   % k = corresponde a la posición en el vector "components" del índice que fue seleccionado a ser instalado.
   k = aux(1,1);
   % i = corresponde al índice del componente a ser instalado.
   i = components(k);
   % Se retira del vector "components" el indice que fue seleccionado.
   components1 = [components(1,1:k-1) components(1,k+1:n)];
   % Se organiza el vector respuesta.
   components_new = [i components1];
return

function [components_new] = argumax_c(components,h,f,alpha,beta,n);
% components = vector de índices asociados a componente a ser instalados.
% h = valor heuristico.
% f = rastro de feromonas.
% alpha = importancia del rastro de feromonas.
% beta = importancia del valor heurístico.
% n = dimensión del vector "components".
arg = zeros(n,1);
% Se completa el vector arg con los valores = (f(i)^alpha)*(h(i)^beta).
for i = 1:n;
   arg(i,1) = ((f(components(1,i),1))^alpha)*((h(components(1,i),1))^beta);
end;
% Se encuentra la posición del mayor de todos los argumentos.
% i = valor máximo de los argumentos.
% j = posición en la cual se encuentra ese valor máximo.
[i j] = max(arg);
% Se prepara el vector respuesta.
```

```
components_new = [j components];
return;

function [components_new] = probabilityc(components,h,f,alpha,beta,n);
% components = vector de índices asociados a componente a ser instalados.
% h = valor heuristico.
% f = rastro de feromonas.
% alpha = importancia del rastro de feromonas.
% beta = importancia del valor heurístico.
% n = dimensión del vector "components".
% Se quiere generar un vector p con las probabilidades de seleccionar un componente en particular. Básicamente es generar un CDF.
p = zeros(n,1);
for i = 1:n;
      d = (f(components(1,i),1))^alpha;
      e = (h(components(1,i),1))^beta;
      p(i,1) = d*e;
end;
sump = sum(p);
for i = 1:n;
      p(i,1) = p(i,1)/sump;
end;
for i = 2:n;
      p(i,1) = p(i-1,1)+p(i,1);
end;
% aleatorio = valor entre 0 y 1 para seleccionar el componente.
aleatorio = rand(1);
flag=0;
j = 0;
while flag==0;
   j = j+1;
   if aleatorio<= p(j,1);
         flag=1;
   end;
end;
% j = corresponde al índice del componente que fue seleccionado.
components_new = [j components];
return;

function [components] = retirar_colocados(components,components_nozzles);
aux = components;
[a b] = size(components);
[d e] = size(components_nozzles);
for i = 1:e;
  d = components_nozzles(1,i);
   pos = find(aux==d);
   aux1 = aux;
   aux = [ aux1(1,1:pos-1) aux(1,pos+1:b) ];
end;
components=aux;
return;
```

```
function [orden_new]=selec_orden(i,hsr,fsr,orden,alpha,beta,q)
   %i = Indice asociado al componente seleccionado a instalarse.
   %hsr = vector de valores heuristicos para la selección de orden.
   %fsr = vector de rastro de feromonas para la selección de orden.
   %orden = vector con los índices asociados a ordenes disponibles.
   %hsr = [ 1 L ] (vector fila)
   %fsr = [ I, L ]
   %alpha = valor de importancia para el rastro de feromonas.
   %beta = valor de importancia para el valor heuristico.
   %q = probabilidad de mejora de buenas soluciones encontradas.
   % q0 = variable aleatoria que ayuda a determinar si se hace mejora de buenas soluciones o la búsqueda de nuevas soluciones
   [ m n ] = size(orden);
   q0 = rand(1);
   aux = [];
   if q0 <= q;
      % Mejora de buenas soluciones encontradas previamente.
      aux = argumax_r1(i,orden,hsr,fsr,alpha,beta);
   else;
      % Búsqueda de nuevas soluciones.
      aux = probabilityr1(i,orden,hsr,fsr,alpha,beta);
   end;
   % k = Posición del índice asociado al orden.
   k = aux(1,1);
   % l = Orden que se le ha asignado a un carrete del componente tipo i.
   l = orden(k);
   % Se actualiza el vector de índices asociados al orden.
   orden1 = [orden(1,1:k-1) orden(1,k+1:n)];
   % Se prepara el vector de respuesta, donde la primera componente es el orden asignado, y lo siguiente es el vector ordenes no asignados aún.
   orden_new = [l orden1];
return

function [orden_new] = argumax_r1(i,orden,hsr,fsr,alpha,beta);
   %i = Indice asociado al componente seleccionado a instalarse.
   %hsr = vector de valores heuristicos para la selección de orden.
   %fsr = vector de rastro de feromonas para la selección de orden.
   %hsr = [ 1 L ] (vector fila)
   %fsr = [ I, L ]
   %alpha = valor de importancia para el rastro de feromonas.
   %beta = valor de importancia para el valor heuristico.
[m n] = size(orden);
arg = zeros(n,1);
% arg = vector de la forma arg(i)= (f(i)^alpha)*(h(i)^beta).
for j = 1:n;
   arg(j,1) = ((fsr(i,orden(1,j)))^alpha)*((hsr(1,orden(1,j)))^beta);
end;
% i = Mayor valor dentro del vector arg.
% j = Posición del mayor valor dentro del vector arg.
[i j] = max(arg);
% Se prepara el vector respuesta
orden_new = [j orden];
```

```
return;

function [orden_new] = probabilityr1(i,orden,hsr,fsr,alpha,beta);
% i = Indice del componente a ser instalado.
% h = valor heuristico.
% f = rastro de feromonas.
% alpha = importancia del rastro de feromonas.
% beta = importancia del valor heurístico.
% n = dimensión del vector "components".
% Se quiere generar un vector p con las probabilidades de seleccionar un componente en particular. Básicamente es generar un CDF.
[ m n ] = size(orden);
p = zeros(n,1);
for j = 1:n;
      d = ((fsr(i,orden(1,j)))^alpha);
      e = ((hsr(1,orden(1,j)))^beta);
      p(j,1) = d*e;
end;
sump = sum(p);
for j = 1:n;
      p(j,1) = p(j,1)/sump;
end;
for j = 2:n;
      p(j,1) = p(j-1,1)+p(j,1);
end;
% aleatorio = valor entre 0 y 1 para seleccionar el orden.
aleatorio = rand(1);
flag=0;
j = 0;
while flag==0;
   j = j+1;
   if aleatorio<= p(j,1);
         flag=1;
   end;
end;
% j = corresponde a la posición del índice del orden que fue seleccionado.
orden_new = [j orden];
return;

