INSTITUTO TECNOLOGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

CAMPUS MONTERREY

DIVISION DE INGENIERIA Y ARQUITECTURA

PROGRAMA DE GRADUADOS EN INGENIERIA

Tec de Monterrey

CAMPUS MONTERREY

"ALGORITMO INTELIGENTE PARA SECUENCIACION DINAMICA DE MOVIMIENTOS EN UN HSP"

TESIS

PRESENTADA COMO REQUISITO PARCIAL PARA OBTENER EL GRADO ACADEMICO DE:

MAESTRO EN CIENCIAS EN AUTOMATIZACION CON ESPECIALIDAD EN INGENIERIA DE CONTROL

POR:

ENRIQUE MARTINEZ ORTIZ

MONTERREY, N. L. DICIEMBRE DE 2002

INSTITUTO TECNOLOGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

CAMPUS MONTERREY

DIVISION DE INGENIERIA Y ARQUITECTURA
PROGRAMA DE GRADUADOS EN INGENIERIA

"ALGORITMO INTELIGENTE PARA SECUENCIACION DINAMICA DE MOVIMIENTOS EN UN HSP"

TESIS

PRESENTADA COMO REQUISITO PARCIAL PARA OBTENER EL GRADO ACADEMICO DE:

MAESTRO EN CIENCIAS EN AUTOMATIZACION
CON ESPECIALIDAD EN INGENIERIA DE CONTROL

POR:

ENRIQUE MARTINEZ ORTIZ

MONTERREY, N. L. DICIEMBRE 2002

Dedicatoria

Con mucho cariño y amor dedico tesis a mi familia.

A mis padres: Froylan y Rosario por todo el amor, esfuerzo y dedicación que han tenido para guiarme y darme la oportunidad de llegar hasta donde hoy he llegado logrando una meta más en mi vida. Gracias mamá por el coraje y amor que me has dado. Gracias papá por tu apoyo incondicional y tus sabios consejos.

A mi esposa y mi hija: Lucy y Ale, porque ustedes son toda la motivación que me mueve para lograr lo que me propongo. Gracias Lucy por estar siempre en las buenas y en las malas apoyándome incondicionalmente, gracias por todo el amor que me tienes. Y gracias a ti Ale por ser la luz de nuestras vidas.

A mis hermanos: Dany y Froy, por todo el respeto y cariño que me tienen y porque se que siempre contare con ellos, y ellos conmigo, los quiero mucho y estoy feliz y orgulloso de ser su hermano.

A todos ustedes dedico esta tesis porque en los mejores y peores momentos de mi vida siempre han estado conmigo. Los quiero mucho y son parte muy importante de mí. Mi familia. Gracias.

Agradecimientos

Al mi asesor el Dr. Jorge Limón Robles, por su ayuda incondicional y por su apoyo, entusiasmo y dedicación para la realización de este trabajo.

Al Dr. Hugo Terashima Marín, por sus valiosas aportaciones, y recomendaciones a lo largo de todo este trabajo.

Al Ing. Gerardo Vallejo por su cooperación y apoyo para la realización de este trabajo, y en especial por su amistad y por los consejos que me dio durante mi estancia en el CSIM. Gracias Gera.

A todos mis amigos del CSIM, Gera, Paco, Alfredo, Moni, Jorge, Amparito, Juanito, Octavio y Rodolfo. A todos ustedes gracias por su amistad sincera, por todos los consejos que me dieron, y por todos los momentos que compartimos.

A mis amigos de la maestría, Octavio Rangel, Luis Rosas, Francisco Calleja, Francisco Montes, Raúl Ceceña, Miguel Medina, Teodoro Delgado y Fernando Díaz. A todos gracias por su amistad y por todos los momentos buenos y difíciles que compartimos.

Resumen.

En esta investigación se estudia un *Hoist Scheduling Problem* (Problema de Secuenciación de los movimientos de una Grúa en una línea de Electrodeposición). Este tipo de problema usualmente se presenta en procesos de recubrimiento. Se propone un método para encontrar secuencias para los movimientos de las grúas, que tiendan a maximizar el *throughput* (taza de producción promedio, partes/periodo de tiempo). El método propuesto es un algoritmo inteligente basado en árboles de búsqueda para solucionar el problema de secuenciación de grúas.

Nuestra hipótesis es que un algoritmo dinámico de búsqueda en árboles como el propuesto tiene un mejor desempeño ante condiciones de demanda cambiante que los algoritmos cíclicos no dinámicos, y que los heurísticos de toma de decisiones para procesos manuales.

Los resultados computacionales demuestran que el método propuesto genera secuencias con throughput igual o mejor que los obtenidos con otros métodos publicados en la literatura. La solución propuesta puede llevarse a la implementación a diferentes niveles. Estos niveles dependerán del nivel de automatización deseado ya que el sistema puede utilizarse para generar secuencias para aplicación manual o integrarse en el sistema de control para generar dinámicamente la mejor secuencia y ejecutarla automáticamente.

ÍNDICE

Lista de Figuras.

Lista de Tablas.

1. INTRODUCCION.

1.1. Contexto de la investigación.

Electrodeposición.

Descripción.
Electrodeposición es el proceso de recubrir objetos con una capa metálica. Tales capas realizan una función principalmente protectora, para prevenir la corrosión del metal. Normalmente se utiliza el galvanizado que consiste en un recubrimiento de zinc. Otra aplicación es la función decorativa, baños de oro o plata. O pueden ser ambas funciones como es el cromado, que protege de la corrosión y al mismo tiempo da una mejor vista de las partes recubiertas.

El proceso de electrodeposición se lleva a cabo de la siguiente manera: en un tanque o celda se deposita una sustancia que puede ser de dos tipos, ácida o alcalina. En esta sustancia se sumergen dos piezas, la primera es la que será recubierta, esta es llamada "pieza de trabajo"; la segunda pieza es el metal que se va a electrodepositar en la pieza de trabajo. Las dos piezas se conectan a una fuente de voltaje CD. La pieza de trabajo es el cátodo el cual se conecta al polo negativo. Y el metal a electrodepositar es el ánodo y se conecta al polo positivo.

Ya conectadas las piezas se hace circular una corriente del cátodo hacia el ánodo (del metal a electrodepositar hacia la pieza a recubrir) a través de la solución del tanque. Al circular esta corriente los iones metálicos del ánodo tienen una carga positiva que es atraída por el cátodo o pieza de trabajo. Y de esta forma el ánodo se empieza a carcomer y el cátodo se empieza a recubrir por el metal que pierde el ánodo. Figura 1.1.

Antes de aplicar la electrodeposición hay una serie de pasos anteriores, la primera parte del proceso es esencial y se trata de la limpieza y preparación del material, estos procesos son críticos porque de ellos depende en gran parte la calidad del revestimiento.

A continuación se presentan dos ejemplos para mostrar los pasos a seguir en dos proceso diferentes de electrodeposición. Cromado y galvanizado.

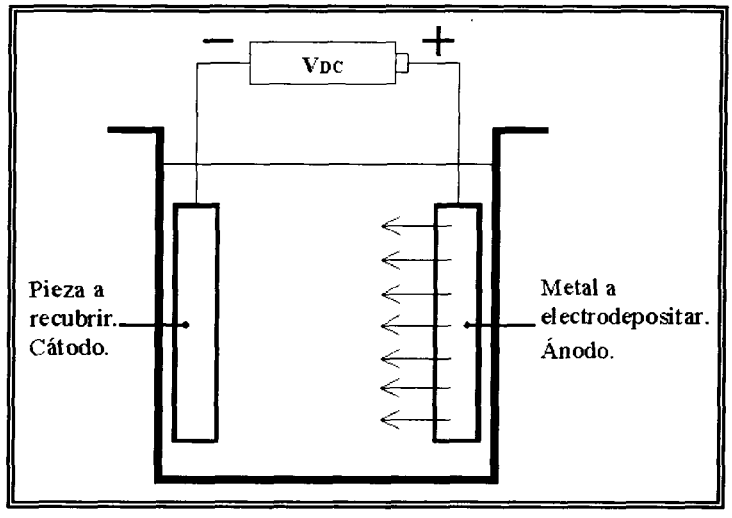


Figura 1.1. Proceso de electrodeposición.

Proceso de Cromado:

- Despintado. Las piezas son sumergidas en una solución de sosa al 30%, a una temperatura de 50-60ºC.
- Desoxidado. Las piezas son sumergidas en una solución de ácido sulfúrico al 20% a temperatura ambiente.
- Desengrase. Las piezas se sumergen en un desengrase por inmersión en baños alcalinos. Después se enjuagan y se dejan escurrir.
- Desengrase. Las piezas se sumergen en un desengrase electrolítico cianurado. Después se enjuagan las piezas en agua sin agitación.
- Activación. Las piezas se activan en una solución ácida para después pasar por un enjuague sin agitación.
- Recubrimiento. Cuando las piezas son de ZAMAC (aleación de zinc, aluminio, magnesio y cobre), pasan por un proceso de cobrizado, siguiendo posteriormente con los procesos de niquelado y cromado. Las piezas de fierro se someten directamente a los procesos de niquelado y cromado, después de los cuales se enjuagan en agua sin agitación.

Proceso de Galvanizado Alcalino.

- Predesengrase. Las piezas se sumergen en un desengrase por inmersión.
- Desengrase. Se sumergen en un desengrase electrolítico cianurado. Después se enjuagan las piezas en agua sin agitación.
- Decapado. Las piezas se activan en una solución ácida para después pasar por un enjuague sin agitación.

- Galvanizado. Las piezas se sumergen en una solución de galvanizado con cianuro y posteriormente se enjuagan en agua sin agitación.
- Sellado. Por ultimo, las piezas pasan por un proceso de sellado y su enjuague respectivo.

Los ejemplos descritos anteriormente pueden variar dependiendo del material a recubrir y la técnica a utilizar.

Dependiendo de las características de las piezas a recubrir estas se cuelgan en soportes adecuados para ello (*racks*) o se introducen en barriles (contenedores). Estos dos elementos transportan las piezas a lo largo de todo el proceso.

Tiempos de producción.

A continuación se presentan los rangos de tiempo de proceso de las etapas de electrodeposición. Estos tiempos se definen dependiendo del tamaño y cantidad de piezas a recubrir, así como también del espesor de la capa y las condiciones en que se encuentre la pieza antes del proceso.

- Desengrase por inmersión. De 3 a 30 minutos.
- Desengrase electrolítico. De 3 a 30 minutos.
- Enjuague. De 1 a 2 minutos.
- Activado. De 1 a 15 minutos.
- Cobrizado. De 5 a 60 minutos.
- Niquelado. De 1 a 30 minutos.
- Recuperador de níquel. De 2 a 4 minutos.
- Latonado. De 2 a 60 minutos.
- Cromado. De 30seg. a 15 minutos.
- Recuperado de cromo. De 2 a 4 minutos.
- Sellado. De 1 a 4 minutos.

Estos tiempos tienen un porcentaje de tolerancia permisible que se establece dependiendo del proceso que se este realizando y de la pieza que se este trabajando.

Etapas de producción.

Las **etapas comunes** para todos los procesos de electrodeposición son:

- Desengrase por inmersión.
- Desengrase electrolítico.
- Enjuague.
- Activado.
- Sellado.

Las **etapas principales** del proceso, es decir, en las que se electrodeposita metal en las piezas de trabajo son las siguientes:

- Cobrizado.
- Niquelado y recuperado de níquel.
- Latonado.
- Cromado y recuperado de cromo.

Estos procesos de electrodeposición son los más comunes en la industria. Una línea de electrodeposición de un sólo proceso de recubrimiento constaría de una de las etapas principales y de varias etapas comunes.

Características físicas.

Usualmente todas las etapas del proceso están identificadas como tanques, es decir, cada etapa del proceso se lleva a cabo en un tanque de baño químico. De esta forma una línea de electrodeposición se compone de varios tanques normalmente acomodados en línea recta, con la entrada en un extremo y la salida en el otro extremo. En el *layout* de estos procesos hay muchas variantes, pueden existir líneas circulares, o con la entrada y salida en un mismo extremo, etc.

El **acomodo de los tanques** puede variar dependiendo de los pasos del proceso y del *layout*. Por ejemplo, pueden estar acomodados uno tras de otro en el mismo orden que sigue el proceso, o pueden estar intercalados, etc.

El número de tanques por cada etapa también puede variar, no necesariamente tiene que ser uno por etapa. Normalmente las etapas más tardadas son en las que se lleva a cabo el recubrimiento; estas etapas por ser de mucha más duración que las demás, usualmente se componen de más de un tanque.

En los procesos en línea con alto nivel de producción, la **forma en que se transporta el material** de un tanque a otro es por medio de grúas viajeras. Las grúas se mueven sostenidas por un riel, el movimiento de las grúas es a lo largo de ese riel en línea recta, y suben y bajan material en los tanques. El número de grúas aumenta dependiendo del tamaño de la línea de producción, puede ser de una en adelante. Estas grúas comúnmente no tienen problemas de movimientos pendulares ya que eso se evita gracias al diseño de la sujeción de la grúa.

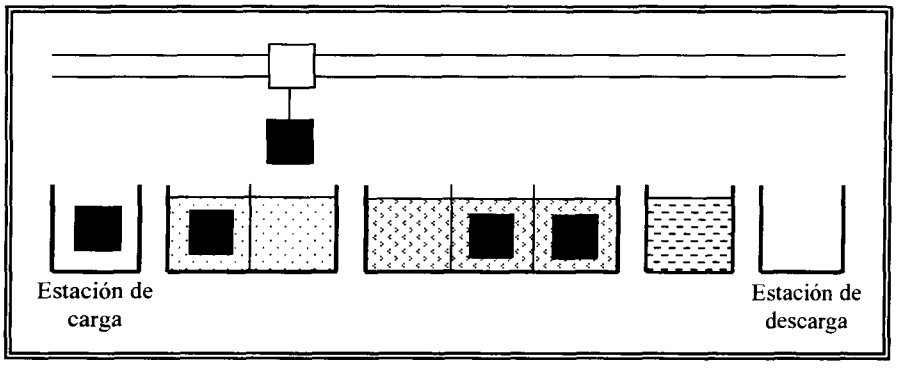


Figura 1.2. Ejemplo de línea de electrodeposición.

Tipos de pieza.

Las líneas de electrodeposición pueden ser flexibles, ya que se pueden dedicar a procesar un solo tipo de pieza o pueden procesar diferentes tipos de piezas. Cada tipo de pieza tiene una secuencia propia, de tiempos y movimientos, es decir no todos los tipos de piezas son procesadas de la misma forma ya que varían en forma, tamaño y en calidad de acabado.

Secuencias.

Una línea de producción puede estar preparada para realizar uno o varios tratamientos diferentes. Un factor importante para esto es el volumen de producción ya que pueden existir líneas dedicadas a un solo proceso y a un producto, así como también pueden existir líneas lo suficientemente flexibles como para trabajar diferentes procesos y diferentes tipos de producto.

La forma más común de hacer secuencias es desarrollando heurísticos basados en la experiencia de los dueños del proceso, ellos con su experiencia deciden en que momento meter material a la línea y como realizar los movimientos dentro de esta.

Debido la complejidad de esta actividad se han investigado y desarrollado diferentes algoritmos para realizar las secuencias. Una vez que se desarrollan las secuencias, los movimientos pueden ser manuales o se pueden programar en un PLC que se encargue de comandar las grúas. El sistema de producción automático más usual, es el que se implementa después de encontrar una secuencia cíclica que arroje buenos resultados, la cual se programa en un PLC para que se ejecute en forma repetida. Este tipo de solución es conocido como HSP (*Hoist Scheduling Problem*) cíclico. Normalmente está diseñado para un sólo tipo de pieza y no permite cambios.

La parte fundamental de este proceso es el desarrollo de secuencias, ya que una buena secuencia puede ahorrar mucho tiempo y recursos. Desgraciadamente la complejidad de esta tarea es muy alta.

1.2. Situación problemática.

En los procesos de electrodeposición se deben generar secuencias para los tiempos y movimientos del material dentro de la línea que maximicen un criterio predefinido típicamente el *throughput*. En este tipo de sistemas de producción es difícil generar secuencias ya que son complicados por sus características físicas y de proceso. La primer dificultad que se presenta son las restricciones del sistema.

Restricciones.

- Una vez que una pieza termina su procesamiento en un tanque tiene que ir tan rápido como sea posible al siguiente tanque. Esto significa que no hay *buffers* en el sistema, es decir, todo inventario en proceso esta en producción o en traslado.
- El tiempo de baño, para completar el proceso, no es exacto. Pero para asegurar la calidad del baño, y de todo el tratamiento, los químicos proporcionan un tiempo mínimo y un máximo para cada proceso. Así que este tiempo de tolerancia es la única fuente de flexibilidad en la línea.
- La grúa puede transportar solo un contenedor o *rack* a la vez.
- Los tanques pueden procesar solo un *rack* o contenedor a la vez. Pueden existir tanques múltiples, es decir, varios tanques que realicen el mismo proceso.
- Cuando hay más de una pieza en la línea, la grúa debe de tener el tiempo suficiente para realizar los movimientos de todas las piezas a su debido tiempo.

Uno de los problemas que se presentan en estas instalaciones consiste en secuenciar las operaciones de baño y de movimientos de la grúa para llegar a una meta definida (por ejemplo maximizar el *throughput*), reducir el *WIP* (*Work in Process*), reducir los costos de producción, etc. mientras se respeta todas las restricciones del sistema y del medio ambiente de producción. Este problema es conocido en la literatura como *Hoist Scheduling Problem*.

Las restricciones específicas implican que una vez que algún producto entra en la línea, se tiene que asegurar que este podrá salir. Por consecuencia los movimientos de la grúa tienen que ser bien planeados.

Este problema es específico, es diferente de los problemas clásicos de secuenciación. Existen diferentes tipos de instalaciones para los procesos de electrodeposición, de acuerdo al número y las características de los tratamientos y a los recursos de transporte (capacidad). Consecuentemente, una gran diversidad de sistemas físicos pueden ser

estudiados. Además, el ambiente de producción con el que se lidia depende del número y tipo de especificaciones de proceso (secuencias, tiempos de proceso) a ser considerados.

1.3. Relevancia del problema.

Muchos procesos industriales emplean grúas controladas por computadoras para el manejo de materiales. Las grúas están programadas para ejecutar una secuencia fija de movimientos repetitivos. Cada repetición de la secuencia de movimientos es llamada ciclo, y el tiempo total requerido por la grúa para completar el ciclo es llamado tiempo de ciclo. La importancia de minimizar el tiempo de ciclo es evidente por el hecho de que los tamaños de los lotes son usualmente muy grandes y una corrida de producción puede requerir semanas. El problema fundamental es no contar con una secuencia adecuada ya que esto significa aumentar los tiempos de ciclo con lo que la utilización de las instalaciones será baja ocasionando altos costos de producción. Por otra parte, la variabilidad del mercado implica la necesidad de desarrollar nuevas secuencias para los nuevos productos. En México en particular según reportan las empresas entrevistadas es común hacer cambios a la secuencia de tiempos en función de la calidad de la materia prima. Si se usara en estos casos el esquema de secuencias fijas se deben proveer tiempos de proceso holgados que garanticen la calidad. Este enfoque no permite aprovechar las ventajas de una materia prima de buena calidad. Las secuencias desarrolladas por los métodos convencionales llevan mucho tiempo, y no aseguran maximizar la utilización y/o evitar producir desperdicio si se cambian las características del proceso, esto aumenta demasiado los costos de cambios de proceso.

1.4. Justificación de la investigación.

Dadas estas circunstancias es muy importante, el poder contar con un mecanismo de generación flexible de secuencias eficientes que aseguren maximizar la utilización de las instalaciones. El tiempo de generación de la nueva secuencia debe ser razonablemente corto en comparación con el tiempo de ejecución.

La comunidad de investigadores interesados en el problema, constantemente están proponiendo nuevos métodos para realizar secuencias cada vez más eficientes, desgraciadamente estas soluciones no están al alcance de la mayoría de empresas dedicadas a este ramo, o simplemente todavía no están en la etapa de poderse implementar. Al investigar la situación del HSP en la industria local se notó la falta de desarrollo de soluciones a nivel automatización, por lo que es muy conveniente desarrollar investigaciones que permitan llevar a cabo soluciones fáciles de implementar. Por otra parte la industria local está interesada en participar directamente en el desarrollo de nuevas tecnologías que les permitan mejorar sus procesos y hacerlos más flexibles.

1.5. Objetivos de la investigación.

El objetivo general de la investigación es:

Desarrollar un algoritmo inteligente para la generación dinámica, en tiempo real, de secuencias para el HSP sin intervención del ser humano partiendo de los datos de proceso.

Aunque para esta tesis nos restringiremos a generar secuencias para líneas de un sólo producto y de una sola grúa, se buscara un enfoque que pueda ser extendido en investigaciones posteriores a líneas multiproducto y con más de una grúa.

1.6. Alcances.

Se desarrollará la metodología y se implantará mediante un algoritmo computacional. El algoritmo computacional deberá tener las siguientes características:

- Permitir especificar por el usuario:
    - El layout del proceso.
    - La secuencia de proceso de acuerdo a los datos reales (etapas y tiempos) recabados en entrevistas con usuarios de este tipo de procesos.
    - Las velocidades de la grúa y las distancias de los tanques.
- Tendrá la capacidad para:
    - Generar la secuencia de movimientos y simularla automáticamente.
    - Simular secuencias en modo manual.

La aplicación a un proceso real en la cual el programa envíe la secuencia optima a un PLC para que este la ejecute en tiempo real esta fuera del alcance de esta tesis.

La metodología se evaluara vía simulación comprobándola contra el modelo *benchmark* de Phillips y Unger [10] (12 tanques de capacidad unitaria y una sola grúa) que es utilizado en la mayoría de los artículos relacionados con el tema. Además de probarlo con otros procesos reales y propuestos con tanques de capacidad múltiple.

1.7. Aportaciones de esta investigación.

En este apartado se presentara un resumen de las principales aportaciones de esta investigación. Las aportaciones son a nivel metodología y a nivel *software*.

Aportes a nivel metodología son las siguientes:

1. En reglas de ramificación:

   - Especialmente la forma de introducir piezas nuevas al sistema con retrasos de tiempo.
   - Las reglas de selección de opciones para mover material dentro de la línea, las opciones son mover la pieza con tiempo máximo mas critico y mover las piezas con tiempo infinito de tolerancia.

2. En reglas de evaluación de nodos y control de búsqueda:

   - Forma de evaluar y seleccionar nodos para expandir. Específicamente el algoritmo que selecciona el mejor nodo y después realiza *backtracking* para elegir a uno de sus antecesores asegurando así un mejor desempeño.

3. En reglas de poda:

   - Reglas para identificar si un nodo generara desperdicio (*scrap*) en su descendencia y evitar su ramificación.
   - Las reglas de asegurar lugar en el siguiente tanque ocupado y en la estación cuello de botella.

Aportes a nivel *software* son las siguientes:

- Un *software* de simulación de procesos de electrodeposición en modo manual y automático, con una interfase grafica que permite comprender mejor los procesos y sus restricciones.
- A este software se integra el algoritmo de búsqueda que genera las secuencias basándose en todas las reglas desarrolladas. De tal forma que a parte de simular secuencias se pueden generar automáticamente.

1.8. Descripción del documento.

En este apartado se describe brevemente el contenido de la tesis.

En el capitulo 2 se describen los datos más importantes del HSP, del proceso de electrodeposición, técnicas usadas por otros autores para resolverlo, así como investigaciones realizadas y enfoques propuestos por distintos autores.

En el capitulo 3 se muestran los datos reales investigados, se define el problema de HSP a resolver en particular, la metodología propuesta, las técnicas utilizadas y la forma en que se aplicaron.

En el capitulo 4 se explica la forma en que se implemento la metodología en un *software* computacional, mostrando el sistema resultante y su forma de uso.

Por ultimo en el capitulo 5 se muestran los resultados obtenidos comparándolos contra otros métodos propuestos en la literatura y con procesos reales de la industria.

2. ANTECEDENTES

2.1 Marco teórico.

2.1.1 Definiciones importantes del HSP.

Baño o etapa.
Se le llama baño o etapa al proceso en el cual una pieza es sumergida en un tanque de proceso químico. Cuando un baño puede ser ejecutado en uno de varios tanques, se dice que es un baño o etapa multitanque.

Racks y Contenedores.
En la industria de la electrodeposición hay dos formas muy comunes de transportar las piezas que va a ser procesadas. Una de ellas es mediante estructuras metálicas conocidas como racks en las que se cuelgan las piezas. En cada *rack* pueden ser colgadas muchas piezas, la otra forma es mediante contenedores dentro de los que se depositan las piezas. En la literatura relacionada al tema se puede referir a estos elementos como: material, trabajo, pieza o como sus propios nombres, contenedor o *rack*. Ver figuras 2.1 y 2.2.

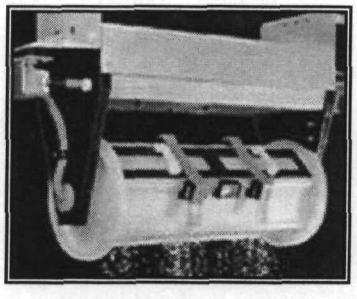


Figura 2.1. Contenedor.

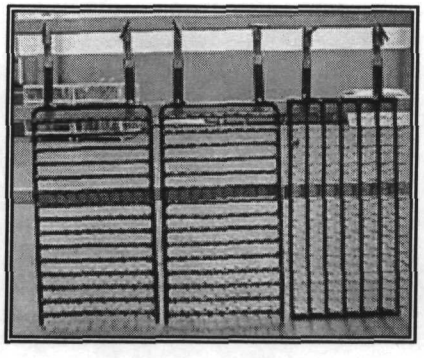


Figura 2.2. *Rack*.

Corrida.
Una corrida es la cantidad total de piezas iguales a procesar. Usualmente la cantidad de piezas a producir en una corrida es enorme, se llega a tardar la producción varias semanas.

Línea Multi-producto.
En el HSP el término *multi-producto* se refiere a que la línea de electrodeposición va a manejar más de una pieza con procesos diferentes. Cuando la entrada de estas piezas es aleatoria, es decir, cuando no se sabe de qué tipo será la siguiente en entrar, la secuenciación se debe realizar en **tiempo real**, ya que a cada entrada de una pieza nueva el algoritmo debe calcular la secuencia para esa pieza y para las piezas remanentes dentro de

la línea. Este problema también es conocido como HSP dinámico (DHSP *Dynamic Hoist Scheduling Problem*).

Línea Multi-grúa.

Cuando en una línea de electrodeposición las piezas son transportadas por más de una grúa se dice que este tipo de sistemas es multi-grúa. Es importante mencionar que cuando se maneja este tipo de HSP la complejidad del problema aumenta exponencialmente.

Makespan.

Lapso de tiempo total en que son ejecutados todos los trabajos.

Ciclo simple.

Se le llama ciclo simple a la secuencia repetitiva realizada en una línea de electrodeposición. En estos ciclos normalmente el número de piezas en proceso es limitado y fijo. Entra una nueva pieza a la línea después de que otra la abandona, y así se inicia un nuevo ciclo. Este tipo de procesos también se conoce como HSP cíclico.

Conflictos.

También conocidos como inconsistencias. Un conflicto se puede llamar a aquel nodo que encuentra la búsqueda en el cual las restricciones no han sido cumplidas y no puede seguir siendo expandido. Por ejemplo, un conflicto ocurre cuando dos operaciones que usan el mismo recurso (grúa o tanques) tienen que ejecutarse durante el mismo periodo de tiempo.

Throughput.

Es la taza de producción promedio (partes/periodo de tiempo) de la línea o de un sistema.

WIP.

Work in process. La traducción es "Trabajo en proceso" y se refiere al promedio de piezas que están en proceso dentro de la línea.

Benchmark de Phillips y Unger [10].

En la comunidad de investigadores del HSP existe un problema *benchmark* que la mayoría usa para probar sus algoritmos propuestos. Este sistema *benchmark* fue propuesto por Phillips y Unger en 1976, consta de 13 etapas de un solo tanque cada una y de una sola grúa, con piezas idénticas. El *benchmark* es conocido como el *benchmark* de Phillips y Unger.

2.1.2 Principales técnicas utilizadas en el HSP.

El HSP es considerado un Problema de Satisfacción de Restricciones. Un problema de satisfacción de restricciones puede ser formalmente definido por:

- Un grupo de variables.

- Cada una de las cuales tiene un grupo de posibles valores discretos y finitos (su dominio).
- Y un grupo de restricciones entre esas variables.

La solucion a un CSP es encontrar un valor para cada variable, de su respectivo dominio, el cual satisfaga todas las restricciones.

El HSP es estratégico para la industria, por lo que deben existir muchos trabajos sin publicar. Ya que en muchas empresas las secuencias son generadas usando heurísticos desarrollados empíricamente. Pero por otra parte, hay una gran variedad de técnicas científicas utilizadas para solucionar el HSP. Antes de mencionar las principales técnicas y herramientas que han sido usadas para resolver el HSP, conviene mencionar que el HSP es un problema NP-completo [4][6][10].

Se les llama así a problemas en los cuales el tiempo para obtener la solución optima por los métodos actualmente existentes crece en forma no-polinomial (exponencial) con la complejidad del problema [23].

Clase NP-completos.- Se conoce una amplia variedad de problemas de tipo NP, de los cuales destacan algunos de ellos de extrema complejidad. Gráficamente podemos decir que algunos problemas se hayan en la "frontera externa" de la clase NP. Son problemas NP, y son los peores problemas posibles de clase NP. Estos problemas se caracterizan por ser todos "iguales" en el sentido de que si se descubriera una solución P para alguno de ellos, esta solución sería fácilmente aplicable a todos ellos. Y por tanto, la clase NP desaparecería del mundo científico al carecerse de problemas de ese tipo. Realmente, tras años de búsqueda exhaustiva de dicha solución, es un hecho ampliamente aceptado que no debe existir, aunque nadie ha demostrado, todavía, un algoritmo P para resolver problemas NP. Por lo tanto cuando se demuestra que un problema es NP Completo, esta justificada la búsqueda de heurísticos para obtener soluciones suboptimas en tiempos razonables.

A continuación se presentan algunas de las principales técnicas y herramientas que han sido usadas para resolver el HSP.

a) *Branch and Bound* [18].

La traducción de este término es Ramificación y poda, es un método de búsqueda para solución de problemas.
Es un algoritmo de búsqueda con el cual se encuentra la solución optima, esto se realiza manteniendo la mejor solución encontrada, si una solución parcial es encontrada y no es mejor que la que se tiene se siguen explorando las demás opciones, hasta que se encuentre una mejor o se terminen las opciones. Este sistema de búsqueda es conocido también como árbol de búsqueda y existen muchas estrategias de búsqueda para podar las ramificaciones y hacer más eficientes las búsquedas.

Pensando en las rutas que ocupan un área geográfica se suele simular el espacio de problema donde se halla la ruta entre un nodo de inicio y un nodo meta, a la manera de un árbol repetidamente ramificado. Estrictamente, ya que el nodo inicial se suele ubicar arriba, estaría mencionándose una estructura de raíces de árbol. Ver figura 2.3.
En la figura 2.3 se puede observar el esquema de un árbol de búsqueda, el nodo marcado con la letra A es el nodo inicial, por ejemplo este nodo tiene 3 posibles opciones, que son los nodos B, C, y D. Hay muchas estrategias de búsqueda, en las que se incluye heurísticos para recortar el área de búsqueda. La forma más sencilla de realizar la búsqueda es checar todos los nodos de uno por una hasta encontrar el nodo meta. Si no se conoce cual es el nodo meta entonces se realiza la búsqueda de la misma forma pero se evalúa cada uno de los nodos expandidos y el que haya dado mejores resultados ese será el meta.

Como se puede apreciar este método es poco eficiente cuando el área de búsqueda es grande, normalmente en el HSP representado en árboles de búsqueda tiene un factor de ramificación exponencial así que sería muy difícil buscar por todos los nodos la solución. Aquí es donde entran las estrategias de búsqueda y los heurísticos, ya que estos nos pueden ayudar a recortar enormemente el área de búsqueda.

Dentro de la técnica de búsqueda *Branch and Bound* se puede distinguir las siguientes estrategias:

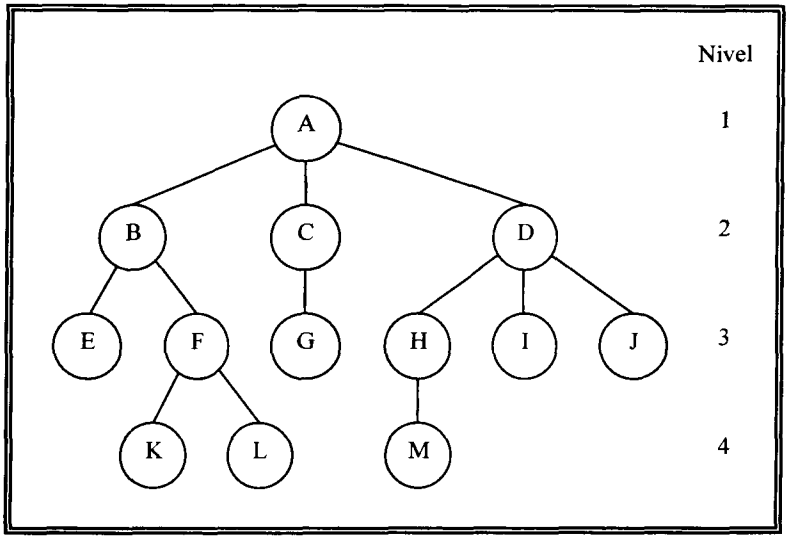


Figura 2.3. Esquema de un árbol de búsqueda.

1. Búsqueda primero en profundidad [18].

Mejor conocida como DFS por sus siglas en inglés (*Depth First Search*). Se trata de una búsqueda desinformada donde el nodo más profundo no terminal es el que primero se expande. Aquí la lista de espera de nodos por procesar crece por el tope. Los "sucesores",

que son los nodos recién expandidos (siempre que no sean terminales, que no se podrían expandir) son anotados en el tope de la lista de espera (*last in first out* - LIFO). El inconveniente de este método es que se puede ciclar infinitamente para lo cual se corrige el método original por uno modificado que verifica si el nodo bajo análisis se halla también en una "lista cerrada" donde aparecen los nodos que ya fueron procesados una vez. Si es así, se descarta sin expandir. Fig. 2.4.

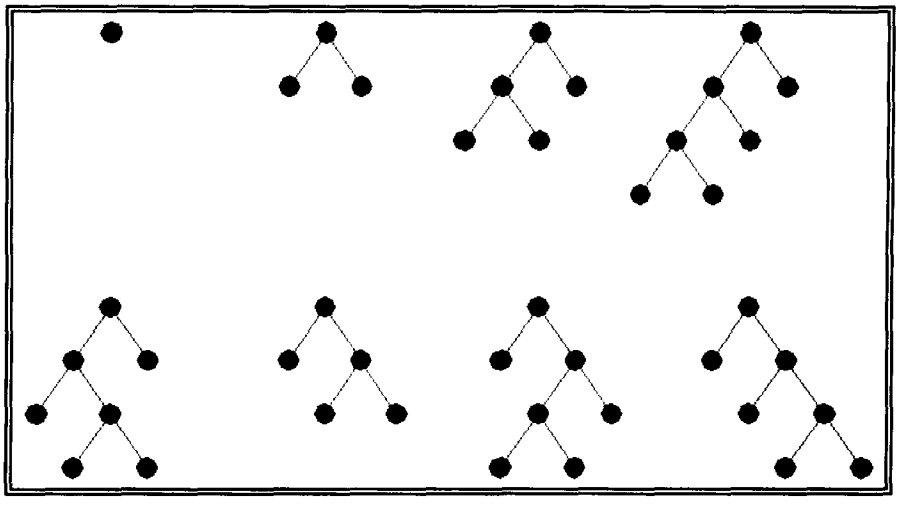


Figura 2.4. Búsqueda DFS para un árbol binario.
Se asume que los nodos en el nivel 4 no tienen sucesores.

2. Búsqueda Primero en Anchura [18].

Mejor conocida como BFS por sus siglas en inglés (*Breadth First Search*). En esta estrategia de búsqueda primero se ramifica el nodo inicial que se encuentra en un nivel cero de nodos, después de ramificado ese nodo, sus hijos forman entre todos el siguiente nivel de búsqueda que es el nivel 1. Después son ramificados todos los nodos del nivel 1 generando así el nivel 2, y así sucesivamente prosigue la ramificación. No se puede ramificar ningún nodo de un nivel determinado hasta que no hayan sido ramificados todos los nodos del nivel anterior.

Aquí la lista de espera de nodos por procesar crece por la cola. Los "sucesores", que son los nodos recién expandidos (siempre que no sean terminales, que no se podrían expandir) son anotados en el fin de la lista de espera (*first in first out* - FIFO). Fig. 2.5.