function [asig_ant x y] = mejorarelorden(asig_ant,r,co,ant,x,y,L);
% asig_ant = Vector con la asignacion de ordenes a cada componente establecidos por la hormiga en cuestión. La estructura de este vector es de la forma [r co ant i l u w].
% 5 = Columna en la cual se encuentra el orden asignado.
% x = Vector donde se asigna cuantas unidades se extraen desde cada orden.
% y = Vector donde se señala si el orden l o no al componente i.
% r = Corresponde a la réplica que se esta ejecutando.
% co = Corresponde a la colonia que se esta generando.
% ant = Corresponde a la hormiga que se acaba de seleccionar.
% L = Corresponde al orden máximo que podría ser asignado.
% Lo primero que se hace es organizar el vector asig_ant en forma ascendente con respecto al orden.
asig_ant = sortrows(asig_ant,5);
[ m n ] = size(asig_ant);
```

```
% Se realiza un ciclo para tener ordenes consecutivos en cada lado de la máquina
orden = 1;
ranura1 = 0;
for i = 1:m;
   if( asig_ant(i,5)<=(L/2));
      ranura1 = asig_ant(i,7)+ranura1;
      asig_ant(i,5) = orden;
      asig_ant(i,7) = ranura1;
      orden = orden+1;
   end;
end;
orden = (L/2)+1;
ranura2 = 0;
for i = 1:m;
   if( asig_ant(i,5)>(L/2));
      ranura2 = asig_ant(i,7)+ranura2;
      asig_ant(i,5) = orden;
      orden = orden+1;
      asig_ant(i,7) = ranura2;
   end;
end;
% Se realiza un ciclo para llenar las matrices "x" y "y".
for c = 1:m;
   i = asig_ant(c,4);
   l = asig_ant(c,5);
   u = asig_ant(c,6);
   x(r,co,ant,i,l)= u;
   y(r,co,ant,i,l)= 1;
end;
return;

function [aux1, flag] = calculardistancia(asig_ant,L);
% El propósito de esta función es el de calcular la distancia que recorre
% la hormiga en ir a recoger todos los componentes que se deben utilizar
% para completar un board. Adicionalmente se utilizará la señal flag para
% determinar si la solucion es factible o no (0 = no se sobrepasa el número
% ranuras estandar disponibles, 1 = se supera el número de ranutas estandar
% disponibles).
% asig_ant = [r co ant i l const ranura_mas_distante]
flag = 0;
[ m n ] = size(asig_ant);
% Se verifica cual es la ranura mas lejana que se ha asignado.
max_ranuras = max(asig_ant(:,7));
% En caso que la ranura mas lejana sobrepasa el límite se señala flag=1.
if max_ranuras > (L/2);
   flag = 1;
end;
% Se calcula el total de distancia recorrida.
aux1 = 0;
for i = 1:m;
   aux1 = (asig_ant(i,6))*(asig_ant(i,7))+aux1;
end;
```

```
return;

function[ mejores5hormigas ] = encontrar_mejores5hormigas(val_dis_co);
% Retorna las mejores 5 soluciones factibles que se han generado en la colonia. Retorna una columna con el número de la hormiga, el valor de la función objetivo y un indicador de factibilidad han generado en un colonia. Es decir, dice cuales son las hormigas y los correspondientes valores de la función objetivo.
%val_dis_co = la columna 3 corresponde a la hormiga, la 4 al valor de la
%función objetivo y la 5 a si es factible o no.
mejores5hormigas = [];
% aux = Matrix donde se guardaran las columnas 3 a 5 de las soluciones factibles.
aux = [];
[m n] = size(val_dis_co);
% Completar la matriz aux con las hormigas factibles solamente.
for i = 1:m;
   if val_dis_co(i,5)==0;
      aux = [aux; val_dis_co(i,3:n)];
   end;
end;
% Se ordenan ascendemente las hormigas de acuerdo al valor de la función objetivo.
aux = sortrows(aux,2);
% Se extrean las primeras 5 filas solamente.
mejores5hormigas = aux(1:5,:);
return;

function delta_fsr = update_fsr(r,co,hor,mejores5hormigas,y,maxfsr);
[f g] = size(mejores5hormigas);
e = size(y);
delta_fsr=zeros(e(4),e(5));
mindis = min(mejores5hormigas(:,2));
for h= 1:f;
   for i = 1:e(4);
      for l = 1:e(5);
         if y(r,co,mejores5hormigas(h,1),i,l) == 1;
            delta_fsr(i,l) = mindis/mejores5hormigas(h,2);
         end;
      end;
   end;
end;
maxdelta_fsr = max(max(delta_fsr));
delta_fsr= delta_fsr*(maxfsr)/maxdelta_fsr;
return;
```

Apéndice A.2 Caso 2 – Modelo con Boquillas

Apéndice A2.1 Función Principal

```
clc;
clear all;
```

```
% DEFINICION DE LOS PARAMETROS DEL MODELO.
% datos.txt [ Número_de_componentes  3 ]
% Se encuentra la información asociada al tipo de tarjeta a producir, al igual que las características de los tipos de carretes (uno por componente). La primera columna corresponde al número de componentes que pueden almacenarse en cada carrete. La segunda columna se asocia al  número de componentes requeridos por cada tipo para completar una única tarjeta. Mientras que la tercera columna indica el número de ranuras estandar que utiliza cada tipo de carrete
load datos1.txt;
% Vector con los requerimientos por tipo de componente en una tarjeta.
b = datos1(:,2);
% Vector con la capacidad de almacenaje por tipo de carrete.
v = datos1(:,1);
% Vector con los requerimientos de espacio por tipo de carrete.
w = datos1(:,3);
%w = ones(34,1);
% C.txt [ Número_de_componentes   Número_de_Nozzles]
load C.txt;
% El número de tipos de componentes(i) y de nozzles(j) son extraidos.
[ I J ] = size(C);
% L = Número de ranuras estandar disponibles en la máquina.
L = 74;
% W = Número de ranuras estandar disponibles en cada lado de la máquina.
W = L/2;
or = [linspace(1,W,W) linspace(1,W,W)];