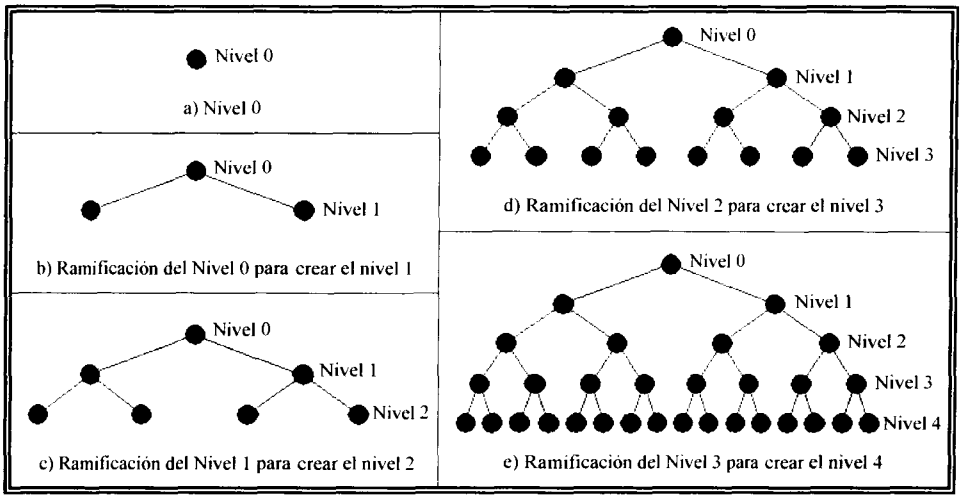


Figura 2.5. Búsqueda BFS para un árbol binario.
Se asume que los nodos en el nivel 4 no tienen sucesores.

3. Evaluación de nodos.

Pueden existir diferentes formas de evaluar un nodo. Cuando el nodo meta es conocido, solo se hace una comparación a cada ramificación de la búsqueda hasta encontrarlo. Cuando no se sabe cual es el nodo meta, entonces se realiza una evaluación más complicada, dependiendo las características de la búsqueda es el criterio de evaluación. Se dice que un nodo es bueno cuando su evaluación es buena o si es el meta. Y se dice que un nodo es malo cuando este produce conflictos o al evaluarlo resulta mal calificado.

4. Backtracking [18].

Para entender el *backtracking* suponga un árbol de búsqueda en el que se esta buscando un nodo meta especifico. La forma en que se solucionaría el problema usando *backtracking* seria la siguiente.
El algoritmo busca por cada nodo, siguiendo un orden especifico, hasta que:

1. El algoritmo encuentra una solución o
2. Se descubre que el nodo que se expandió no puede ser expandido más o no va a llevar a una solución.

Si ocurre el paso numero 1 el algoritmo ha encontrado el nodo meta y se termina la ejecución. Pero si ocurre el paso número 2 entonces el algoritmo regresara por los nodos anteriormente expandidos hasta encontrar un nodo en el cual pueda seguir buscando por una solución.

5. Backtracking dinámico (dynamic backtracking) [18].

Este algoritmo es una adaptación que hizo Matthew Ginsberg del algoritmo original de *backtracking*.
Al resolver un problema de búsqueda, es importante recordar que posibles soluciones han sido consideradas y cuales no lo han sido. Haciéndolo, se reducirá drásticamente el número de posibilidades que no son necesarias considerar en el futuro. Esta técnica es generalmente conocida como *backtracking* dinámico, hasta recientemente, se aplicaba el *backtracking* sin guardar información de las inconsistencias y estas se repetían constantemente en cada punto de la búsqueda.
Estas técnicas han sido viables en la práctica por la introducción de *backtracking* dinámico o "aprendizaje de la relevancia de los límites". La idea básica es almacenar solo la información acerca de las posibles soluciones que parezcan que probablemente serán consideradas después durante la búsqueda en general y no borrarlas junto con los nodos malos; si una pieza de información parece ser que no será útil después, esta es eliminada junto con el nodo malo. Esta técnica es de mucha utilidad en un amplio rango de casos prácticos.

b) Herramientas de programación matemática.

Esta enfocada a resolver problemas de programación lineal. También conocidos como LPP's por sus siglas en inglés (*Lineal Programming Problems*) [24]. Un problema de programación lineal tiene por finalidad optimizar (maximizar o minimizar) una función lineal.

Cuando un modelo incluye restricciones en forma de enteros, se dice que es un problema de programación de enteros. Cuando algunas, pero no todas las variables de decisión son restringidas a enteros, se dice que es un problema de programación mixta MIP (*Mixed Integer Programming*). Resolver estos problemas puede requerir mucho as tiempo de cálculos que los mismos problemas sin las restricciones en forma de enteros.

c) Programación Lógica de Restricciones

Este tipo de programación es conocido en la literatura como CLP por sus siglas en inglés *Constraint Logic Programming*. Es uno de los primeros lenguajes que manejan restricciones.

Usualmente este tipo de programación se realiza con softwares especiales que permiten modelar y desarrollar algoritmos para ligar variables a restricciones de tal forma que se puedan procesar y satisfacer.

La programación lógica de restricciones emerge del área conocida como Inteligencia Artificial. Una de las herramientas usadas por este tipo de programación es la **propagación de restricciones**.

El conocer y explotar las restricciones del problema, permiten reducir la búsqueda haciendo selecciones "inteligentes". Pueden detectar inconsistencias y remediarlas. El algoritmo de propagación de restricciones funciona de la siguiente manera. Cuando un valor es asignado a una variable dada, ya sea por el usuario o por el sistema, el algoritmo recalcula los posibles grupos de valores y asigna valores a todas sus variables dependientes. Este proceso continúa recursivamente hasta que no haya más cambios en la búsqueda. Más específicamente, cuando una variable X cambia su valor, el sistema evalúa el dominio de la expresión para cada variable Y dependiente de X. Esto podría generar un nuevo grupo de posibles valores para Y. Si este grupo cambia, las restricciones son evaluadas para seleccionar una de los posibles valores como el valor nuevo asignado para Y. Si el valor asignado es diferente del previo, causa que el sistema recalcule los valores en forma descendiente en las variables de restricciones. Los valores que han sido asignados por el usuario tienen preferencia siempre y cuando no causen inconsistencias.

A continuación se presentan los lenguajes de programación más usuales par resolver este tipo de problemas.

PROLOG

Prolog, cuyo nombre se deriva de *PROgramming in LOGic*, es el lenguaje de mayor impacto en el paradigma de la programación lógica. La base de Prolog son las nociones matemáticas de relaciones y de inferencia lógica. Prolog es un lenguaje declarativo, lo que significa que en vez de describir cómo computar una solución, un programa consiste en una base de datos con hechos y relaciones lógicas que describe las relaciones relacionadas con una aplicación dada. Por lo que en vez de correr el programa para obtener una solución, el usuario hace una pregunta a dicha base. Una vez que se genera una pregunta, el sistema busca a través de la base de datos para determinar (por deducción lógica) la respuesta.

Algunas veces puede que haya más de una manera para deducir la respuesta o que haya más de una respuesta, en esos casos se puede solicitar al sistema por más soluciones, por lo que se genera un proceso de "*backtracking*" para encontrar soluciones alternas.

No existe una estructura definida para crear un programa en Prolog. No hay un procedimiento principal y no existen las anidaciones de definiciones. Todas las reglas y hechos son globales y una variable se resuelve por el hecho o regla donde esta aparezca y la o lo haga válida.

Prolog es usado en aplicaciones de inteligencia artificial tales como interfaces de lenguaje natural, sistemas de razonamiento automático y sistemas expertos. Consisten generalmente en una base de datos con hechos y reglas junto con una interfaz que accesa dicha base.

CHIP

Este lenguaje es llamado CHIP por sus siglas en inglés *Constraints Handling in Prolog* (Manejo de Restricciones en Prolog).

Las restricciones son pieza central del medioambiente de aplicación CHIP, pero otros componentes importantes juegan un rol importante en el desarrollo de aplicaciones.

El sistema CHIP consiste de un numero diferente de componentes los cuales juntos simplifican enormemente el desarrollo de aplicaciones. La herramienta extiende la funcionabilidad de un lenguaje base con restricciones y otros subsistemas.

2.2 Investigaciones anteriores.

El HSP ha sido enfrentado por varios investigadores, las primeras investigaciones se iniciaron en la década de los 70's del siglo pasado. A lo largo de todo este tiempo las técnicas y resultados han ido mejorando. Muchas de las soluciones propuestas están muy limitadas a los casos específicos de estudio por lo que es difícil aplicar estas soluciones a los diferentes problemas reales que existen en la industria. A continuación se describen los trabajos más sobresalientes en el HSP.

1992. Baptiste P., Legeard B., Varnier C. [9]

Un algoritmo aproximado fue propuesto en 1976, y los algoritmos óptimos aparecieron en 1988. En este estudio los autores proponen un nuevo modelo para solucionar el HSP, el modelo esta orientado a **Programación Lógica de Restricciones** *CLP*, desarrollado en PROLOG III. Este sistema produce soluciones óptimas con un simple programa. En los casos en que fue probado el modelo, es tan efectivo como los métodos anteriores propuestos por otros autores.

El objetivo de los autores es encontrar una secuencia óptima de movimientos de la grúa. La solución encontrada debe ser aquella que tenga el menor tiempo de ciclo y debe respetar estrictamente todos los tiempos de proceso.

Los autores subrayan que muchos trabajos previos son herramientas muy útiles de investigación, han abierto caminos presentando modelos interesantes, y encontrado soluciones optimas bajo un esquema académico. Su principal desventaja es que están restringidos al problema inicial de su estudio.

Los lenguajes de CLP se extienden más allá del lenguaje Prolog, añadiéndole características de resolución de muchas formas distintas como pueden ser restricciones lineales en forma de números racionales, enteros, variables booleanas o árboles infinitos. Muchas implementaciones han sido propuestas en esos términos. Los *softwares* principales que se encuentran en el mercado son: PROLOG III para aritmética lineal racional, términos booleanos y árboles infinitos; y CHIP (*Constraints Handling in Prolog*) para dominios finitos, términos booleanos y términos lineales racionales.

Para solucionar el HSP que los autores proponen eligieron el lenguaje de programación PROLOG III, por la necesidad de manejar números racionales en las restricciones. Así que su investigación esta basada en CLP hecha en PROLOG III.

Los autores expresan que su enfoque propuesto es capaz de encontrar secuencias para ciclos simples de secuenciación para HSP con una sola grúa. La extensión a multi-grúas, afirman los autores, que es obvia y no modifica la complejidad del problema ni tampoco el tiempo de solución.

Las restricciones que manejan en su investigación son las siguientes:

- Todas las operaciones de baño tienen un tiempo mínimo y máximo.
- Solo hay una grúa que realiza todos los movimientos.
- Los tiempos de movimiento de la grúa entre los tanques son conocidos y constantes.
- Los tanques están acomodados de acuerdo al proceso de electrodeposición, la numeración va de C_0 hasta C_N. Siendo C_0 el tanque de entrada y C_N el tanque de salida.
- Dos contenedores nunca pueden ocupar el mismo tanque al mismo tiempo.
- Hay tiempo suficiente entre los movimientos de la grúa para regresar desde donde se encuentre después de realizar el último viaje hasta el sitio del siguiente trabajo.

El software de la investigación lo desarrollaron y probaron en una maquina SUN 4-SLC8 Mega RAM, con PROLOG III. Su software fue probado vía simulación e indican que su solución fue comparada con el modelo *benchmark* de Phillips y Unger **[10]** y los resultados obtenidos fueron los mismo obtenidos por Shapiro y Nuttle **[8]** en la investigación que realizaron en 1988.

1994. Lamothe J., Correge M., Delmas J. [10]

En su investigación consideraron una producción multi-producto en un contexto de tiempo real. La principal dificultad es secuenciar los movimientos de las grúas con respecto a los tiempos de baño y con la restricción de que no hay inventario en proceso. Así que la única de forma de resolver el problema en tiempo real consiste en calcular una nueva secuenciación de los movimientos de la grúa, cada vez que una nueva pieza llega a la línea. Esta nueva secuencia debe asegurar la producción de la nueva pieza y de las que ya están dentro de la línea. Este tipo de problema es conocido como HSP dinámico (DHSP *Dynamic Hoist Scheduling Problem*). En su investigación consideran este tipo de problema con la restricción de que una sola grúa debe mover todo el material.

En su trabajo, los autores, calculan un pronóstico de secuencia basándose en un modelo analítico, y un método de resolución optimo; ese enfoque parase ser el de un HSP cíclico. Pero, como el DHSP debe ser resuelto en tiempo real, reglas y heurísticos deben ser introducidos en la solución y su eficiencia debe ser analizada.

Para comprobar esa estrategia, simularon una línea con un horizonte determinado, con datos del *benchmark* de Phillips y Unger para medir la taza producción global y los tiempos de cálculo computacional.

También asumieron que una sola grúa debe realizar todos los movimientos, los tiempos de viaje de tanque a tanque son datos dados y que un tanque no puede atender más de una pieza a la vez. Su objetivo general fue maximizar el *throughput* global de la línea.

Adoptaron el método de árboles de búsqueda utilizando la estrategia de primero profundidad. En cada nodo de la búsqueda, un grupo de restricciones son dadas y la solución óptima debe satisfacerlas. Para determinar esta solución para ese nodo, buscan conflictos: un conflicto consiste de dos operaciones que usan el mismo recurso (grúa o tanques) durante el mismo periodo de tiempo. Si no se detectaron conflictos, una solución viable de DHSP es encontrada. De otra manera el grupo de todos los conflictos es establecido y un heurístico de ramificación selecciona uno del grupo. Entonces de esa forma se definen las nuevas ramificaciones y/o restricciones a añadir a la búsqueda.
En los nodos de la búsqueda, en el heurístico de ramificación que lleva a elegir una separación o conflicto, se determina la velocidad del algoritmo teniendo en cuenta la eficiencia de este heurístico. Un buen heurístico debe llevar rápidamente cerca de una solución óptima de tal forma que el límite de ramificación sea más efectivo. Compararon diferentes heurísticos de ramificación en términos de velocidad de solución.

El algoritmo que desarrollaron fue probado vía simulación con el sistema de Phillips y Unger, la simulación reveló la habilidad de repitibilidad con respecto a los límites de tiempo de cálculo. También indican que demostraron la eficiencia en la optimización del *throughput* en comparación con los heurísticos simples. El problema es que el algoritmo requiere de largos tiempos de cálculo.

1994. Haoxun Chen, Chengbin Chu, Proth J.M. [6]

En su investigación consideraron un HSP cíclico. Proponen un modelo y un algoritmo para encontrar una secuencia cíclica óptima para los movimientos de la grúa. El modelo muestra propiedades analíticas del problema. Esas propiedades permiten eliminar soluciones no viables. Su algoritmo está basado en árboles de búsqueda. El cálculo de límites bajos de ramificación y detección de soluciones no viables requiere la solución de una clase específica de problemas de programación no lineal. Desarrollaron un algoritmo polinominal para resolver esos problemas. Muestran resultados en los que afirman que su algoritmo supera otros algoritmos publicados en la literatura.

El modelo en el cual basan su investigación consta de n tanques (o estaciones), de los cuales uno es de entrada y otro de salida. Consideran una secuenciación cíclica simple. El objetivo del HSP que investigaron es minimizar el tiempo de ciclo, lo cual es equivalente a maximizar el *throughput*.

Dos procedimientos de búsqueda son utilizados. El procedimiento "A" implícitamente enumera todas las posibles distribuciones iniciales de piezas en un ciclo, sabiendo que el primer movimiento en un ciclo consiste en transportar una pieza de la estación de carga al tanque 1. Algunas restricciones numéricas relacionadas a los tiempos de proceso de las piezas son usadas par para reducir el número inicial de distribuciones de piezas. El procedimiento "B" implícitamente enumera las secuencias de los movimientos de la grúa para cada distribución potencialmente factible. El LPPs (*Linear Programming Problems*) relacionado con el límite inferior de cálculos para el algoritmo de búsqueda en el proceso "A" y en el proceso "B" se transforma en evaluación de problemas de tiempo de ciclo. Estos problemas son resueltos usando un algoritmo iterativo basado en herramientas gráficas. La experiencia computacional en algunos problemas *benchmark* es presentada para comparar los resultados obtenidos usando este nuevo algoritmo con los resultados provistos por la literatura.

Calculan un límite superior del máximo número de piezas que puede atender la grúa en un ciclo. El límite superior es usado para reducir el espacio de la solución en el algoritmo del árbol de búsqueda. Su algoritmo en términos generales usa una búsqueda en primero profundidad más *backtracking* para seleccionar el próximo nodo.

Aplicaron el algoritmo a 5 problemas. Los problemas fueron resueltos en una maquina IBM 6091/19.

1995. J. Lamothe, M.Correge and J.Delmas [3]

Estos investigadores consideran que una grúa debe realizar todos los movimientos, y que todos los tanques son de capacidad unitaria. Consideran una producción multi-producto en un contexto de tiempo real. A cualquier tiempo puede llegar cualquier trabajo, dado el estado de toda la línea (la posición y el tiempo remanente de los baños en proceso de todos los trabajos que ya están dentro de la línea), cada vez que llegue un trabajo nuevo a la entrada toda la secuencia debe volver a calcularse.

Anteriormente fue propuesto un enfoque más genérico **[13]**. En este estudio, su enfoque consiste en mejorar el tiempo de computación para un DHSP (para respetar los límites reales de los tiempos de computación) mientras se preserva la eficiencia en la producción. Un sistema basado en un árbol de búsqueda es usado para optimizar su enfoque. Pero proponen remplazar el clásico algoritmo de *backtracking* por el algoritmo dinámico de *backtracking* de Ginsberg.

En su algoritmo utilizan la estrategia de búsqueda de primero profundidad. En cada nodo de la búsqueda, se da un grupo de restricciones potenciales, y una grafica es definida. El criterio de evaluación proporciona el límite de profundidad para la evaluación. Pueden existir conflictos durante la búsqueda. Si no son detectados los conflictos entonces una secuenciación viable de DHSP es encontrada. De otra forma, se ramifica el conflicto elegido buscando solucionar el problema.

Usualmente cuando el *backtracking* es necesario en cualquier nodo, el procedimiento es regresar a los nodos anteriores y borrar toda la información almacenada de los nodos a través de los cuales se ha regresado. Los autores proponen usar un algoritmo de *backtracking* más eficiente: regresa y borra información solo si es necesario.

Durante el algoritmo de búsqueda, a cualquier tiempo puede ser detectada una inconsistencia, por lo que el *backtracking* es necesario. Un nodo malo está relacionado solo con inconsistencias y errores pasados. El algoritmo clásico de *backtracking* borra los nodos malos y las restricciones de búsqueda del nodo. Pero Ginsberg modifico este algoritmo para tomar las restricciones del nodo y usarlas para evitar nodos malos.

Para probar sus resultados simularon el modelo *benchmark* de Phillips y Unger en tiempo real. Esta línea esta compuesta de 13 tanques. Los trabajos son todos idénticos. Supusieron que un nuevo trabajo llega a cualquier tiempo y otro deja el sistema.

1995. C. Cheng, S. Smith [11]

Basaron su investigación en la aplicación de un modelo de CSP (*Constraint Satisfaction Problem*) desarrollado anteriormente para un complicado problema de secuenciación multi-producto. La meta es maximizar el *throughput* de una planta que produce tarjetas de circuitos impresas mientras se asegura satisfacer las restricciones de procesos, capacidad y manejo de material. Construyéndolo desde un procedimiento general de heurísticos que se refiere a él como PCP (*Precedence Constraint Posting*), el cual se basa en una grafica de representación de restricciones del problema temporal, definieron un procedimiento extendido de la solución para minimizar el *makespan*. En una serie de experimentos comparativos, su procedimiento fue encontrado significativamente con mejores resultados que los procedimientos publicados anteriormente.

Para demostrar la viabilidad de su modelo secuenciador basado en CSP, consideraron aplicarlo a un complejo problema de secuenciación previamente estudiado, uno multi-producto. Todos los materiales son manejados por una sola grúa, la cual es capaz de transportar cada pieza de trabajo desde la entrada llevándola de tanque en tanque hasta la salida. La grúa puede transportar sólo una pieza a la vez, tiene una velocidad horizontal constante y una velocidad vertical constante. La línea no tiene capacidad de almacenar trabajos durante el proceso, así que los trabajos deben ser movidos de tanque a tanque inmediatamente después de que terminan su proceso en cada tanque. El objetivo es maximizar el *throughput*.

Para determinar el desempeño, se realizó el siguiente estudio computacional. Fue asumida una planta de electrodeposición de tarjetas de circuitos impresas con 5 tanques. Todos los experimentos generados consistieron de 100 trabajos, para cada uno se determino ruta de trabajo y restricciones de tiempo de procesamiento aleatorios, y se asumió que todos los trabajos estaban disponibles simultáneamente.

Para verificar los resultados, los problemas también fueron resueltos usando un algoritmo de HSP desarrollado por **Yhi [11]**, ambos algoritmos fueron implementados en C y ejecutados en una maquina Sun SPARC 10.

1995. Varnier C., Grunder O., Baptiste P. [7]

El *layout* de los tanques es generalmente considerado como un dato fijo. En su estudio, los autores, muestran como el *layout* de los tanques puede influenciar la productividad de la línea. Proponen un enfoque que combina la investigación de una secuenciación óptima cíclica de los movimientos de la grúa y el *layout* optimo de las estaciones.

En el diseño de líneas de electrodeposición las restricciones químicas solo son tomadas en cuenta. Claro, esas restricciones son muy importantes por los riesgos de contaminación de los baños. Sin embargo, eso deja un ancho rango de posibilidades que nunca son exploradas. Particularmente cuando la línea es dedicada a un solo producto.

En su estudio, tratan de mostrar la importancia del *layout* de las estaciones. Su enfoque esta basado en un modelo que desarrollaron previamente. Usa programación lógica de restricciones para encontrar una simple secuencia cíclica de un sólo producto. Adaptaron ese algoritmo para procesar al miso tiempo el mejor *layout* de las estaciones. Afirman que la modificación del *layout* puede apreciablemente mejorar la productividad de la línea. El objetivo principal de su investigación es mejorar el *throughput*.

El modelo matemático de su investigación es implementado con CLP.

La búsqueda de secuenciación viable consiste en usar propagación de restricciones y el procedimiento de primero profundidad los cuales son interesantes propiedades de los lenguajes de CLP.

En los árboles de búsqueda que implementan, en cada ramificación el tiempo mínimo de ciclo encontrado es el límite de aceptabilidad de tiempo de ciclo para la siguiente búsqueda, de no encontrar un tiempo de ciclo menor el algoritmo hace *backtracking* para seguir buscando por otras ramificaciones permitiendo así realizar búsquedas completas y obtener soluciones óptimas.

Ellos han relajado el problema inicial considerando las ubicaciones de las estaciones como datos variables. Y así para cada secuencia obtenida pueden desarrollar el *layout* óptimo asociado a esa secuencia.

Otra dificultad es definir las "buenas" restricciones. Las buenas restricciones son aquellas que permiten podar tan rápido como sea posible la búsqueda de la solución del problema.

En primer plano, trataron de encontrar el *layout* óptimo para secuencias dadas usando las capacidades de optimización y enumeración de CLP.

Como las variables adicionales denotan la posición de las estaciones, son definidas como variables enteras. Así, el primer enfoque para encontrar el *layout* de los tanques consiste en usar un dominio finito de restricciones con la herramienta de programación (CHIP).

Otra opción para encontrar el *layout* óptimo es adaptar un árbol de búsqueda desarrollado para líneas saturadas. De hecho, este algoritmo no considera el límite superior de los tiempos de baño. Además, esto no permite tiempos de pausa para la grúa durante el ciclo. Para líneas no saturadas, el ciclo óptimo generalmente contiene algunas pausas de tiempo debidas a los tiempos de baño.
Los autores adaptaron el procedimiento de árbol de búsqueda convencional para tomar en cuenta esas nuevas restricciones. La resolución consiste en expandir un árbol de búsqueda. En cada nodo de la búsqueda una estación es elegida para la posición actual. Los otros tanques, que todavía no han sido ubicados, son fijados en la siguiente posición. Un programa lineal fue desarrollado para encontrar el valor mínimo de la suma de los tiempos de pausa, porque una evaluación lineal de un limite bajo del periodo no es posible.

Implementaron su método en dos partes. La primera para encontrar las secuencias cíclicas usando CLP en Prolog III. La segunda parte, fue desarrollada en dos algoritmos para encontrar el *layout* asociado a la secuencia generada. Primero usaron CLP programando en CHIP, lo cual les permitió satisfacer restricciones sobre dominios finitos (enteros). Con la enumeración obtuvieron el *layout* óptimo. Después desarrollaron el algoritmo del árbol de búsqueda en C.

1996. J. Lamothe, C. Thierry and J.Delmas [5]

Estos autores presentan un enfoque de tiempo real para optimizar la producción en una línea de HSP multi-producto.
Como es considerado un contexto de tiempo real, a cualquier tiempo puede llegar un nuevo trabajo, entonces una nueva secuencia de todos los trabajos tiene que ser calculada. Este problema es llamado DHSP. Diferentes enfoques que consisten en optimizar un criterio de DHSP han sido propuestos. El objetivo de su investigación es extender esos resultados previos a un problema de varias grúas.

En el plano de tiempo real, los trabajos llegan aleatoria mente a la entrada. Después, dado el estado de la línea completa (la posición de la grúa y los tiempos remanentes de los trabajos en proceso) y dados los nuevos trabajos en la entrada, una nueva secuencia debe ser calculada para toda la línea.

El algoritmo de *backtracking* dinámico de Ginsberg ha sido adaptado y permite almacenar información de DHSP previos para agilizar los próximos; un criterio eficiente basado en heurísticos para poda de ramificaciones fue entonces presentado. Pero la investigación la presentaron con solo una grúa.

El objetivo de su nueva investigación fue presentar un modelo multi-grúas y multi-producto.

En muchas de las líneas reales, las grúas son recursos saturados. Así que, para reducir los tiempos de viajes con grúas vacías, la línea es dividida en zonas y cada grúa es asignada a una zona. Cada grúa tiene su zona segura en donde no puede entrar ninguna otra. Los conflictos ocurren cuando los productos son pasados de una zona a otra.

El enfoque de resolución propuesto anteriormente por ellos mismos, de una solo grúa con DHSP, tuvo buenos índices de producción y un pequeño tiempo de cálculo. Aquí se pretende analizar si dichos resultados se mantienen al aumentar el número de grúas.

Simularon el *benchmark* de Phillips y Unger en tiempo real. Esta línea esta compuesta de 13 tanques y todas las piezas son idénticas. Supusieron que una pieza nueva llegaba a cualquier tiempo y otra dejaba la línea. El número de piezas en la entrada permanecía constante.

Por el momento su enfoque esta siendo probado en líneas industriales.

1998. Rodosek R., Wallace M. G. [4]

Presentan un enfoque para resolver problemas de HSP basándose en la integración de CLP y programación entera mixta (MIP).
En su algoritmo híbrido la búsqueda es separada del manejo de restricciones. El manejo de restricciones es desempeñado usando propagación de restricciones y solución lineal de restricciones. En la búsqueda utilizan variables enteras y booleanas.
La experiencia computacional anterior muestra que los algoritmos híbridos, combinando CLP y MIP, resuelven clases de HSPs los cuales no pueden ser manejados por algoritmos dedicados, desarrollados previamente.

Los algoritmos híbridos para resolver HSPs combinan CLP y MIP, de tal forma que ambos comparten variables y restricciones para cooperar y encontrar una solución optima.

En su investigación dos contribuciones son presentadas. Primero, modelos para clases de HSPs pueden ser definidos independientemente del método de programación que se utilice durante la búsqueda de una solución óptima. Segundo, el algoritmo híbrido propuesto determina una secuencia con el mínimo tiempo de ciclo y provee su optimabilidad para clases de HSPs que no pueden ser manejados por las herramientas previas de CLP y MIP.

Los autores sostienen que una vez que un programa de CLP ha sido desarrollado para una clase de HSP, es relativamente fácil, comparado con en enfoque de programación matemática, adaptar el modelo para otras clases de HSPs y generar diferentes algoritmos solución.

El algoritmo de evaluación que proponen permite una integración de MIP con CLP usando un modelo único para un problema. Las restricciones lineales derivadas son tratadas también con MIP o CLP o por ambos. Su algoritmo híbrido combina ambas herramientas

de tal forma que la búsqueda es separada del manejo de restricciones. La búsqueda es aplicada utilizando variables enteras y booleanas. El manejo de restricciones es desarrollado por propagación de restricciones con CLP y la solución de restricciones lineales con MIP.

Implementaron la integración de un algoritmo de CLP con MIP usando la plataforma de programación logia de restricciones ECLiPSe y el paquete de programación matemática XPRESS-MP. Esto permite al EXPRESS ser usado para resolver problemas modelados en ECLiPSe. El control del proceso de búsqueda y la propagación de restricciones es manejado con CLP mientras que la solución de restricciones lineales con MIP. La propagación de restricciones es desarrollada por un algoritmo consistente en dominio finito. Por otra parte, la solución de restricciones lineales es desarrollada por un algoritmo simple, el cual es un componente del XPRESS.
La comunicación entre los algoritmos es soportada por el ECLiPSe.
El principal beneficio de funcionamiento del algoritmo híbrido es debido a la rápida detección de fallas por los diferentes sistemas. Cada algoritmo detecta ciertas fallas las cuales no podrían haber sido detectadas por el otro antes de que el nodo este en el árbol.

1998. Collart Dutilleul S., Denat J. P. [12]

El enfoque de su investigación es presentar la habilidad de una nueva herramienta de redes de Petri para encarar HSPs. El objetivo general perseguido es secuenciar la grúa para mejorar la taza de producción de la línea.
En su investigación, primero encuentran una secuencia para la operación. Después, con programación lineal determinan los instantes de inicio de las tareas.
Primero proponen usar una útil herramienta grafica para modelar diferentes especificaciones en una planta. La segunda parte del problema es encontrar la secuencia adecuada. La tercera parte consiste en determinar un control local, eso dos últimos pasos son resueltos usando propiedades matemáticas del modelo grafico. Este modelo es llamado: redes de petri.

La línea de electrodeposición considerada contiene lo siguiente: una entrada independiente de la salida, un grupo de tanques y una grúa. No se tiene la capacidad de almacenar partes durante el proceso. Cada tanque realiza un tratamiento específico y no puede atender más de una pieza a la vez. La duración de los procesos esta dada entre un tiempo mínimo y un máximo. La grúa mueve las piezas de un tanque a otro.
El problema es encontrar la secuencia de la grúa permitiendo una taza de producción especificada.

1999. Bloch C., Manier M.A. [8]

Los autores proponen una notación y una tipología para el HSP. Su trabajo pretende hacer más fácil la comunicación entre investigadores y/o industriales en el área de tratamientos de

superficies, permitiendo una identificación exacta de los varios casos estudiados y de la comparación de los resultados obtenidos.

1999. Kats V., Levner E, Meyzin L. [13]

Los autores proponen un algoritmo para generar secuencias cíclicas óptimas para los movimientos de una grúa en una línea de electrodeposición de tarjetas de circuitos impresas donde la transportación de las partes entre las estaciones de trabajo es desarrollada por una grúa controlada por computadora. El objetivo del problema de secuenciación es maximizar el *throughput*.

Una buena secuencia que sincroniza las actividades de las maquinas y grúas puede incrementar el *throughput*, reducir el WIP (*work in process*) y reducir los costos de producción.

Proponen un modelo de secuenciación cíclica para *n* partes diferentes como un modelo de programación matemática de *n* variables con restricciones alternativas.
Consideran para sus experimentos una línea de producción compuesta varios tanques siendo el tanque cero la entrada y el último tanque la salida, los trabajos son idénticos entre si.
Estudiaron el problema con tiempos de proceso deterministicos constantes. Debido a la naturaleza del proceso de electrodeposición, una restricción de no esperar fue impuesta, requiriendo que después de que una parte fue procesada en algún baño, esta debe ser movida inmediatamente por la grúa al siguiente baño.

La forma en que manejan multi-partes es agrupándolas en conjuntos de partes idénticas.

2001. Jegou D., Baptiste P., Lee K.H. [14]

Proponen un nuevo enfoque, basado en dos sistemas multi-agente distintos. El primer sistema es responsable de tomar una decisión acerca de a que tiempo meter la siguiente pieza a la línea, mientras que el segundo sistema está relacionado con la asignación de las operaciones de transferencia de material dentro de la línea. Estos sistemas multiagente usan dos mecanismos para lograr la cooperación entre ellos. Los conflictos restantes son resueltos a través de prioridades auto asignadas.

El propósito de su investigación es proponer un sistema que ofrezca una respuesta conveniente a los requerimientos del HSP. El enfoque de multiagente ha sido escogido, por su facilidad de mapeo en los sistemas físicos así como también por su capacidad de escalabilidad. En su investigación dos sistemas multiagente son actualmente propuestos, el primero es llamado IDDS (*Input Date Decisión System*). Desde que no hay predicción, el sistema tiene que manejar incertidumbre sobre los tiempos en los cuales los futuros eventos ocurrirán. Los tanques así como las grúas (como grupo) son considerados como agentes. El segundo sistema es llamado HAS (*Hoist Assignment System*). Este sistema hace uso

sistemático de la flexibilidad permitida por los intervalos mínimos y máximos de los procesos en los tanques. Las grúas son consideradas como agentes. Ambos IDDS y HAS basan sus decisiones en tomar las mejores opciones, y están basados en diferentes mecanismos de trabajo.

En resumen el primer problema es asignar las operaciones de transferencia a las grúas. El segundo problema es decidir el mejor tiempo en el cual un nuevo trabajo debe ser introducido a la línea.

La velocidad vertical es considerada constante y las grúas no pueden cruzarse y consideraron que los tanques tienen capacidad de 1. Tienen dos objetivos: minimizar el número de trabajos defectuosos y maximizar el *throughput*.

Hasta el momento solo HAS ha sido bien probado. Ha sido comparado con heurísticos comúnmente utilizados para la asignación de operaciones de transferencia a la grúa. Esos heurísticos comúnmente usados son:

- Primero el más cercano a la grúa. (NRF)
- Asignación media de la grúa. (ARA)
- Cambio de límite por la localización de trabajos. (BSJL)

Otras investigaciones.

Phillips y Unger [10][13][4] en 1976 introdujeron un HSP en el cual todos los trabajos eran idénticos. Su meta era encontrar una secuencia cíclica en la cual un trabajo entrara en la línea periódicamente (este problema es llamado el HSP periódico 1). Ellos desarrollaron un modelo de programación integral para determinar el mínimo tiempo de ciclo. El modelo está definido con restricciones en forma de ecuaciones, usando variables continuas y booleanas. Para realizar su investigación propusieron un sistema con una sola grúa y 13 tanques, este sistema ha sido *benchmark* para gran parte de la comunidad dedicada a la investigación del HSP.

Thesen y Lei [10] en 1986 propusieron una arquitectura para un sistema secuenciador experto.

Shapiro y Nuttle [8] [10], en 1988, habían resuelto el mismo problema *benchmark*, con un método de árboles de búsqueda. Usando programación lineal para limitar el espacio de búsqueda, el algoritmo trabaja en base a un heurístico que consiste en introducir productos sucesivamente en el sistema con el mismo tiempo de entrada entre cada uno; después enumeran los posibles ordenes para las tareas de la grúa. La solución utiliza una búsqueda en primero profundidad, y si un ciclo parcial no es factible el programa regresa a través de los nodos. Estos autores enfrentaron el problema con tanques duplicados, y afirman que su modelo tiene la capacidad de encontrar *n* ciclos de secuenciación pero requiere de mucho tiempo de ejecución computacional.

Lei y Wang [8][13], en 1989 propusieron el primer algoritmo capaz de encontrar un ciclo de secuencia para sistemas de dos grúas. Proponen un algoritmo heurístico que reduce el número de secuencias. Evalúa alternativamente particiones del problema y para cada partición resuelve dos subproblemas de grúas, aplicando su procedimiento. Cuando la solución para los dos subproblemas es encontrada, el algoritmo, con un procedimiento iterativo, busca el mínimo tiempo de ciclo. El tiempo de cálculos computacionales es muy alto y también probaron que el problema es NP-duro.

Aizenshtat y Tanayev [13] desarrollaron modelos matemáticos para problema de secuenciación de múltiples partes alrededor de 30 años atrás. Su trabajo no fue continuado ni extendido excepto por el articulo que publico Kats donde heurísticos de primero el mejor fueron implementados para encontrar secuencias cíclicas de múltiples partes usando "intervalos prohibidos" de Azenshtat.
Presentaron condiciones suficientes y necesarias en el tiempo de inicio de partes de las cuales la grúa estaba disponible para realizar todas sus operaciones en tiempo. Sin embargo, no propusieron ningún método constructivo para resolver el problema.

3. MÉTODO PROPUESTO

3.1 Identificación de restricciones industriales reales.

Uno de los principales objetivos de esta investigación es realizar un algoritmo funcional que sea de gran utilidad para la industria, que resuelva el problema del HSP considerando todas las restricciones reales que existen. La implementación computacional debe ser amigable y de fácil uso. Para poder realizar este trabajo fue necesario conocer las necesidades de la industria, las características de un proceso real y conocer los puntos de vista de personas involucradas directamente con el proceso de la electrodeposición.