% Z = Número máximo de nozzles que se pueden instalar en cada lado es Z.
Z = 6;
% nt = Tiempo requerido para hacer un cambio de nozzle esta dado por nt
% (notese que este tiempo en mayor que 2*W).
nt = 80;
% N = Número mínimo de tarjetas que deben ser completadas con un setup
% (este corresponde a un parámetro que puede ser modificado).
N = 100;
% K = Número de lados en la máquina.
K = 2;

% DEFINICIÓN DE PARAMETROS GENERALES DEL HEURISTICO.
% rep = Número de réplicas del heurístico.
rep = 1;
% col = Número de colonias generadas en cada réplica.
col = 20;
% hor = Número de hormigas que componen cada colonia.
hor = 50;
% Los valores se han tomado de acuerdo a la tabla 3.1 - Dorigo
% Alpha = Parámetro de importancia del rastro de feromonas.
alpha = 1.5;
% Beta = Parámetro de importancia de la información heurística.
beta = 0.8;
% ro = Valor asociado al coeficiente de evaporación entre cada colonia.
ro = 0.5;
% q = Valor asociada a la probabilidad de mejorar las mejores soluciones ya
% encontradas, frente a la búsqueda de nuevas soluciones factibles.
```

```
q = 0.6;

% DEFINICION DE LAS VARIABLES DEL MODELO.
y = zeros(rep, col, hor,I,L);
x = zeros(rep, col, hor,I,L);
a = zeros(rep, col, hor, I);
% t = [rep, col, hor, J, K]
% Matriz 5D en la cual t(rep, col, hor, J, K)=1, significa que la hormiga
% hor, de la colonia col, en la réplica rep, asigno el nozzle tipo J en el
% lado K de la máquina.
t = zeros(rep, col, hor,J,K);

% VALORES HEURISTICOS Y RASTROS INICIALES DE FEROMONAS
% 1. SELECCION DE NOZZLE.
%    VALOR HEURISTICO
hsn0 = zeros(1,J);
hsn0 = sum(C);
hsn0 = hsn0/max(hsn0);
%    RASTRO DE FEROMONAS
fsn0 = ones(1,J)*(1/J);

% 2. SELECCION DE COMPONENTES
%    VALOR HEURISTICO
hsc = b;
%    RASTRO DE FEROMONAS
fsc0 = (1/I)*ones(I,1);

% 3. ASIGNAR UN ORDEN AL COMPONENTE QUE ESTA SIENDO CONSIDERADO
%    VALOR HEURISTICO.
hsr = ones(1,L);
   for c = 1:L;
      hsr(1,c) = (hsr(1,c))/(or(1,c));
   end;
%   RASTRO DE FEROMONAS
fsr0 = ones(I,L);
   for m = 1:I;
      for n = 1:L;
         fsr0(m,n) = (fsr0(m,n))/(b(m,1)*or(1,n));
      end;
   end;

%VERIFICACION
% flag = valor de factibilidad, Si flag=1 el problema no es factible. 0 de
% lo contrario.
% N_tarjetas = Indica el número de tarjetas que podrán ser completadas con
% una única configuración.
% minc = Vector, donde minc(i)=n, se interpreta como que se deben asignar
% un total de n carretes de tipo i.
[flag N_tarjetas minc] = res_tarjetas(b, v, w, I, L, N);

if flag == 1;
   flag;
```

```
   fprintf(1,'El problema no es factible. No es posible completar el número\n');
   fprintf(1,'de tarjetas requeridas, dada la restricción de ranuras estandar\n');
   return;
end;

% IMPLEMENTACION DEL HEURISTICO
asig = [];
 val_dis = [];
 mejorhormiga = [];
infactible = 0;
% Procedimiento iterativo hasta completar el total de réplicas.
for r = 1:rep;
   r
   fsc = fsc0;
   fsr = fsr0;
   fsn = fsn0;
   for co = 1: col;
      co
      val_dis_co = [];
      for ant = 1:hor;
         ant;
         % Se inicializa el heuristico para la selección de nozzles.
         hsn = hsn0;
         % Se inicializa una matriz auxiliar para la actualización del heuristico de nozzles.
         C1 = C;
         asig_ant =[];
         % nozzles = Vector con los indices asociados a los nozzles disponibles para instalar componentes.
         nozzles = linspace(1,J,J);
         % components = Vector con los índices asociados a los componentes que no han sido instalados.
         components = linspace(1,I,I);
         [m,n] = size(components);
         % orden = Vector con los índices asociados a los ordenes que no  han sido asociados a componentes.
         orden = linspace(1,2*W,2*W);

         %CICLO DE PARA LA INSTALACION DE CADA TIPO DE COMPONENTE.
         %Esta es una rutina cíclica que se ejecutará hasta tanto el vector de indices asociados a componentes
"components" este vacio.
         %pr = Contador de referencia para verificar cuantos componentes  se han asignado.
         pr = 0;
         while n >= 1;
            aux = seleccion_nozzle(nozzles,hsn,fsn,alpha,beta,components,C1,q);
            [ u v ] = size(aux);
            j = aux(1,1);
            % components_nozzles = Corresponde a un vector con los indices asociados a los componentes
que se pueden instalar con esta herramienta.
            components_nozzles = aux(1,2:v);
            % Copia del vector con indices que se van a instalar con esta herramienta.
            components_nozzles1 = components_nozzles;
            % Se actuliza el vector de índices de nozzles que faltan por utilizarse.
            [ m o ] = size(nozzles);
            d = find(nozzles==j);
            nozzles1 = nozzles;
```

```
    nozzles = [ nozzles1(1,1:d-1) nozzles1(1,d+1:o)];
    while v > 0;
      [e  q3] = size(components_nozzles);
      if q3 == 0;
        components_nozzles = components;
      end;
      aux = selec_componente(components_nozzles,hsc,fsc,alpha,beta,q);
      % i = Indice asociado al componente seleccionado.
      i = aux(1,1);
      % Se actualiza el vector de índices "components" al retirar el componente seleccionado.
      % n = total de componentes que no se han seleccionado.
      [m,n] = size(aux);
      components_nozzles = aux(1,2:n);
      [m,v] = size(components_nozzles);
      % s = número de los "orden" que no se han asignado.
      [r1,s] = size(orden);