Se contactó dos empresas que se dedican a realizar procesos de electrodeposición así como a vender e implementar líneas de producción dedicadas a este proceso.

A continuación se presentan los datos más relevantes que se consiguieron.

Por razones de confidencialidad serán omitidos los nombres de las empresas y de las personas contactadas en ellas, de aquí en adelante serán nombradas como empresa 1 y empresa 2.

a) Empresa 1.

La empresa proporcionó las características de una línea real de niquelado que actualmente se encuentra trabajando, los datos son los siguientes:

- Consta de 12 etapas contando la entrada y la salida.
- Los tanques están acomodados en línea siguiendo el orden del proceso.
- La entrada es el primer tanque y la salida es el último.
- Una sola grúa mueve todo el material dentro de la línea.
- Cuando la grúa baja algún contenedor a procesar puede esperarlo en esa posición hasta que termine el proceso en ese tanque.
- Después de que la grúa saca los contenedores de los tanques, tiene que esperar a que pase el tiempo de escurrimiento e inmediatamente después debe ser llevado a la siguiente etapa. De no hacer esto así, las piezas se pasivan.
- La grúa tiene dos velocidades. Al arranque y paro usa una velocidad mínima de 6m/min (aproximadamente el 15% de su recorrido), y el resto (85%) a velocidad máxima de 18m/min.
- La producción puede ser por turnos de 8hrs.
- Después de que la grúa saca las piezas de los tanques, debe esperar a que transcurra un tiempo para que las piezas se escurran.

- La etapa de niquelado consta de 2 tanques.

Los datos numéricos de la línea se muestran en la tabla 3.1.

No. etapa	Proceso	Tiempo min.	Tiempo max.	Tmpo. Escurrido	Ancho tanque
0	Entrada	1.5	∞	∞	0.92
1	Desengrasado	3	0.45	0.166	0.92
2	Desengrasado	3	0.45	0.166	1.12
3	Enjuague	0.75	0.112	0.25	1.72
4	Enjuague	0.75	0.112	0.25	1.72
5	Activado	1	0.15	0.333	0.92
6	Enjuague	0.75	0.112	0.25	1.72
7	Enjuague	0.75	0.115	0.25	1.72
8	Niquelado	30	4.5	0.333	1.12
	Niquelado	30	4.5	0.333	1.12
9	Enjuague	0.75	0.112	0.333	1.72
10	Enjuague	0.75	0.112	0.166	1.72
11	Salida	1.5	∞	∞	0.92

Tabla 3.1. Proceso de Niquelado.

b) Empresa 2.

La empresa proporciono las características de una línea real de fosfatado que actualmente se encuentra trabajando, los datos son los siguientes:

- Consta de 12 etapas contando la entrada y la salida.
- Los tanques están acomodados en línea siguiendo el orden del proceso.
- La entrada es el primer tanque y la salida es el último.
- Una sola grúa mueve todo el material dentro de la línea.
- Cuando la grúa baja algún contenedor a procesar puede esperarlo en esa posición hasta que termine el proceso en ese tanque.
- La velocidad promedio de la grúa es 10.32 m/min.
- Los tiempos de producción no están definidos, pueden variar dependiendo del estado de la materia prima y del tipo de pieza.
- Después de que la grúa saca las piezas de los tanques, en algunas etapas debe esperar a que transcurra un tiempo para que las piezas se escurran.
- La etapa de fosfatado consta de 4 tanques.
- La primera etapa de baño consta de 2 tanques.

Los datos numéricos de la línea se muestran en la tabla 3.2.

No. etapa	Proceso	Tiempo min.	Tiempo max.	Tmpo. Escurrido	Ancho tanque
0	Entrada	3	∞	∞	1.079
1	Desengrasado	12.5	1.875	0.2	1.079
	Desengrasado	12.5	1.875	0.2	1.079
2	Enjuague	0.5	0.075	0.2	1.079
3	Enjuague	0.5	0.075	0.2	1.079
4	Acido	5	0.75	0.2	1.079
5	Enjuague	0.5	0.075	0	1.079
6	Enjuague	0.5	0.075	0	1.079
7	Enjuague	0.5	0.075	0	1.079
8	Enjuague	0.5	0.075	0	1.079
9	Fosfatado	30	4.5	0	1.079
	Fosfatado	30	4.5	0	1.079
	Fosfatado	30	4.5	0	1.079
	Fosfatado	30	4.5	0	1.079
10	Enjuague	0.5	0.075	0	1.079
11	Descarga	2	∞	∞	1.079

Tabla 3.2. Proceso de Fosfatado.

Los contactos en las empresas no proporcionaron valores numéricos para el *throughput* ni una secuencia de movimientos.

Los datos y restricciones proporcionados por las empresas se respetaron en su totalidad, el único dato que se modifico fue que se considero una velocidad promedio.

3.2 Definición del problema a resolver.

a) Sistema de electrodeposición.

Se considera un sistema de electrodeposición con las siguientes características y notaciones:

- Distribución en "I" con un máximo de 40 tanques, en donde n es el número total de tanques incluyendo el tanque de carga y descarga.
- La notación de los tanques es: j_0, j_1, $j_2 \ldots . j_n$.
- La carga y descarga se realiza en estaciones independientes, considerando la carga como el tanque j_0 y la descarga como el tanque j_n.
- El número total de etapas o estaciones lo define el usuario, la única limitante es que no deben exceder el número de tanques permitido. La notación para las etapas es: e_1, e_2, $e_3 \ldots . e_m$.

- Los contenedores serán transportados por una sola grúa montada en un riel a lo largo de la línea de tanques.

- Pueden existir estaciones de capacidad múltiple definidas por c_1, c_2, c_3 ..... c_m

  donde: $\sum_{i=0}^{m} c_i = n$, $c_1 = 1$ y $c_m = 1$

- Los contenedores salen del sistema y por otro medio retornan al inicio después de alguna operación de limpieza.

- En cada tanque el tiempo de proceso debe ser mayor o igual a un valor mínimo $T\min_i$ y menor o igual a un valor máximo $T\max_i$ $(i = 1, m)$.

b) Requisitos del algoritmo.

- Se fabricara un solo tipo de pieza en cada ejecución del algoritmo.

- El algoritmo genera secuencias para turnos de 8hrs. de trabajo.

El usuario determinara:

- Velocidad horizontal. Al inicio de la ejecución del algoritmo el usuario define la velocidad horizontal (*Vh*) en m/min. a que puede viajar la grúa, este dato se mantiene constante durante toda la ejecución.

- Ancho de cada tanque. Para calcular el tiempo que tarda la grúa en viajar de un punto a otro a lo largo de la línea de electrodeposición, es necesario que consideremos el ancho de cada tanque en metros empezando desde el tanque de carga a_0, hasta el tanque de descarga a_n, sucesivamente.

- Velocidad vertical. Al igual que la velocidad horizontal, la velocidad vertical (*Vv*) también la define el usuario al inicio del algoritmo y se mantiene constante durante toda la ejecución.

- Tiempo de escurrimiento para cada estación. Después de que una pieza termina su proceso de baño la grúa debe sacarla del tanque, pero antes de iniciar el viaje hacia el siguiente proceso, es necesario que transcurra un tiempo de escurrimiento para no contaminar los demás tanques, ese tiempo de escurrimiento esta definido por Te_1 hasta Te_{n-1}. Las estaciones de entrada y de salida no tienen tiempo de escurrimiento.

- Tiempo de carga y descarga de contenedores. En las líneas de electrodeposición existe un tiempo de carga, que consiste en llenar de material el contendor o *rack* próximo a entrar a la línea, así como un tiempo de descarga que consiste en descargar el contenedor o *rack* que completo su ciclo de proceso.

c) Modos de operación del algoritmo.

El *software* debe trabajar en dos modos, manual y automático.

En modo manual el usuario podrá interactuar con la línea simulada para verificar estrategias o heurísticos sin necesidad de arriesgar recursos materiales. De tal forma que con el *software* se pueda desarrollar nuevos heurísticos y comprobar los ya existentes así como probar secuencias ya existentes y poderlas comparar con las desarrolladas por el *software*.

En modo automático el usuario solo tiene que presionar un botón para que el algoritmo genere una secuencia para la línea previamente definida, al definir esta secuencia la prioridad es producir cero piezas malas y aumentar al máximo posible el *throughput*. Una vez generada la secuencia, el usuario solo tiene que presionar otro botón para que la simulación del proceso inicie. Esta simulación automática puede tener varias aplicaciones:

- Indicarle al operador la secuencia a seguir manualmente en la línea.
- Mover la grúa desde un cuarto de control por medio de una interfase física entre el usuario y la grúa, basando las decisiones en la secuencia mostrada por el algoritmo.
- Conectar un PLC entre la grúa y la PC en la que se corra el algoritmo para que este controle de una forma totalmente automática todo el proceso.

En esta investigación el *software* solo se limita a generar las secuencias y mostrarlas vía simulación, llevar a la practica los dos puntos anteriores es relativamente fácil, solo se tiene que configurar el *software* desde el algoritmo de programación principal para que entregue las secuencias en el formato deseado, y obviamente el *hardware* necesario.

3.3 Método propuesto.

En esta investigación se presenta un enfoque de búsqueda *branch and bound* para solucionar problemas de HSP. El método de búsqueda desarrollado es una mezcla de DFS y BFS con *backtracking*.

3.3.1 Representación del HSP en un sistema de búsqueda.

Nodo.

Un nodo representa el estado de la línea de electrodeposición en un punto determinado en el tiempo. Los estados representados en un nodo pueden ser después de dejar un contendor en algún tanque (Figura 3.2), o puede ser el nodo inicial (Figura 3.1) en el que no hay material en proceso y la grúa puede estar en cualquier posición en la línea. El nodo debe contener toda la información del proceso en ese punto de tiempo, esta información es el tiempo actual, los tiempos remanentes de los contenedores en proceso (en caso de que los haya), la posición de la grúa, etc.

El algoritmo que expande los nodos guarda los nodos generados en variables matriciales, correspondiendo una matriz de dos dimensiones para cada nodo, en donde guarda todos los datos antes mencionados.

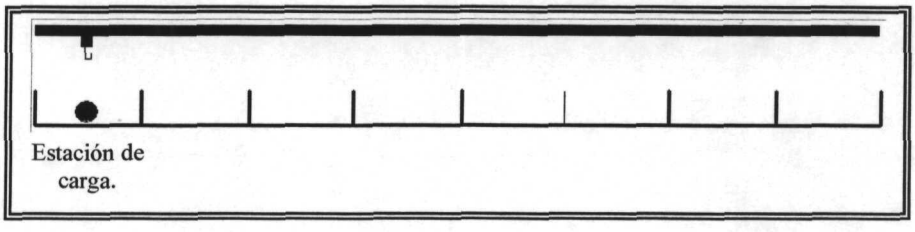


Figura 3.1. Nodo inicial.

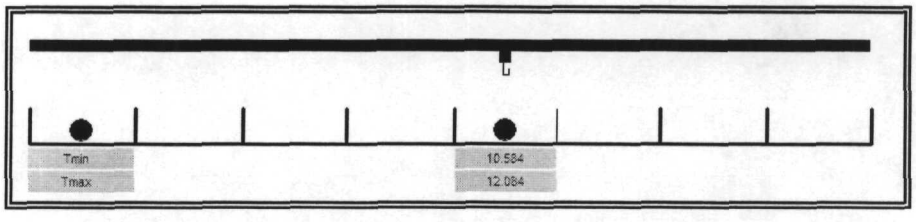


Figura 3.2. Nodo especifico.

Ramificación.

Se desarrolló un algoritmo el cual se encarga de enumerar las posibles opciones básicas de ramificación. Por ejemplo, en el Nodo mostrado en la figura 3.2, fácilmente se puede apreciar que sólo hay dos opciones básicas de ramificación, es decir, que a partir de ese nodo se pueden generar otros dos que serian considerados como nodos hijos del de la figura 3.2. La primera de esas dos opciones sería que la grúa metiera una pieza nueva al proceso. Y la segunda opción seria mover la pieza que ya esta dentro hacia el siguiente proceso de la secuencia. La representación grafica de esta expansión se presenta en la figura 3.3, en

donde el nodo de la figura 3.2 es representado como el nodo 1, y sus nodos hijos como 2 y 3.

Después los nuevos nodos generados a su vez también producirían más nodos y así continuaría la ramificación creciendo cada vez más en profundidad y en anchura.
El algoritmo diseñado siempre empieza con un nodo inicial, en donde no hay piezas en proceso y el tiempo de proceso es cero.

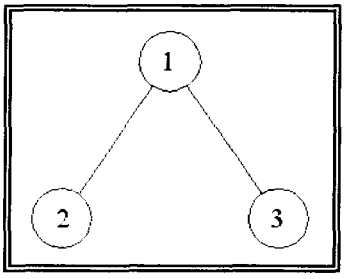


Figura 3.3. Ramificación de un Nodo.

Las reglas específicas para determinar los hijos se dan después.

3.3.2 Estrategia de búsqueda.

Se investigaron diferentes tipos de estrategias de ramificación, de las cuales las principales son *branch and bound*, DFS (Búsqueda Primero en Profundidad), y BFS (Búsqueda Primero en Anchura).

También se estudiaron otras estrategias de búsqueda que basan la decisión de que nodo expandir en una evaluación del nodo con respecto al nodo meta (Nodo meta es el nodo especifico que se está buscando). Pero en este caso, dada la complejidad del problema, no se tiene un nodo meta. Por lo que se decidió elegir una estrategia de ramificación básica para después agregarle heurísticos y reglas de poda para reducir el espacio de búsqueda y poder elegir el mejor nodo de los ramificados.

Se realizaron pruebas con las estrategias de búsqueda, BFS y DFS. Ambas estrategias realizan búsquedas completas, es decir buscan en todos los nodos posibles sin faltar uno solo. Ambas estrategias de búsqueda requieren de una gran capacidad de memoria, ya que al carecer de un nodo meta, estas retienen todos los nodos expandidos en memoria. El hecho de no conocer el nodo meta implica hacer algunos arreglos para limitar las búsquedas ya que estas se pueden hacer infinitas de tal suerte que pude llegar a terminarse la capacidad de memoria de la computadora y esto causaría que se inhiba.

La elección fue utilizar la estrategia BFS como base, aunque las dos estrategias tienen, en nuestro caso, las mismas ventajas y desventajas en cuanto a su funcionalidad, la razón por

la que se escogió la estrategia BFS es porque es más fácil de programar y ocupa menos líneas. Más adelante se describe como se limita y poda la búsqueda.

3.3.3 Reglas de ramificación de nodos.

Para cada nodo pueden existir dos posibles formas u opciones de ramificación y para cada nodo se expanden todas las posibles. Las opciones son las siguientes:

1. Mover material que ya este dentro de la línea (en caso de que haya).
2. Meter una pieza nueva a la línea (en caso de existir lugar disponible en la etapa 1).

1. Mover material dentro de la línea.

a) Mover pieza con tiempo máximo más crítico.

En esta opción de ramificación el algoritmo determina la posibilidad de mover el material que ya esta dentro de la línea con tiempo de proceso máximo más critico, el procedimiento es el siguiente:

1. Verifica si hay piezas dentro de la línea y cuales son sus tiempos de salida críticos.
2. En caso de existir material dentro de la línea, el algoritmo selecciona la pieza con tiempo de salida máximo más crítico, es decir, el tiempo de proceso más el tiempo de tolerancia más cercano al tiempo actual. Y en caso de no existir material dentro de la línea se da por terminada esta opción.
3. El siguiente paso es verificar si hay o no espacio en la etapa siguiente.
4. En caso de haber espacio en la siguiente etapa el algoritmo mueve la pieza y así crea un nodo hijo del nodo en ramificación.
5. En caso de no existir espacio disponible para la pieza, se da por terminada esta opción y el algoritmo no realiza la ramificación.

b) Mover piezas con tiempo máximo infinito.

Pueden existir tanques de proceso con tiempos máximos infinitos. Las piezas que estén dentro de esas etapas nunca serán vistas por el algoritmo como material con tiempo máximo de salida crítico, pero si como opciones para moverse. Es decir, todas las piezas que estén dentro de una de estas etapas (con tiempo máximo infinito) son vistas por el algoritmo como opciones de movimiento siguiendo el mismo procedimiento descrito en el inciso "a" para cada una de las piezas.

2. Introducir una pieza nueva a la línea.

Esta es la primera opción en la que el algoritmo busca introducir material nuevo a la línea, pueden existir dos posibles formas de expansión:

a) Cuando la línea esta vacía.

Esto es cuando no hay material en proceso y la forma en que trabaja es la siguiente:

1. Verifica si en el Nodo a expandir hay algún espacio disponible en la etapa 1.
2. Si hay espacio disponible es introducida una pieza nueva ramificando de esta manera un Nodo nuevo que es hijo del nodo en proceso de ramificación.
3. Si no hay espacio disponible se finaliza esta opción.

b) Cuando ya hay material en proceso.

En caso de existir material en proceso dentro de la línea se genera un rango de tiempo durante al cual puede ser introducida la pieza nueva. Este rango de tiempo es desde el tiempo actual del nodo hasta el tiempo máximo más crítico dentro de la línea. De tal forma que la nueva pieza puede ser introducida en cualquier instante de tiempo dentro de ese rango. Los lapsos de tiempo pueden ser tan pequeños o tan grandes como se desee multiplicando así el número de nodos expandidos resultantes del nodo en expansión. Este aspecto es muy importante y clave ya que dispara el factor de ramificación.

Por ejemplo, en la figura 3.5, se puede apreciar la representación de un nodo, en este nodo hay la opción de meter una pieza nueva a la línea, el rango de tiempo dentro del cual puede entrar la pieza nueva es desde el tiempo actual (5 minutos) hasta el tiempo máximo más crítico (38.613 minutos) lo cual nos da un rango de 33.613 minutos. Esto quiere decir que si se decidiera que los intervalos de tiempo de entrada fueran, por ejemplo, de 1 minuto se tendrían que expandir 33 nodos para la misma opción de meter una pieza nueva al sistema.

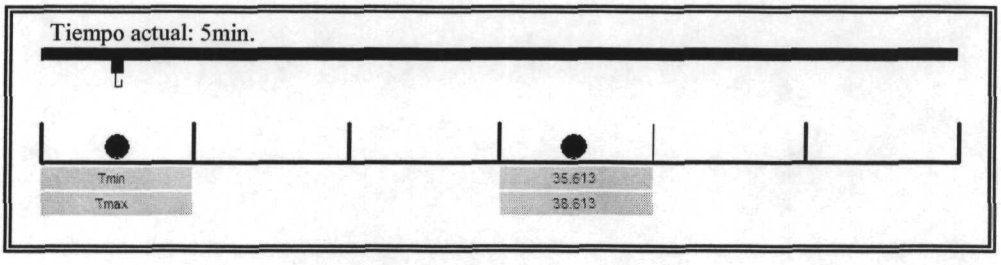


Figura 3.4. Ejemplo de opción múltiple.

Dadas estas circunstancias se determinó una forma de controlar este factor. Para esto, los tiempos de intervalo entre cada una de las entradas en esta opción se decidió que se calculara de la siguiente forma.

$$T_{\text{intervalo}} = \frac{T\min_{CB}}{10\ C_{CB}}$$

$T_{\text{intervalo}}$ = Tiempo entre cada entrada de la nueva pieza.

$T\min_{CB}$ = Tiempo de proceso mínimo en la estación cuello de botella.

C_{CB} = Numero de tanques existentes en la etapa cuello de botella.

El tiempo de intervalo en todos sus valores posibles debe ser mayor o igual al tiempo actual del nodo y menor que el tiempo de salida próximo dentro de la línea.

$$T_A \leq T_{\text{intervalo}} < T_{SP}$$

T_A = Tiempo actual del nodo.

T_{SP} = Tiempo de salida próximo dentro de la línea.

Nota: Estos mismos intervalos de tiempo no se pueden utilizar para mover piezas que ya están dentro de la línea, porque cuando una pieza termina su proceso y su tiempo de escurrimiento tiene que ir inmediatamente al siguiente proceso porque de lo contrario la pieza se pasiva.

Opciones en estaciones de tanques múltiples.

Después de definir las posibles ramificaciones estas se van a multiplicar por el número de tanques disponibles en la estación en la que se introducirá la pieza. Por ejemplo si es opción introducir una pieza nueva a la línea y existen 2 tanques disponibles en la etapa 1, entonces las posibles ramificaciones para introducir una pieza nueva son 2. En la Figura 3.6 se puede ver que la única opción posible es meter una pieza nueva, esta opción se duplica ya que en la etapa uno hay dos tanques disponibles. De tal forma que de este nodo se ramifican 2.

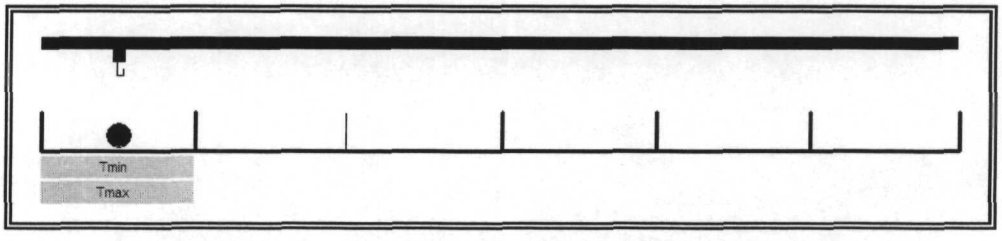


Figura 3.5. Dos opciones de introducir una pieza nueva a la línea.

Estas son todas las posibilidades de expansión de un solo nodo, es decir, todos los nodos pueden llegar a tener la posibilidad de tener esa cantidad de hijos. Y a su vez cada nodo nuevo ramificado puede tener también las mismas posibilidades de ramificación. A esto se le llama factor de ramificación, mientras más existan estaciones con tanques múltiples el factor de ramificación aumenta y la ramificación crece de forma exponencial.

Cuando el algoritmo ramifica algún nodo creando sus hijos, pasa toda la información del padre al hijo, la que se mantiene constante como puede ser los tiempos de salida de las piezas que no se movieron y la información que cambia el algoritmo la calcula y la pone en el nuevo nodo como puede ser el tiempo actual por ejemplo.

3.3.4 Reglas de poda.

La diferencia entre reglas de poda y heurísticos es que en las reglas de poda está comprobado matemáticamente que no se corre el riesgo de eliminar nodos buenos que pudieran llegar a ramificar algún nodo bueno. Y los heurísticos son reglas basadas en la experiencia o en un análisis intuitivo. Los heurísticos son de mucha utilidad pero se corre el riesgo de llegar a eliminar algún nodo bueno.

1) No producir desperdicio.

Al ramificar nuevos nodos, el algoritmo esta diseñado para no ramificar nodos que contengan piezas con desperdicio, es decir al realizar cada movimiento el algoritmo esta sujeto a restricciones las cuales no le permiten que el tiempo actual sobre pase los tiempos máximos de proceso dentro de la línea.
Sin embargo, esto no siempre es infalible porque puede ser que después de un movimiento dentro de la línea el tiempo actual no sobrepase ningún tiempo máximo de proceso, pero puede ser que la grúa este muy retirada de algún contenedor muy cercano al tiempo máximo, lo cual ocasiona que al moverse la grúa hacia ese contenedor, debido al tiempo que lleva el movimiento, el tiempo actual sobrepase el tiempo máximo del contenedor, esto automáticamente produce desperdicio.
Dadas estas circunstancias, adicionalmente después de ramificar cada nodo, el algoritmo checa que el tiempo actual resultante en los nuevos nodos creados, no sobrepase ningún tiempo de salida máximo dentro de la línea. En caso de presentarse esta situación, el algoritmo llama a una función llamada "Mata Hijo" la cual elimina al nodo en el que se produjo el desperdicio.

2) Reglas de poda.

A continuación se describen las reglas utilizadas, los heurísticos de poda se explicaran en el siguiente punto y más adelante se aplican también en el control de la búsqueda y selección de nodos ganadores.

En los algoritmos de búsqueda en los que no se sabe cual es el nodo meta el procedimiento a seguir es ramificar y escoger el mejor de los ramificados. Se ha comprobado que el HSP es un problema NP-duro [10], su factor de ramificación puede llegar a ser enorme. Debido a estos factores es de gran dificultad determinar reglas para podar las búsquedas. En este apartado se describen reglas de poda utilizadas.

Reglas de poda al introducir una pieza nueva a la línea.

En una búsqueda mientras más bajo sea el factor de ramificación más rápida es. Una de las causas que dispara el factor de ramificación es introducir una pieza nueva con retardos de tiempo, ya que el hacer esto lo aumenta drásticamente, y por consecuencia, en las pruebas realizadas, al ejecutar el algoritmo se tardaba mucho expandiendo nodos, y muchos de esos nodos no valían la pena porque pasos más adelante producían desperdicio y el algoritmo los eliminaba. Pero ¿como reducir este índice de ramificación sin eliminar nodos buenos? Se realizó un análisis de la entrada de material nuevo a la línea. Se encontró que en ocasiones, al introducir una pieza nueva a una línea que tenia material en proceso, esta pieza nueva llegaba al lugar en el que se encontraba el material introducido previamente, con el detalle de que esas piezas que entraron antes tardaban más en salir de lo que la pieza nueva en ocupar ese mismo lugar por lo que se creaba una inconsistencia ya que dos piezas tenían que utilizar el mismo recurso al mismo tiempo.

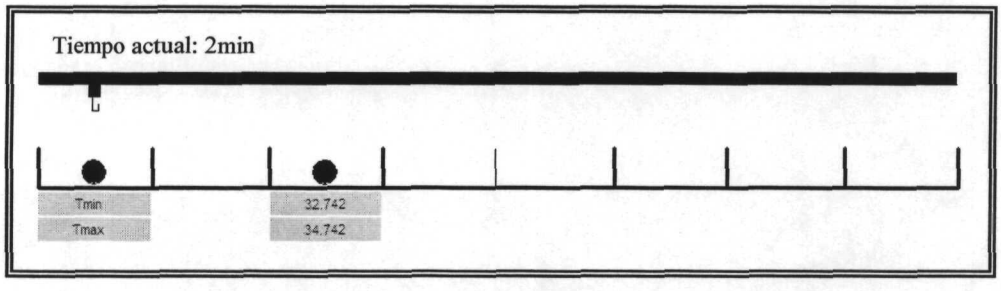


Figura 3.6. Pieza próxima a entrar produciendo desperdicio.

Por ejemplo en la figura 3.6, se muestra una pieza que ya esta en proceso en la estación 2, con tiempo de salida mínimo de 32.742min. y tiempo de salida máximo de 34.742min. Supongamos que el tiempo de procesamiento de la estación 1 es de 1min. Si se introduce una pieza nueva en el tiempo actual (2min.) este nodo terminaría produciendo desperdicio, ya que al terminar su procesamiento de la etapa 1 (terminaría aproximadamente en el minuto 3.5) tendría que pasar a la siguiente etapa, pero esta no se desocupa antes del minuto 32.742.

Con la opción de introducir una pieza nueva con retardos de tiempo, las opciones serian desde el tiempo actual hasta el tiempo mínimo de salida de la pieza que se encuentra en la

estación dos, en intervalos de tiempo definidos. Todas estas opciones representarían los nodos ramificados del nodo de la figura 3.7. De todos estos nodos creados algunos serian desperdicio en la siguiente ramificación por lo explicado antes. Pero algunos no producirían desperdicio, que serian los que entraron al final de los intervalos de entrada ya que estos, al terminar su procesamiento en la etapa 1 encontrarían disponible la etapa 2. De aquí se puede deducir una regla para las entradas a intervalos de tiempo a la línea.

Regla: Asegurar lugar en la siguiente estación ocupada.

Esta regla funciona de la siguiente manera, cuando el algoritmo va a buscar meter piezas nuevas a la línea realiza la siguiente secuencia:

1. Verifica si hay material dentro de la línea y cual es la estación ocupada más próxima a la entrada (físicamente).

2. Si no ha material dentro de la línea, procede a meter la pieza nueva en el tiempo actual.

3. Si hay material dentro de la línea, el algoritmo ya sabe cuál es la estación ocupada más cercana a la entrada. Verifica de cuántos tanques consta esa estación.

4. Si es múltiple checa si hay espacios disponibles.

5. Si hay espacios disponibles, se considera como si la estación estuviera libre y busca la siguiente estación ocupada y regresa al paso 3.

6. Si ya no encuentra estaciones ocupadas mete la pieza nueva en el tiempo actual.

7. Cuando ya se definió cual es la estación con material más cercana al inicio, calcula el tiempo de disponibilidad de esa estación.

$$T_{disp} = T\min_{sal} + Tesc + Tmov_{V} + Tmov_{H}$$

T_{disp} = Tiempo en que estará disponible la estación ocupada más cercana al inicio.

$T\min_{sal}$ = Tiempo mínimo de salida de la pieza que se encuentra en la estación ocupada más cercana al inicio.

$Tesc$ = Tiempo de escurrimiento de la pieza que se encuentra en la estación ocupada más cercana al inicio.

$Tmov_{V}$ = Tiempo que tarda la grúa en subir la pieza que se encuentra en la estación ocupada más cercana al inicio y bajarla en el siguiente proceso.

$Tmov_H$ = Tiempo que tarda la grúa en mover la pieza que se encuentra en la estación ocupada más cercana al inicio hasta el siguiente proceso y regresar hasta una estación antes de la estación ocupada más cercana al inicio.

8. El algoritmo calcula el tiempo en que estaría lista la pieza nueva para entrar a la estación ocupada más cercana al inicio.

$$Tlleg = T_A + \sum_{e1}^{eo-1} T\min_e + \sum_{e1}^{eo-1} Tesc_e + Tmov_V + Tmov_H$$

$Tlleg$ = Tiempo mínimo en que estaría lista la pieza nueva para entrar a la estación ocupada más cercana al inicio.

T_A = Tiempo actual del nodo a expandir.

$\sum_{e1}^{eo-1} T\min_e$ = Sumatoria de los tiempos de proceso mínimos desde la etapa 1 hasta una etapa antes de la estación ocupada más próxima al inicio.

$\sum_{e1}^{eo-1} Tesc_e$ = Sumatoria de los tiempos de escurrimiento desde la etapa 1 hasta una etapa antes de la estación ocupada más próxima al inicio.

$Tmov_V$ = Tiempo que tarda la grúa en subir y bajar la pieza nueva las veces que sean necesarias para llevarla desde el inicio hasta una estación antes de la estación ocupada más próxima al inicio.

$Tmov_H$ = Tiempo que tarda la grúa en mover la pieza nueva para llevarla desde el inicio hasta una estación antes de la estación ocupada más próxima al inicio.

9. Una vez calculados estos tiempos se debe de cumplir la siguiente regla.

Empezando desde el tiempo actual, para introducir una pieza nueva a la línea se debe satisfacer lo siguiente:

$$Tlleg > Tdisp$$

Si $Tlleg < Tdisp$ entonces ese nodo no se expande y se pasa al siguiente intervalo de tiempo, y así sucesivamente hasta satisfacer la regla.

De esta forma no se mete material en tiempos que se generaría desperdicio, el algoritmo de meter piezas a intervalos de tiempo trabaja de la misma forma solo que antes de expandir cada uno de los nodos que se crean checa esta regla.

Después de realizar pruebas con esta regla, se comprobó su funcionalidad, pero sin embargo, no era suficiente la poda que realizaba, ya que pasaba lo siguiente, normalmente las estaciones cuello de botella, son de más de un tanque, con los tiempos de proceso más grandes, por lo que estas estaciones son las que disparan más fuerte el factor de ramificación.

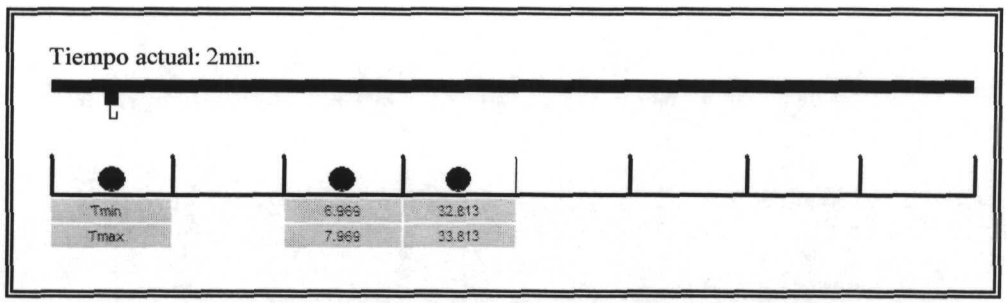


Figura 3.7. Pieza por entrar y generar desperdicio.

Por ejemplo, en la figura 3.7, se puede apreciar que hay dos piezas ya en proceso, la segunda esta en la etapa cuello de botella. Siguiendo la regla anterior se crearían solo los nodos que asegurarían su lugar en la etapa en que se encuentra la primer pieza, cumpliendo así con el objetivo de la regla anterior. Pero aquí el conflicto va más allá, ya que aunque la pieza asegura su lugar en la primer etapa ocupada, no lo asegura en la estación cuello de botella, ya que cuando esta nueva pieza termine los procesos anteriores al del cuello de botella, no encontrara lugar disponible en el cuello de botella, lo que generara desperdicio inevitablemente. Dadas esas circunstancias, es necesario crear una regla para que cuando se este llevando a cabo la Opción de meter material nuevo a la línea se asegure, aparte de un lugar en la primer estación ocupada, también un lugar en la etapa cuello de botella.

Regla: Asegurar lugar en la estación cuello de botella.

Esta regla funciona de la siguiente manera, cuando el algoritmo va a buscar meter piezas nuevas a la línea realiza la siguiente secuencia:

1. Verifica si hay o no material de la etapa cuello de botella hacia atrás.
2. Si no hay material mete la pieza en el tiempo actual.
3. Si hay material, calcula el tiempo en que llegaría la pieza nueva a la etapa cuello de botella.

$$Tlleg = T_A + \sum_{e1}^{eo-1} T\min_e + \sum_{e1}^{eo-1} Tesc_e + Tmov_V + Tmov_H$$

$Tlleg$ = Tiempo mínimo en que estaría lista la pieza nueva para entrar a la estación cuello de botella.

T_A = Tiempo actual del nodo a expandir.

$\sum_{e1}^{eo-1} T\min_e$ = Sumatoria de los tiempos de proceso mínimos desde la etapa 1 hasta una etapa antes de la estación cuello de botella.

$\sum_{e1}^{eo-1} Tesc_e$ = Sumatoria de los tiempos de escurrimiento desde la etapa 1 hasta una etapa antes de la estación cuello de botella.

$Tmov_V$ = Tiempo que tarda la grúa en subir y bajar la pieza nueva las veces que sean necesarias para llevarla desde el inicio hasta una estación cuello de botella.

$Tmov_H$ = Tiempo que tarda la grúa en mover la pieza nueva para llevarla desde el inicio hasta una estación antes de la estación cuello de botella.

4. Determina el número de contenedores en proceso que se encuentran desde la etapa 1 hasta la etapa cuello de botella (n_{1-CB}).

5. Calcular cuantos de los contenedores que ya están en proceso desde la etapa 1 hasta la etapa cuello de botella tienen que terminar su proceso en la etapa cuello de botella para asegurar lugar para la pieza nueva.

$$n_{cpCB} = \left(\frac{n_{1-CB}}{c_{CB}} \right) c_{CB}$$

n_{cpCB} = Numero de piezas dentro de la línea, entre la etapa 1 y la etapa cuello de botella, que deben ser procesadas, por el cuello de botella, para asegurar un lugar en la etapa cuello de botella para la pieza nueva.

c_{CB} = Capacidad (numero de tanques) de la etapa cuello de botella.

Nota: Esta formula debe realizarse en números enteros, tomando las bases de los números con fracciones decimales.

6. Se calcula el tiempo de disponibilidad de la estación cuello de botella para la pieza nueva. Esto se hace calculando en que tiempo se terminaran de procesar el numero de piezas mencionada en el punto anterior, de adelante hacia atrás, es decir, empezando por las piezas que ya están en el cuello de botella hacia atrás hasta completar el numero de piezas calculado en el punto anterior.

$$T_{dispCB} = \sum_{n_{1-CB}}^{n_{ejCB}} \left(\sum_{Eactual}^{ECB} T\min_{sal} + \sum_{Eactual}^{ECB} Tesc + Tmov_{1^{\cdot}} + Tmov_{H} \right)$$

El tiempo de disponibilidad de la estación cuello de botella (T_{dispCB}) es la sumatoria de todos los tiempos de proceso mínimos y movimientos necesarios para terminar el procesamiento de la etapa cuello de botella de los contenedores mencionados en el punto anterior.