      % ASIGNACION DE ORDEN A LOS CARRETES QUE DEBEN INSTALARSE
      % DEL COMPONENTE SELECCIONADO
      % car = número de carretes del componente seleccionado que
      %       deben instalarse.
      car = 0;
      car = minc(i);
      % Ciclo de asignación de orden, mientras existan carretes son asignación.
      while car > 0;
        aux = selec_orden(i,hsr,fsr,orden,alpha,beta,q);
        % l = orden que se le asigna al carrete de tipo i.
        l = aux(1,1);
        % Se actualiza el vector de orden.
        [rr,s] = size(aux);
        orden = aux(1,2:s);
        const = b(i)/minc(i);
        asig_ant = [asig_ant ; r co ant i l const w(i)];
        car = car - 1;
        if l <= W;
          t(r,co,ant,j,1)=1;
        else;
          t(r,co,ant,j,2)=1;
        end
      end;
    end;
    components = retirar_colocados(components,components_nozzles1);
    % Se actualiza el heuristico para la selección de nozzles.
    C1 = actualizacion_C(C1,j,components_nozzles1);
    hsn = sum(C1);
    if max(hsn)>0;
      hsn = hsn/max(hsn);
    end;
    [m,n] = size(components);
  end;
```

% Función para tener ordenes consecutivos de asignación y el llenar las matrices "x" y "y". En el mismo sentido esta función es útil para determinar la primera ranura que ocupa cada carrete.

```
        [asig_ant x y] = mejorarelorden(asig_ant,r,co,ant,x,y,L);
        % La estructura de asig_ant ahora es: asig_ant = [r co ant i l const ranura_mas_distante]
        asig = [ asig; asig_ant ];
        % aux1 = Valor de la función objetivo (distancia).
        % flag = Factor de factibilidad (0=factible, 1=infactible).
        [aux1, flag] = calculardistancia(asig_ant,L);
        % Infactible = Contador del número de soluciones infactibles generadas.
        %[aux2, flag] = seleccion_nozzle(asig_ant,t,C,flag,);
        nozzle_value = sum(sum(t(r,co,ant,:,:)))*nt;
        infactible = infactible + flag;
        % aux = Valor de la función objetivo;
        aux = aux1 + nozzle_value;
        % En la matriz val_dis_co se almacena la información general de cada hormiga que se ha generado. La
replica, la colonia, el número de hormiga, el valor de la función y si es factible
        % o no.
        val_dis_co = [ val_dis_co; r co ant aux flag aux1 nozzle_value];
      end;
      % La matriz val_dis se almacena la información de todas las hormigas generadas.
      val_dis = [val_dis; val_dis_co]
      % mejores5hormigas = Debe retornar el número de la hormiga, el valor de la función y si es factible o no.
      [ mejores5hormigas ] = encontrar_mejores5hormigas(val_dis_co);
      % mejorhormiga = Corresponde a la mejor solución factible encontrada en cada una de las colonias.
      aux1 = mejores5hormigas(1,:);
      mejorhormiga = [ mejorhormiga; r co aux1(1,1) aux1(1,2) aux1(1,3)];
      % Actualización del rastro de feromonas para la seleccion de ranuras una vez que se ha fijado un tipo de
componente a asignar.
      delta_fsr = [];
      maxfsr=max(max(fsr));
      delta_fsr = update_fsr(r,co,hor,mejores5hormigas,y,maxfsr);
      fsr = fsr*(1-ro) + delta_fsr;

      % Actualización del rastro de feromonas para la selección de nozzles.
      delta_fsn = [];
      maxfsn = max(fsn);
      delta_fsn = update_fsn(r,co,hor,mejores5hormigas,t,maxfsn);
      fsn = fsn*(1-ro) + delta_fsn;
   end;
   % Se presenta el vector con las mejores soluciones encontradas en cada colonia.
   mejorhormiga
   mejorsolucion = min(mejorhormiga(:,4))
   % Total y porcentaje de hormigas infactibles que fueron generadas.
   infactible
   Porcentaje_infactible = 100*(infactible/(rep*col*hor))
end;
```

Apéndice A.2.2 Funciones Utilizadas

function [flag N_tarjetas minc] = res_tarjetas(b, v, w, I, L, N)

```
% flag = valor de factibilidad, Si flag=1 el problema no es factible. 0 de lo contrario.
% N_tarjetas = Indica el número de tarjetas que podrán ser completadas con una única configuración.
% minc = Vector, donde minc(i)=n, se interpreta como que se deben asignar un total de n carretes de tipo i.
% I = Total de componentes que hay que asignar.
```

```
% L = Total de ranuras estandar disponibles.
% b = Vector columna con los requirimientos por tipo de componente.
% v = Vector columna con el número de componentes que se pueden almacenar por tipo de carrete.
% w = Vector columna con el número de ranuras estandar utilizadas por cada tipo de carrete.
 flag = 0;
N_tarjetas = 0;
minc = zeros(1,I);
% Si solamente se requiere un componente por board, se debe asignar solamente un carrete.
for i = 1:I;
   aux = 0;
   % Si solamente se requiere un componente por board, entonces se ha asignar un único carrete de dicho tipo.
   if b(i)==1;
      minc(i) = 1;
   else;
      % Aux = Número entero mínimo no factible de carretes de tipo i que deben ser utilizados.
      aux = ceil(N*b(i)/v(i));
      % Con el propósito de encontrar un valor factible para aux, se prosigue a buscar el siguiente valor entero que sea divisor del número de unidades utilizadas para completar una tarjeta. consigue a buscar el siguiente valor entero
      while ( mod( b(i),aux) > 0);
         aux = aux+1;
      end;
      minc(i) = aux;
   end;
end;
% Una vez se ha calculado el número de carretes de cada tipo requeridos para completar un mínimo de N tarjetas, se prosigue a calcular el número de tarjetas realmente pueden ser completadas en dicha configuración y el número de ranuras estandar que son requeridas para dicha configuración.
aux = minc.*v';
aux1 = aux./b';
aux1 = floor(aux1);
N_tarjetas= min(aux1);
N_ranurasutilizadas= sum(minc.*w');
% Se verifica si el número de ranuras requeridas supera el número de tarjetas disponibles y se activa la señal de infactibilidad. De lo contrario se retorna la señal de factibilidad, el número de ranuras estandar utilizadas y el número de carretes a instalar por cada tipo de componente.
if (N_ranurasutilizadas > L);
   flag = 1;
   N_tarjetas= 0;
end;
return;


function [aux] = seleccion_nozzle(nozzles,hsn,fsn,alpha,beta,components,C,q);
%nozzles = Vector fila con los nozzles que todavía no han sido utilizados.
%hsn = Vector fila donde se encuentra el valor heurístico para la selección de nozzles.
%hsn = [ 1, J ];
%fsn = Vector fila donde se encuentra el rastro de feromonas para la selección de nozzles.
%fsn = [ 1, J ];
%alpha = Valor de importancia relativa del rastro de feromonas.
%beta = valor de importancia relativa del valor heurístico.
%components = vector fila con los índices asociados a los componentes que faltan por ser asignados.
```

```
%C = Matriz bidimensional, donde C(i,j) = 1 indica que el componente i puede ser instalado por el nozzle j, 0 de lo contrario.
%q = Proporción para la mejora de buenas soluciones previamente encontradas, frente a la búsqueda de mejores soluciones.
% Se hace una copia del vector components y se extraen sus dimensiones.
aux1 = components;
[m n] = size(aux1);
% Se extrae el número de componentes totales y el número de nozzles.
[ i j ] = size(C);
% q0 = Variable aletoario para definir si se mejoran soluciones o se buscan
%      nuevas soluciones.
q0 = rand(1);
if q0 <= q;
   j = argumax_n(nozzles,hsn,fsn,alpha,beta);
else
   j = probability_n(nozzles,hsn,fsn,alpha,beta);
end;
% Se crea un vector auxiliar con los componentes que pueden ser instalados con el nozzle j, donde su primer componente es j
aux = [];
aux = [aux; j];
for i = 1:n;
   if C(aux1(1,i),j)==1;
      aux = [aux ; aux1(1,i) ];
   end;
end;
% Se retorna un vector fila.
aux = aux';
d = find(nozzles==j);
[a b] = size(d);
if b == 0;
   q0
   bloques = input('Ingrese el número de bloques\n');
end;
return;

function j = argumax_n(nozzles,hsn,fsn,alpha,beta);
%nozzles = Vector fila con los índices asociados a los nozzles que no se han instalado.
%hsn = Vector fila donde se encuentra el valor heurístico para la selección de nozzles.
%hsn = [ 1, J ];
%fsn = Vector fila donde se encuentra el rastro de feromonas para la selección de nozzles.
%fsn = [ 1, J ];
%alpha = Valor de importancia relativa del rastro de feromonas.
%beta = valor de importancia relativa del valor heurístico.
[m n] = size(nozzles);
% arg = vector donde se calculará arg = (fsn^alpha)*(hsn^beta);
arg = zeros(n,1);
for i = 1:n;
   arg(i)= (fsn(1,nozzles(i))^alpha)*(hsn(1,nozzles(i))^beta);
end;
% Se extrae la posición asociada al mayor argumento.
[ f d ] = max(arg);
```

```
% Se actualiza el valor del indice asociado al nozzles seleccionado.
j = nozzles(d);
return;

function j = probability_n(nozzles,hsn,fsn,alpha,beta);
%nozzles = Vector fila con los índices asociados a los nozzles que no se han instalado.
%hsn = Vector fila donde se encuentra el valor heurístico para la selección de nozzles.
%hsn = [ 1, J ];
%fsn = Vector fila donde se encuentra el rastro de feromonas para la selección de nozzles.
%fsn = [ 1, J ];
%alpha = Valor de importancia relativa del rastro de feromonas.
%beta = valor de importancia relativa del valor heurístico.
[m n] = size(nozzles);
% pr = Vector donde se calculará las probabilidades.
aux = zeros(n,1);
pr = zeros(n,1);
for i = 1:n;
   aux(i,1)= (fsn(1,nozzles(i))^alpha)*(hsn(1,nozzles(i))^beta);
end;
aux = aux/sum(aux);
% Se procede a completar el vector pr.
pr(1,1) = aux(1,1);
for i = 2:n;
   pr(i,1) = aux(i,1) + pr(i-1,1);
end;
% aleatorio = valor entre 0 y 1 para seleccionar el componente.
% j = corresponde al índice del componente que fue seleccionado.
aleatorio = rand(1);
flag=0;
j = 0;
while flag==0;
   j = j+1;
   if aleatorio<= pr(j,1);
         flag=1;
   end;
end;
j = nozzles(j);
j;
return;

function [components_new]=selec_componente(components,hsc,fsc,alpha,beta,q)
   %components = vector de indices asociados a componentes sin instalarse.
   %hsc = vector de valores heuristicos para la selección de componente.
   %fsc = vector de rastro de feromonas para la selección de componente.
   %hsc = [ I 1 ] (vector columna)
   %fsc = [ I 1 ] (vector columna)
   %alpha = valor de importancia para el rastro de feromonas.
   %beta = valor de importancia para el valor heuristico.
   %q = probabilidad de mejora de buenas soluciones encontradas.

   [m, n] = size(components);
```

```
   % q0 = Variable aleatoria para determinar si se busca nuevas soluciones o se mejoran las buenas previamente
encontradas.
   q0 = rand(1);
   aux = [];
   if q0 <= q;
      % Mejorar buenas soluciones encontradas previamente.
      aux = argumax_c(components,hsc,fsc,alpha,beta,n);
   else;
      % Búsqueda de nuevas soluciones
      aux = probabilityc(components,hsc,fsc,alpha,beta,n);
   end;
   % k = corresponde a la posición en el vector "components" del índice que fue seleccionado a ser instalado.
   k = aux(1,1);
   % i = corresponde al índice del componente a ser instalado.
   i = components(k);
   % Se retira del vector "components" el indice que fue seleccionado.
   components1 = [components(1,1:k-1) components(1,k+1:n)];
   % Se organiza el vector respuesta.
   components_new = [i components1];
return

function [components_new] = argumax_c(components,h,f,alpha,beta,n);
% components = vector de índices asociados a componente a ser instalados.
% h = valor heuristico.
% f = rastro de feromonas.
% alpha = importancia del rastro de feromonas.
% beta = importancia del valor heurístico.
% n = dimensión del vector "components".
arg = zeros(n,1);
% Se completa el vector arg con los valores = (f(i)^alpha)*(h(i)^beta).
for i = 1:n;
   arg(i,1) = ((f(components(1,i),1))^alpha)*((h(components(1,i),1))^beta);
end;
% Se encuentra la posición del mayor de todos los argumentos.
% i = valor máximo de los argumentos.
% j = posición en la cual se encuentra ese valor máximo.
[i j] = max(arg);
% Se prepara el vector respuesta.
components_new = [j components];
return;

function [components_new] = probabilityc(components,h,f,alpha,beta,n);
% components = vector de índices asociados a componente a ser instalados.
% h = valor heuristico.
% f = rastro de feromonas.
% alpha = importancia del rastro de feromonas.
% beta = importancia del valor heurístico.
% n = dimensión del vector "components".
% Se quiere generar un vector p con las probabilidades de seleccionar un componente en particular.
Básicamente es generar un CDF.
p = zeros(n,1);
```