10. Una vez calculados estos tiempos se debe de cumplir la siguiente regla.

Empezando desde el tiempo actual, para introducir una pieza nueva a la línea se debe satisfacer lo siguiente:

$$Tlleg > T_{dispCB}$$

Si $Tlleg < Tdisp$ entonces ese nodo no se expande y se pasa al siguiente intervalo de tiempo, y así sucesivamente hasta satisfacer la regla.

De esta forma se asegura que todas las piezas nuevas que entren tendrán un lugar asegurado en la producción al menos hasta la etapa cuello de botella, podando de esta manera el árbol de búsqueda eliminando nodos que de expandirse serian malos o producirían desperdicio.

El usar estas dos reglas propuestas evita meter material de más a la línea, por lo que el heurístico del *WIP* constante (explicado en el siguiente punto) puede ser omitido. Ya que con estas reglas es un hecho que no se eliminaran nodos prometedores, y con el heurístico si se corre ese riesgo.

3) Heurísticos de poda.

Los heurísticos reglas basadas en la experiencia o en un análisis intuitivo. Una de los heurísticos más comunes en las líneas de producción es mantener un *WIP* constante (*WIP* es descrito en el capitulo 2). Dada la complejidad del problema no se sabía de qué forma afectaría una regla como esta la calidad de las búsquedas o la producción así que se realizaron pruebas.

a) *WIP* máximo constante.

Para implementar un heurístico de *WIP* constante es necesario determinar el máximo posible, para perder el menor número de posibles nodos buenos. Este heurístico ayuda a podar la búsqueda no permitiendo la ramificación de posibles nodos malos que podrían generar desperdicio.

b) Heurístico de *WIP* máximo constante calculado.

El *wip* máximo constante se calculo de la siguiente forma:

$$WIP = th \cdot CTworst$$

$th =$ Es la velocidad de la línea; la velocidad máxima a la que puede trabajar una línea de producción es la velocidad del cuello de botella, el *throughput* es la velocidad de producción del cuello de botella en piezas por minuto, en este caso es la sumatoria de el tiempo que le lleva a la grúa mover un contenedor desde la etapa anterior al cuello de botella hasta la posterior, considerando el tiempo de proceso máximo y de escurrimiento de la estación cuello de botella. *Throughput* (descrito en el capitulo 2).

$$CTworst = \sum_{0}^{n} T\max + \sum_{0}^{n} Tesc + \sum_{0}^{n} Tmov_{V} + \sum_{0}^{n} Tmov_{H}$$

$CTworst =$ El peor tiempo que se llevaría procesar un contenedor desde la estación de carga hasta le estación de descarga, considerando que al inicio la grúa están en la estación de descarga y tiene que regresar por toda la línea para iniciar el proceso de la pieza.

Este calculo del WIP se hace al inicio de la ejecución del algoritmo, después cada vez que el algoritmo quiere introducir una pieza nueva al sistema verifica cuantas piezas ya están dentro si el numero es menor al WIP calculado entonces mete la nueva pieza, de lo contrario no ramifica el algoritmo ese nodo. De esta forma se realiza la poda del árbol de búsqueda.

Con este heurístico se hizo más rápida la búsqueda, pero tal vez se estaban eliminando nodos buenos. Después de analizar este heurístico se determino verificar el número del *wip* máximo. La forma en que se hizo fue por medio de búsquedas completas.

c) Heurístico de *WIP* máximo constante determinado por búsquedas completas.

Se realizaron búsquedas completas de dos horas de producción, después de esas dos horas se elegía el mejor nodo (más adelante se describe la forma de elegir el mejor nodo) y se revisaba toda su secuencia, desde el nodo inicial hasta el ganador, y se determinaba el *wip* máximo alcanzado durante esa secuencia.

En las pruebas realizadas los wip's calculados y determinados por búsqueda no siempre eran iguales, por lo que se decidió hacer el calculo de las dos formas y utilizar el que resultara mayor, al hacer esto la poda aumenta. De cualquier forma, en los heurísticos no se elimina el riesgo de podar nodos buenos, ayudan a reducir el área de búsqueda pero siempre se correrá ese riesgo, por lo que en la versión final de la metodología se omitieron los heurísticos de *wip* constante y se usan las reglas de entrada para material nuevo.

Más adelante, en el control de la búsqueda se describirán otros heurísticos para podar el árbol de búsqueda.

3.3.5 Control de la búsqueda y evaluación de nodos.

Por los datos obtenidos en las empresas visitadas, se determino que las búsquedas estarían limitadas a 8hrs. Por lo explicado anteriormente, en este tipo de problema no se sabe cual es el nodo meta, por lo que se debe ramificar y elegir el mejor.

Ramificar 8hrs de producción para después elegir el mejor, ocuparía mucha memoria y seria muy tardado, por lo que se decidió seccionar la búsqueda en periodos de tiempo. Después de realizar la búsqueda para cada periodo de tiempo se elige el mejor nodo de ese periodo de tiempo.

a) Primer enfoque de evaluación de nodos propuesto.

El primer enfoque propuesto para seleccionar el mejor nodo es elegir aquel que tenga mayor número de piezas producidas contando las fracciones de las piezas que estén en proceso, el método es el siguiente:

1. Se realiza una búsqueda BFS, con todas las reglas anteriores, hasta completar cierto periodo de tiempo (1 hora de producción por ejemplo).

2. En el árbol ramificado se busca cual es el nodo con la mayor producción.

3. Una vez que se tiene el nodo ganador se determina su secuencia hasta el nodo de inicio.

4. Se inicia una nueva búsqueda pero ahora el nodo inicial es el nodo ganador de la búsqueda anterior, la búsqueda será de 1 hora de producción, es decir, hasta las 2 horas. Después se regresa al paso 2 y así sucesivamente se repite el proceso hasta completar las 8 horas de producción. Ver figura 3.8.

Este enfoque propuesto fue poco funcional por lo siguiente:

Después de elegir el nodo ganador, el de mayor producción, y querer ramificarlo, este producía inmediatamente desperdicio. La razón es porque en este tipo de selección siempre

los ganadores son nodos "ambiciosos". A estos nodos los llamamos ambiciosos porque son los nodos ramificados en los que se mete material a la línea en las ultimas etapas de ramificación, este material, por las reglas propuestas anteriormente, tiene su producción asegurada hasta el cuello de botella pero no para toda la línea. Por lo que el desperdicio se produce pasos más adelante. Esto quiere decir que no siempre es el mejor el que tenga más piezas producidas y en proceso.

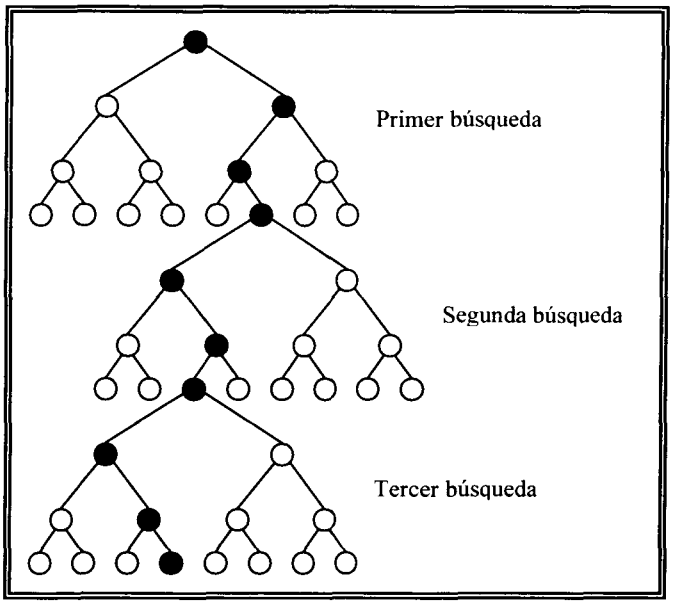


Figura 3.8. Zonas de búsqueda.

b) Segundo enfoque de búsqueda propuesto.

Dado que no fue muy exitosa la selección de nodos ganadores, se analizo el comportamiento del proceso para determinar una forma más eficiente de realizar la selección. Se decidió seguir utilizando el mismo esquema de búsqueda propuesto con la diferencia de que para seleccionar el mejor nodo no se haría por medio de la producción sino por la **utilización del cuello de botella**. Después de realizar cada búsqueda se elige el que tenga mayor utilización del cuello de botella y ese seria el ganador. Las desventajas encontradas en esta selección son las siguientes:

1. En este caso la selección por medio de la utilización es más eficiente, pero se siguen eligiendo ganadores ambiciosos.

2. Se probo este algoritmo con diferentes procesos, con algunos no funciono, porque el tiempo de ciclo era demasiado grande, es decir, el CT (Tiempo de Ciclo) era mayor

a 1 hora, por lo que en la primer búsqueda había ocasiones en que ninguna pieza había sido procesada por la etapa cuello de botella.

3. En ocasiones los nodos ganadores se encontraban muy por debajo del tiempo de búsqueda, es decir, por ejemplo si la búsqueda fue de la segunda hora, a veces el ganador estaba dentro de la primera hora, y esto no dejaba avanzar la búsqueda de una manera adecuada.

En las pruebas realizadas se determino que el método de evaluación por medio de la utilización de la etapa cuello de botella, es bueno, solo falta diseñar una mejor forma de aplicarlo.

c) Tercer enfoque de búsqueda propuesto.

Después de los resultados obtenidos en el segundo enfoque se llego a la conclusión de limitar el área de búsqueda. Y se propuso un nuevo arreglo para definir los tiempos de búsqueda y solucionar así los problemas presentados en el segundo enfoque.

Definición de los tiempos de búsqueda.

Para evitar que los nodos mejor evaluados caigan en una zona de tiempo no deseable, se acotaron las búsquedas en diferentes zonas de tiempo. Figura 3.9.

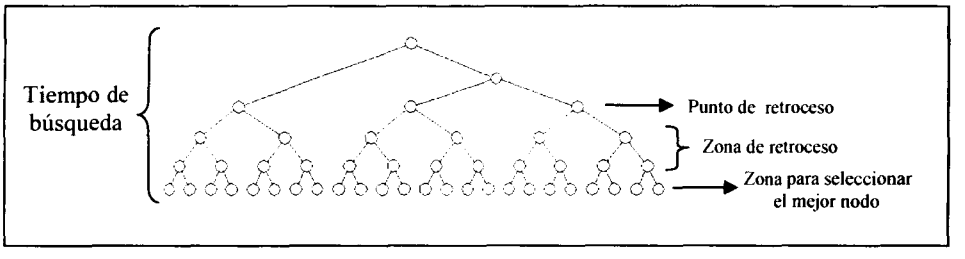


Figura 3.9. Zonas de búsqueda y retroceso.

Tiempo de búsqueda (T_{Bj}), donde j es el número de la búsqueda. Las búsquedas están organizadas en lapsos de 1 hr. y las búsquedas de secuencias están acotadas hasta 8 hrs. por lo que j puede tomar valores desde 1 hasta 8.

Zona para seleccionar el mejor nodo $(Z_{selccion})$, en esta zona de tiempo se encuentran todos los nodos que serán evaluados, de entre todos ellos se elegirá el que mayor utilización tenga del cuello de botella.

Zona de retroceso $(Z_{retroceso})$, es la zona por la que se hará un retroceso a través de los antepasados del nodo ganador hasta llegar lo más cerca posible del punto de retroceso $(P_{retrocesoj})$.

Asignación de valores a las zonas de búsqueda.

El objetivo de la búsqueda es ramificar en periodos de 1 hr. eligiendo el mejor de cada hora para iniciar con este la búsqueda siguiente. Pero al inicio esto no puede ser así de simple porque en ocasiones una hora no vasta para llegar al cuello de botella por lo que la primer búsqueda esta acotada diferente de las demás para dar oportunidad a la línea de que tenga un tiempo de calentamiento.

Para la primer búsqueda los valores son los siguientes:

- $T_{B1} = CT_{worst} + 2(T\max_{CB})$

  El tiempo de búsqueda es igual al peor tiempo de ciclo de la línea completa más dos veces el tiempo máximo de la estación cuello de botella.

- La zona de selección es desde $CT_{worst} + T\max_{CB}$ hasta T_{B1}. Todos los nodos en los que su tiempo actual cae dentro de esta zona será evaluada su utilización del cuello de botella. Eligiendo el que más alta utilización tenga.

- La zona de retroceso es en donde se realiza un regreso, por los antecesores, desde el nodo seleccionado hasta el punto de retroceso, que en la primer búsqueda, es igual a $P_{retroceso1} = 60\min$.

Una vez que se realiza el retroceso, el nodo que quede más cercano al punto de retroceso será le nodo ganador, de esta manera aseguramos de que en caso de que el nodo que se elija en el área de búsqueda se un nodo ambicioso, se retrocede a sus antepasados, para eliminar los nodos en que se metió material de más.

Después de que se tiene el nodo ganador, se borra todo el espacio de búsqueda dejando solo la secuencia desde el nodo inicial hasta el nodo ganador, y el nodo ganador será el nodo inicial de la siguiente búsqueda, que seria desde 1 hora hasta 2 horas.

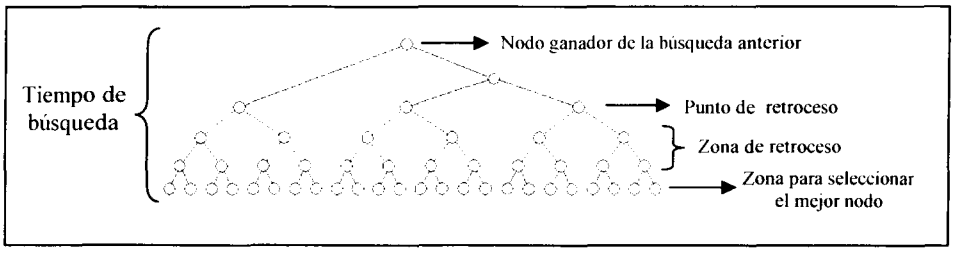


Figura 3.10. Zonas de las búsquedas posteriores a la primera.

Los valores de las zonas a partir de la segunda búsqueda son:

En la figura 3.10 podemos observar como se distribuyen las zonas de búsqueda, para las búsquedas posteriores a la primera.

- $T_{Bj} = j \cdot 60 + 2(T\max_{CB})$

  El tiempo de búsqueda inicia en el tiempo actual del nodo ganador de la búsqueda anterior, y termina en la suma de 60multiplicado por el número de la búsqueda, puede ser desde 2 hasta 8, más dos veces el tiempo máximo de la estación cuello de botella.

- La zona de selección es desde $j \cdot 60 + T\max_{CB}$ hasta T_{B1}. Todos los nodos en los que su tiempo actual cae dentro de esta zona será evaluada su utilización del cuello de botella. Eligiendo el que más alta utilización tenga.

- La zona de retroceso es en donde se realiza un regreso, por los antecesores, desde el nodo seleccionado hasta el punto de retroceso, que en las búsquedas posteriores a la primera es igual a $P_{retrocesoj} = j \cdot 60\min$.

Después de tener el nuevo nodo ganador se borran todos los nodos sobrantes dejando solo la secuencia desde el nuevo ganador hasta el ganador de la búsqueda pasada, y se repite el proceso de búsqueda para la siguiente hora. De esta forma poco a poco se va generando la secuencia hasta completar las 8 horas de producción.

Al terminar la ultima búsqueda, ya se tiene la secuencia generada para 8 horas de proceso.

4. IMPLEMETACION COMPUTACIONAL

Para poder realizar la implementación del algoritmo desarrollado, fue necesario hacer dos *softwares*.

El primero es el *software* de simulación, el cual esta diseñado para poder simular cualquier línea de electrodeposición con las características generales presentadas en este documento. Este *software* es una herramienta visual que permite entender de una manera más fácil el proceso, así como probar heurísticos y estrategias de producción, y posteriormente correr de una forma automática las secuencias generadas por el algoritmo propuesto. Este *software* es totalmente visual, es decir, a tiempo real se puede observar en pantalla todos los elementos de la línea y sus movimientos.

El segundo *software* es el algoritmo de búsqueda, este algoritmo es el que genera las secuencias de movimientos, al integrar este *software* con el de simulación, se puede ver los movimientos de la grúa con la secuencia resultante del algoritmo. El algoritmo de búsqueda fue descrito en el capitulo 3.

Ambos programas fueron desarrollados en ANSI C en LabWindows CVI 5.5 de National Instruments.

4.1 Descripción del software de simulación.

A continuación se describe la forma de uso y apariencia del software de simulación.

Al iniciar el software aparecerá la ventana principal, en esta ventana se encuentran todas las opciones, una vista de esta ventana se muestra en la figura 4.1. La ventana principal esta compuesta de varios controles y del área de simulación. El área en blanco es el área de simulación, en donde se generan las imágenes de la línea de producción.

Ahora se describirá cada uno de los controles de la ventana.

Generar Sistema.

El primer control que se utiliza es con el que se genera el sistema de electrodeposición, esto se realiza pulsando el botón *Generar* en la sección de Generar Sistema. En la figura 4.2 se muestra este botón.

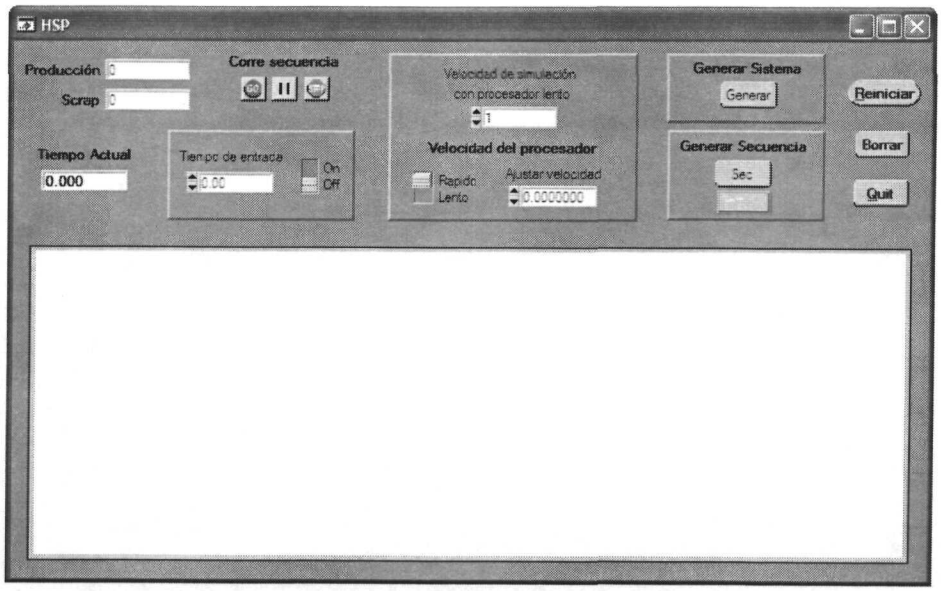


Figura 4.1. Ventana principal.