```
for i = 1:n;
      d = (f(components(1,i),1))^alpha;
      e = (h(components(1,i),1))^beta;
      p(i,1) = d*e;
end;
sump = sum(p);
for i = 1:n;
      p(i,1) = p(i,1)/sump;
end;
for i = 2:n;
      p(i,1) = p(i-1,1)+p(i,1);
end;
% aleatorio = valor entre 0 y 1 para seleccionar el componente.
aleatorio = rand(1);
flag=0;
j = 0;
while flag==0;
   j = j+1;
   if aleatorio<= p(j,1);
         flag=1;
   end;
end;
% j = corresponde al índice del componente que fue seleccionado.
components_new = [j components];
return;

function [components] = retirar_colocados(components,components_nozzles);
aux = components;
[a b] = size(components);
[d e] = size(components_nozzles);
for i = 1:e;
   [a b] = size(aux);
   d = components_nozzles(1,i);
   pos = find(aux==d);
   aux1 = aux;
   aux = [ aux1(1,1:pos-1) aux(1,pos+1:b) ];
end;
components=aux;
return;

function [orden_new]=selec_orden(i,hsr,fsr,orden,alpha,beta,q)
   %i = Indice asociado al componente seleccionado a instalarse.
   %hsr = vector de valores heuristicos para la selección de orden.
   %fsr = vector de rastro de feromonas para la selección de orden.
   %orden = vector con los índices asociados a ordenes disponibles.
   %hsr = [ 1 L ] (vector fila)
   %fsr = [ I, L ]
   %alpha = valor de importancia para el rastro de feromonas.
   %beta = valor de importancia para el valor heuristico.
   %q = probabilidad de mejora de buenas soluciones encontradas.
```

```
   % q0 = variable aleatoria que ayuda a determinar si se hace mejora de buenas soluciones o la búsqueda de nuevas soluciones
   [ m n ] = size(orden);
   q0 = rand(1);
   aux = [];
   if q0 <= q;
      % Mejora de buenas soluciones encontradas previamente.
      aux = argumax_r1(i,orden,hsr,fsr,alpha,beta);
   else;
      % Búsqueda de nuevas soluciones.
      aux = probabilityr1(i,orden,hsr,fsr,alpha,beta);
   end;
   % k = Posición del índice asociado al orden.
   k = aux(1,1);
   % l = Orden que se le ha asignado a un carrete del componente tipo i.
   l = orden(k);
   % Se actualiza el vector de índices asociados al orden.
   orden1 = [orden(1,1:k-1) orden(1,k+1:n)];
   % Se prepara el vector de respuesta, donde la primera componente es el orden asignado, y lo siguiente es el vector ordenes no asignados aún.
   orden_new = [l orden1];
return

function [orden_new] = argumax_r1(i,orden,hsr,fsr,alpha,beta);
   %i = Indice asociado al componente seleccionado a instalarse.
   %hsr = vector de valores heuristicos para la selección de orden.
   %fsr = vector de rastro de feromonas para la selección de orden.
   %hsr = [ 1 L ] (vector fila)
   %fsr = [ I, L ]
   %alpha = valor de importancia para el rastro de feromonas.
   %beta = valor de importancia para el valor heuristico.
[m n] = size(orden);
arg = zeros(n,1);
% arg = vector de la forma arg(i)= (f(i)^alpha)*(h(i)^beta).
for j = 1:n;
   arg(j,1) = ((fsr(i,orden(1,j)))^alpha)*((hsr(1,orden(1,j)))^beta);
end;
% i = Mayor valor dentro del vector arg.
% j = Posición del mayor valor dentro del vector arg.
[i j] = max(arg);
% Se prepara el vector respuesta
orden_new = [j orden];
return;

function [orden_new] = probabilityr1(i,orden,hsr,fsr,alpha,beta);
% i = Indice del componente a ser instalado.
% h = valor heuristico.
% f = rastro de feromonas.
% alpha = importancia del rastro de feromonas.
% beta = importancia del valor heurístico.
% n = dimensión del vector "components".
```

```
% Se quiere generar un vector p con las probabilidades de seleccionar un componente en particular. Básicamente es generar un CDF.
[ m n ] = size(orden);
p = zeros(n,1);
for j = 1:n;
      d = ((fsr(i,orden(1,j)))^alpha);
      e = ((hsr(1,orden(1,j)))^beta);
      p(j,1) = d*e;
end;
sump = sum(p);
for j = 1:n;
      p(j,1) = p(j,1)/sump;
end;
for j = 2:n;
      p(j,1) = p(j-1,1)+p(j,1);
end;
% aleatorio = valor entre 0 y 1 para seleccionar el orden.
aleatorio = rand(1);
flag=0;
j = 0;
while flag==0;
   j = j+1;
   if aleatorio<= p(j,1);
         flag=1;
   end;
end;
% j = corresponde a la posición del índice del orden que fue seleccionado.
orden_new = [j orden];
return;
```

```
function [asig_ant x y] = mejorarelorden(asig_ant,r,co,ant,x,y,L);
% asig_ant = Vector con la asignacion de ordenes a cada componente establecidos por la hormiga en cuestions. La estructura de este vector es de la forma [r co ant i l u w].
% 5 = Columna en la cual se encuentra el orden asignado.
% x = Vector donde se asigna cuantas unidades se extraen desde cada orden.
% y = Vector donde se señala si el orden l o no al componente i.
% r = Corresponde a la réplica que se esta ejecutando.
% co = Corresponde a la colonia que se esta generando.
% ant = Corresponde a la hormiga que se acaba de seleccionar.
% L = Corresponde al orden máximo que podría ser asignado.
% Lo primero que se hace es organizar el vector asig_ant en forma ascendente con respecto al orden.
asig_ant = sortrows(asig_ant,5);
[ m n ] = size(asig_ant);
% Se realiza un ciclo para tener ordenes consecutivos en cada lado de la máquina
orden = 1;
ranura1 = 0;
for i = 1:m;
   if( asig_ant(i,5)<=(L/2));
      ranura1 = asig_ant(i,7)+ranura1;
      asig_ant(i,5) = orden;
      asig_ant(i,7) = ranura1;
      orden = orden+1;
```

```
   end;
end;
orden = (L/2)+1;
ranura2 = 0;
for i = 1:m;
   if( asig_ant(i,5)>(L/2));
      ranura2 = asig_ant(i,7)+ranura2;
      asig_ant(i,5) = orden;
      orden = orden+1;
      asig_ant(i,7) = ranura2;
   end;
end;
% Se realiza un ciclo para llenar las matrices "x" y "y".
for c = 1:m;
   i = asig_ant(c,4);
   l = asig_ant(c,5);
   u = asig_ant(c,6);
   x(r,co,ant,i,l)= u;
   y(r,co,ant,i,l)= 1;
end;
return;
```

function [aux1, flag] = calculardistancia(asig_ant,L);

% El propósito de esta función es el de calcular la distancia que recorre la hormiga en ir a recoger todos los componentes que se deben utilizar para completar un board. Adicionalmente se utilizará la señal flag para determinar si la solucion es factible o no (0 = no se sobrepasa el número ranuras estandar disponibles, 1 = se supera el número de ranutas estandar disponibles).

```
% asig_ant = [r co ant i l const ranura_mas_distante]
flag = 0;
[ m n ] = size(asig_ant);
% Se verifica cual es la ranura mas lejana que se ha asignado.
max_ranuras = max(asig_ant(:,7));
% En caso que la ranura mas lejana sobrepasa el límite se señala flag=1.
if max_ranuras > (L/2);
   flag = 1;
end;
% Se calcula el total de distancia recorrida.
aux1 = 0;
for i = 1:m;
   aux1 = (asig_ant(i,6))*(asig_ant(i,7))+aux1;
end;
return;
```

function[mejores5hormigas] = encontrar_mejores5hormigas(val_dis_co);

% Retorna las mejores 5 soluciones factibles que se han generado en la colonia. Retorna una columnna con el número de la hormiga, el valor de la función objetivo y un indicador de factibilidad. han generado en un colonia. Es decir, dice cuales son las hormigas y los correspondientes valores de la función objetivo.

```
%val_dis_co = la columna 3 corresponde a la hormiga, la 4 al valor de la
%función objetivo y la 5 a si es factible o no.
mejores5hormigas = [];
% aux = Matrix donde se guardaran las columnas 3 a 5 de las soluciones
```

```
%       factibles.
aux = [];
[m n] = size(val_dis_co);
% Completar la matriz aux con las hormigas factibles solamente.
for i = 1:m;
   if val_dis_co(i,5)==0;
      aux = [aux; val_dis_co(i,3:n)];
   end;
end;
% Se ordenan ascendentemente las hormigas de acuerdo al valor de la función objetivo.
aux = sortrows(aux,2);
% Se extrean las primeras 5 filas solamente.
mejores5hormigas = aux(1:5,:);
return;

function delta_fsr = update_fsr(r,co,hor,mejores5hormigas,y,maxfsr);
[f g] = size(mejores5hormigas);
e = size(y);
delta_fsr=zeros(e(4),e(5));
mindis = min(mejores5hormigas(:,2));
for h= 1:f;
   for i = 1:e(4);
      for l = 1:e(5);
         if y(r,co,mejores5hormigas(h,1),i,l) == 1;
            delta_fsr(i,l) = mindis/mejores5hormigas(h,2);
         end;
      end;
   end;
end;
maxdelta_fsr = max(max(delta_fsr));
delta_fsr= delta_fsr*(maxfsr)/maxdelta_fsr;
return;

function delta_fsn = update_fsn(r,co,hor,mejores5hormigas,t,maxfsn);
v = [];
v = size(t);
% J = Número total de nozzles.
J = v(1,4);
% delta_fsn = Vector en el cual se guardara el delta de rastro de feromonas.
delta_fsn = zeros(1,J);
% minNozzle_time = Menor tiempo recorrido en recorrer en la ubicación de nozzles.
minNozzle_time = min(mejores5hormigas(:,5));
for i = 1:5;
   for j = 1:J;
      aux = t(r,co,mejores5hormigas(i,1),j,1)+ t(r,co,mejores5hormigas(i,1),j,2);
      if aux >= 1;
         delta_fsn(1,j) = (minNozzle_time/mejores5hormigas(i,2))+ delta_fsn(1,j);
      end;
   end;
end;
maxdelta_fsn = max(max(delta_fsn));
```

```
delta_fsr= delta_fsn*(maxfsn)/maxdelta_fsn;
return
```

Apéndice B

Apéndice B.1 Formulación Lingo 8.0 Caso 1 – Modelo sin "Nozzles"

```
MODEL:
SETS:
        COMP /1..4/: V, B, TAM, A;
        RAN /1..14/: D, D1;
        RAN1 /1..7/;
        RAN2 /8 9 10 11 12 13 14/;
        MIX (COMP, RAN): X, Y;
ENDSETS
DATA:
        V = 3000 3000 5000 3000;
        B = 12 18 54 24;
        TAM = 3 1 1       2;
        N = 200;
        Nt = 15;
        W = 7;
        Z = 5;
ENDDATA

! El conjunto de restricciones;
        MIN = @SUM( MIX( I, L): X( I, L) * D1( L));

! Se calcula los vectores de distancias;
        @FOR( RAN(L): @SUM( COMP (I): TAM(I)*Y(I,L))-D(L)=0);
        D1(1) - D(1) = 0;
        D1(2) - D1(1) - D(2) = 0;
        D1(3) - D1(2) - D(3) = 0;
        D1(4) - D1(3) - D(4) = 0;
        D1(5) - D1(4) - D(5) = 0;
        D1(6) - D1(5) - D(6) = 0;
        D1(7) - D1(6) - D(7) = 0;
        D1(8) - D(8) = 0;
        D1(9) - D1(8) - D(9) = 0;
        D1(10) - D1(9) - D(10) = 0;
        D1(11) - D1(10) - D(11) = 0;
        D1(12) - D1(11) - D(12) = 0;
        D1(13) - D1(12) - D(13) = 0;
        D1(14) - D1(13) - D(14) = 0;
! Se debe cumplir con la reestriccion del número de tarjetas a procesar con un setup;
        @FOR( MIX( I, L): N*X( I, L) - V(I) <= 0);

! Una ranura no puede ser asignada a mas de un tipo de componente;
        @FOR( RAN( L): @SUM( COMP( I): Y( I, L)) <= 1);

! No se pueden asignar mas ranuras de las disponibles en el lado 1;
        @SUM( COMP(I): @SUM( RAN1(L): TAM(I)*Y(I,L))) <= W;
```

```
! No se pueden asignar mas ranuras de las disponibles en el lado 2;
        @SUM( COMP(I): @SUM( RAN2(L): TAM(I)*Y(I,L))) <= W;

! El número total de carretes de tipo i;
        @FOR( COMP(I): @SUM( RAN(L): Y(I,L)) = A(I));

! Se debe instalar el total de componentes de tipo i;
        @FOR( COMP(I): @SUM( RAN(L): X(I,L)) = B(I));

! El número de componentes que se instalan del mismo componente desde diferentes ranuras debe ser igual;
        @FOR( MIX( I, L) : X(I,L)*A(I) <= B(I));

! Si se coloca un componente se debe activar la variables para ese valor;
        @FOR( MIX( I, L) : X(I,L) <= V(I)*Y(I,L));

! Si se asigna un carrete hay que colocar componentes desde ese sitio;
        @FOR( MIX( I, L) : Y(I,L) <= X(I,L));

! El valor que toma y debe ser uno o cero;
        @FOR( MIX( I, L) : @BIN(Y(I,L)) );

! El valor que toma x debe ser entero;
        @FOR( MIX( I, L) : @GIN(X(I,L)) );
END
```

Apéndice B.2 Formulación Lingo 8.0 Caso 2 – Modelo con "Nozzles"

```
MODEL:
SETS:
        COMP /1..4/: V, B, TAM, A;
        RAN /1..14/: D, D1;
        MIX (COMP, RAN): X, Y;
        RAN1 / 1..7 /;
        RAN2 / 8 9 10 11 12 13 14/;
        NOZ /1..5/;
        CN (COMP, NOZ): C;
        LADOS /1 2/;
        SN (NOZ, LADOS): T;
ENDSETS
DATA:
        V = 3000 3000 5000 3000;
        B = 12 18 54 24;
        TAM = 3 1 1      2;
        N = 200;
        Nt = 15;
        W = 7;
        Z = 5;
        C =     1 0 1 1 0
                0 0 1     1 0
                1 0 0 0 0
                0 1 1 1 1;
```

```
ENDDATA
! El conjunto de restricciones;
        MIN = @SUM( MIX( I, L): X( I, L) * D1( L)) + @SUM( SN( J, K): T(J,K))*Nt ;
! Se calcula los vectores de distancias;
        @FOR( RAN(L): @SUM( COMP (I): TAM(I)*Y(I,L))-D(L)=0);
        D1(1) - D(1) = 0;
        D1(2) - D1(1) - D(2) = 0;
        D1(3) - D1(2) - D(3) = 0;
        D1(4) - D1(3) - D(4) = 0;
        D1(5) - D1(4) - D(5) = 0;
        D1(6) - D1(5) - D(6) = 0;
        D1(7) - D1(6) - D(7) = 0;
        D1(8) - D(8) = 0;
        D1(9) - D1(8) - D(9) = 0;
        D1(10) - D1(9) - D(10) = 0;
        D1(11) - D1(10) - D(11) = 0;
        D1(12) - D1(11) - D(12) = 0;
        D1(13) - D1(12) - D(13) = 0;
        D1(14) - D1(13) - D(14) = 0;

! Se debe cumplir con la reestriccion del número de tarjetas a procesar con un setup;
        @FOR( MIX( I, L): N*X( I, L) - V(I) <= 0);

! Una ranura no puede ser asignada a mas de un tipo de componente;
        @FOR( RAN( L): @SUM( COMP( I): Y( I, L)) <= 1);

! No se pueden asignar mas ranuras de las disponibles en el lado 1;
        @SUM( COMP(I): @SUM( RAN1(L): TAM(I)*Y(I,L))) <= W;

! No se pueden asignar mas ranuras de las disponibles en el lado 2;
        @SUM( COMP(I): @SUM( RAN2(L): TAM(I)*Y(I,L))) <= W;

! El número total de carretes de tipo i;
        @FOR( COMP(I): @SUM( RAN(L): Y(I,L)) = A(I));

! Se debe instalar el total de componentes de tipo i;
        @FOR( COMP(I): @SUM( RAN(L): X(I,L)) = B(I));

! Si se coloca un componente se debe activar la variables para ese valor;
        @FOR( MIX( I, L) : X(I,L) <= V(I)*Y(I,L));

! Si se asigna un carrete hay que colocar componentes desde ese sitio;
        @FOR( MIX( I, L) : Y(I,L) <= X(I,L));

        @FOR( MIX( I, L) : X(I,L)*A(I) <= B(I));

! El valor que toma y debe ser uno o cero;
        @FOR( MIX( I, L) : @BIN(Y(I,L)) );

! El valor que toma x debe ser entero;
        @FOR( MIX( I, L) : @GIN(X(I,L)) );
```

```
! El valor que toma T debe ser uno o cero;
        @FOR( NOZ( J) : @BIN(T(J,1)) );
        @FOR( NOZ( J) : @BIN(T(J,2)) );

! La suma de Nozzles en cada lado no debe sobrepasar la capacidad;
        @SUM( NOZ(J): T(J, 1)) <= Z;
        @SUM( NOZ(J): T(J, 2)) <= Z;

! Si se activa un carrete, entonces debe existir un nozzle que pueda colocarlo;
        @FOR( COMP (I): @SUM( RAN1( L): Y( I,L)) <= B(I)*@SUM( NOZ(J): C(I,J)*T(J, 1)));
        @FOR( COMP (I): @SUM( RAN2( L): Y( I,L)) <= B(I)*@SUM( NOZ(J): C(I,J)*T(J, 2)));

END
```