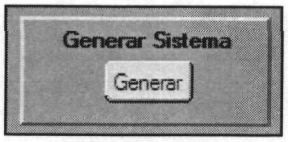


Figura 4.2. Botón para generar el sistema.

Al pulsar sobre el botón *Generar*, aparecerá una ventana como la que se muestra en la figura 4.3.

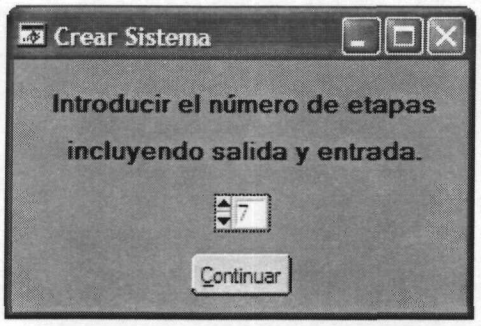


Figura 4.3. Numero de etapas.

En esta ventana se define el numero de etapas o estaciones (incluyendo la estación de carga y la estación de descarga) de las que consta el sistema a simular. El *software* esta programado para manejar un máximo de 40 tanques, por lo que el numero de etapas esta sujeto al numero de tanques permitido. En este caso como ejemplo para explicar el funcionamiento del *software* se definen 7 etapas.

Una vez que se define el número de etapas, se pulsa el botón *Continuar*, con lo que aparecerá la siguiente ventana. Figura 4.4.

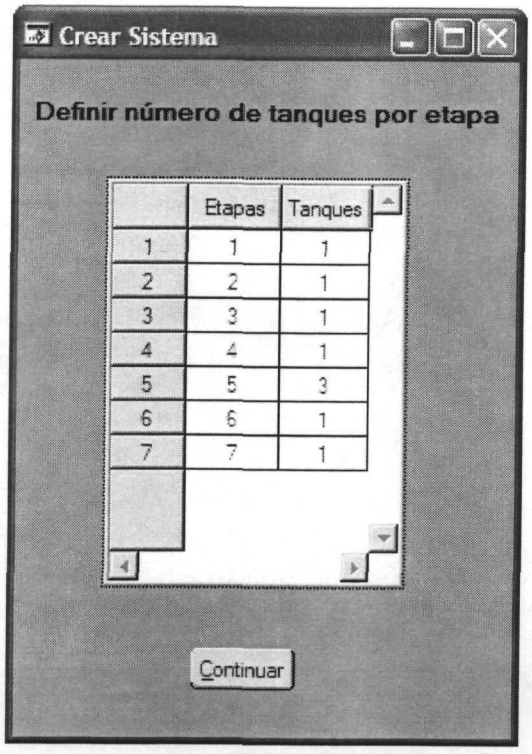


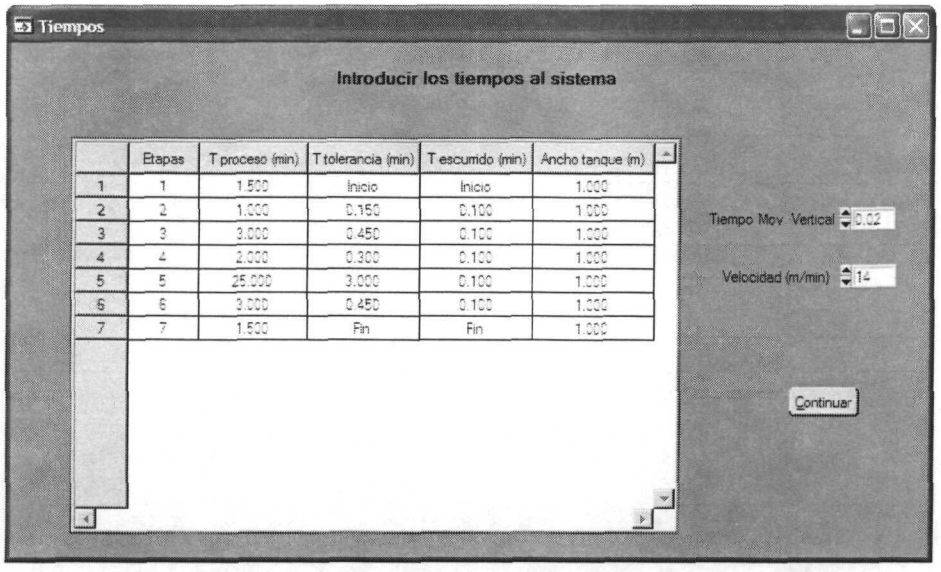
Figura 4.4. Definir capacidades de las estaciones.

Una vez que se ha definido el número de etapas, es necesario definir la capacidad para cada una de las etapas, es decir, el número de tanques para cada etapa.

Ya que se definieron las capacidades de las estaciones se pulsa sobre el botón *Continuar*, después aparecerá una ventana como la que se muestra en la figura 4.5.

En la ventana de diálogo, de la figura 4.5, se definen todos los tiempos del sistema. La primer columna indica el número de la etapa. En la segunda columna se definen los tiempos de proceso para cada una de las etapas, incluyendo la carga y la descarga. En la tercer columna se definen los tiempos de tolerancia para cada etapa, excepto para la carga y la descarga. En la cuarta columna se definen los tiempos de escurrido para cada etapa,

nuevamente sin considerar la carga y la descarga. Por ultimo en la quinta columna se define el ancho de los tanques para cada etapa.



	Etapas	T proceso (min)	T tolerancia (min)	T escurrido (min)	Ancho tanque (m)
1	1	1.500	Inicio	Inicio	1.000
2	2	1.000	0.150	0.100	1.000
3	3	3.000	0.450	0.100	1.000
4	4	2.000	0.300	0.100	1.000
5	5	25.000	3.000	0.100	1.000
6	6	3.000	0.450	0.100	1.000
7	7	1.500	Fin	Fin	1.000

Figura 4.5. Dialogo en que se definen los tiempos del sistema.

En este mismo diálogo hay dos *displays*, el primero es en donde se define el tiempo que tarda la grúa ya sea en bajar o subir algún contenedor de los tanques. Figura 4.6.

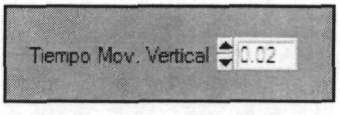


Figura 4.6. Tiempo de movimientos verticales.

En el segundo *display*, es en donde se define la velocidad horizontal de la grúa. Figura 4.7.

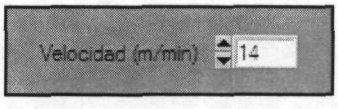


Figura 4.7. Velocidad horizontal de la grúa.

Todos los tiempos se definen en minutos y las distancias en metros.

Una vez que se han definido los tiempos, ya están definidas todas las características del sistema, solo hay que pulsar en el botón de *Continuar* y aparecerá la ventana principal pero ya con el sistema de electrodeposición creado. Figura 4.8.

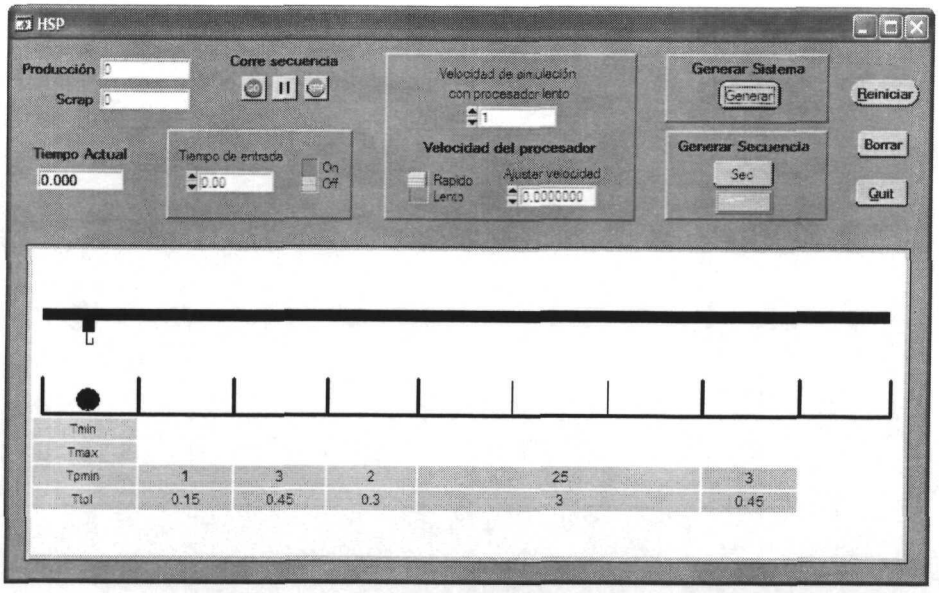


Figura 4.8. Sistema terminado.

Producción y Scrap.

En la ventana principal hay dos *displays* (figura 4.9) relacionados con la producción, ya sea manual o automática. El primero de ellos, el de Producción, muestra el número de contenedores que han terminado su proceso en la línea. El segundo *display* muestra el número de piezas que se han pasado del tiempo máximo en cualquiera de los procesos.

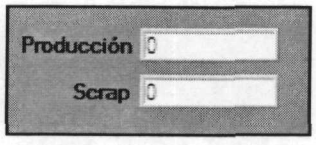


Figura 4.9. *Displays* de Producción y Scrap.

Tiempo Actual.

Este *display* muestra el tiempo, en minutos, transcurrido después de los movimientos en la línea, funciona en modo manual y en automático. Figura 4.10.

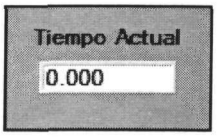


Figura 4.10. *Display* de Tiempo Actual.

Tiempo de entrada.

Esta opción solo se utiliza en modo manual, sirve para meter material en intervalos de tiempos a partir del tiempo actual, es decir, por ejemplo si el tiempo actual es de 20 minutos, y se quiere meter una pieza nueva no en el tiempo actual, sino en el minuto 25, se acciona el *switch* de esta opción para habilitar el *display*, el numero que se deberá poner en el *display* es 25, así se adelantara el tiempo actual 5 minutos y la pieza entrara en el minuto 25. Figura 4.11. Mientras el *switch* de esta opción este apagado, la entrada de las piezas nuevas será en el tiempo actual.

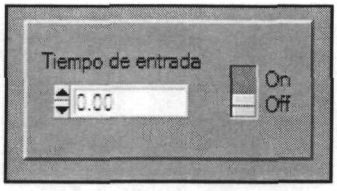


Figura 4.11. Tiempo de entrada.

Velocidad.

En el *software* se puede variar la velocidad de la simulación. El programa esta arreglado para que los movimientos de la grúa tengan la velocidad adecuada para la vista. Pero no todas las computadoras tienen la misma velocidad. Cuando un procesador es muy rápido los movimientos se ven demasiado rápidos, y cuando el procesador es muy lento, los movimientos se ven muy pausados. En la opción *Velocidad del Procesador* (figura 4.12) podemos mover algunos parámetros para ajustar la velocidad a un nivel adecuado para la vista. Podemos hacer la simulación muy lenta o muy rápida. El *switch* de esta opción es en donde es usuario define cual es la velocidad del procesador dependiendo de la velocidad de los movimientos.

Si el procesador es lento, entonces el *switch* se pone en la opción *Lento*, y la velocidad se aumenta incrementando el valor del *display* de la opción *Velocidad de simulación con procesador lento*. Así se puede hacer la simulación tan lenta o rápida como se desee.

Si el procesador es rápido, el *switch* se pone en la opción *Rápido*, la velocidad se disminuye aumentando el valor del *display Ajustar velocidad*. Y de esta forma se controla la velocidad de los movimientos.

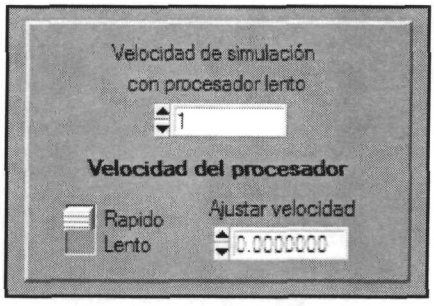


Figura 4.12. Velocidad de simulación.

Estas opciones de velocidad funcionan en modo manual y automático.

Generar secuencia.

Al pulsar sobre el botón *Sec* (Figura 4.13) se activa el algoritmo que realiza la búsqueda para determinar la mejor secuencia para el sistema previamente creado. En esta sección de la pantalla hay un foco que por *defoult* esta apagado, cuando la secuencia este terminada el foco se prende en color rojo.

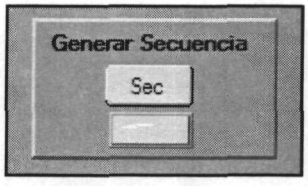


Figura 4.13. Generar secuencia.

Corre secuencia.

Una vez que la secuencia ya fue generada se puede iniciar producción en modo automático, esto se realiza pulsando sobre el botón *GO* de los controles de *Corre secuencia* (Figura 4.14). Al hacer esto automáticamente iniciara la grúa los movimientos de la secuencia. Para poder iniciar la producción en modo automático es necesario que el tiempo actual, la producción y el *scrap* estén en ceros.
Junto el botón *Go* están los de *Pausa* y *Stop* que sirven para pausar y detener la producción automática.

Figura 4.14. Correr secuencia.

Producción manual.

La producción manual se realiza indicándole a la grúa como mover los contenedores. Para que la grúa viaje a un tanque determinado so es necesario dar con el *mouse* un *click* izquierdo sobre el tanque deseado y la grúa se moverá actualizando el tiempo actual. Para subir o bajar contenedores solo es necesario posicionar la grúa sobre el tanque en que se encuentra o desea dejar algún contenedor y se da un *click* derecho sobre el tanque.

Reiniciar, Borrar, Salir.

Al pulsar el botón *Reiniciar* (Figura 4.15) se inicializan los *displays* de producción, como es el tiempo actual, la producción y el de scrap, poniéndose estos en cero.

El botón *Borrar*, elimina el sistema creado, dejando el área en blanco para poder definir un sistema nuevo.

El botón *Quit*, termina la aplicación del *software*.

Figura 4.15. Botones Reiniciar, Borrar y *Quit*.

4.2 Simulaciones.

En el software desarrollado, como ya se menciono anteriormente, puede simularse cualquier línea de electrodeposición como las mostradas en esta investigación. Los datos de todo el proceso son variables para que el usuario pueda configurarlos. Sin embargo, pueden darse casos en que el usuario quiera simular procesos muy rebuscados o caprichosos en sus restricciones de tiempos, esto puede realizarse pero tal vez sea necesario realizar las

configuraciones del horizonte de búsqueda desde el código del programa (no desde la interfase visual). En el Anexo 1 se muestran ejemplos de varios tipos de procesos con los cuales se puede configurar líneas de electrodeposición para varios tipos de recubrimientos, y que en su mayoría pueden ser configurados desde la interfase grafica. Estos tiempos pueden variar dependiendo el proceso y las piezas a recubrir.

5. RESULTADOS Y CONCLUSIONES

5.1 Equipo de trabajo utilizado.

Las pruebas se realizaron en una maquina Toshiba con las siguientes características:

- Toshiba Satellite
- Procesador: Intel Celeron 1.06GHz.
- 284 MB de Ram.

5.2 Experimentación para el desarrollo de la metodología.

Algoritmo de ramificación.

La primera fase del desarrollo de la metodología fue desarrollar el algoritmo de ramificación. Este proceso se creo en varios pasos.

1. Desarrollar un algoritmo que, con un nodo dado, determine todas las posibles opciones de ramificación.
2. Ramificar cada una de las opciones guardando los nodos expandidos en variables.
3. Organizar los nodos expandidos de tal forma que se puedan seguir expandiendo ya sea primero en anchura o en profundidad.

Para hacer funcionar este algoritmo se trabajo con un proceso teórico propuesto, ver tabla 5.1. Este proceso es muy sencillo ya que para efectos de experimentación fue más fácil hacerlo así, y una vez que funciono satisfactoriamente se probó con un proceso más complicado, el de la Empresa 1, y con más procesos propuestos. El sistema propuesto esta definido por los datos indicados en la tabla 5.1. Las unidades de tiempo son genéricas.

Tanques por etapa	1	3	2	1
Tiempo de proceso	2	15	4	2
Tiempo de tolerancia	1	3	2	1
Vel. vertical = 1 unidad de tiempo por cada movimiento.				
Vel. horizontal = 1 unidad de tiempo por cada tanque.				

Tabla 5.1. Proceso propuesto para experimentar con la ramificación.

Para probar que este algoritmo tenía un buen funcionamiento se realizo la ramificación del mismo proceso de forma manual, en pliegos de papel, escribiendo cada una de las opciones de ramificación y su descendencia durante un periodo de tiempo razonable.

Las primeras opciones de ramificación que se programaron contemplaban solo mover la próxima pieza en salir o introducir una nueva en el primer instante de tiempo posible. Cuando el algoritmo funciono favorablemente con estas reglas se incorporaron las demás reglas desarrolladas, que son las siguientes:

- Para mover material dentro de la línea el primer enfoque fue mover la próxima pieza en terminar su tiempo mínimo de proceso. Al analizar esto se determino que se estaban recortando posibilidades ya que se estaba desaprovechando el tiempo de tolerancia. Por esto se cambio el enfoque y ahora la pieza próxima a mover dentro de la línea es aquella que ya termino su tiempo de proceso mínimo y que esta más cerca del tiempo de proceso máximo.
- Otras opciones que no se estaban tomando en cuenta eran los procesos con tiempo de tolerancia infinito, ya que estos pueden realizar la función de *buffer* en el proceso, creando así más opciones de ramificación. En la versión final se incorporaron como opción de ramificación.
- Un factor muy importante en el proceso fue el introducir material nuevo a la línea. Para efectos de hacer funcionar el algoritmo de ramificación, se considero inicialmente que las piezas entraban a la línea en el primer instante de tiempo posible. Sin embargo cuando ya hay material dentro de la línea esto no tiene que ser forzosamente así ya que existe un rango de tiempo entre el primer instante de tiempo en el que se puede meter la pieza nueva y el tiempo máximo de proceso de la pieza más critica en salir (en caso de existir material dentro de la línea). Este rango de tiempo también fue aprovechado y se desarrollaron reglas para poder meter material con retardos de tiempo y así generar más opciones. En algunos casos la mejor secuencia hace uso de entradas retrasadas.
- Un punto interesante es que de todas las opciones de entrada dadas en la regla anterior, algunas producen desperdicio. En la versión final se incorporaron reglas para prever cuando una de estas opciones generara desperdicio en el futuro y evitar su ramificación.

Una vez integrados estos elementos el algoritmo funciono satisfactoriamente. Se continúo experimentando con más sistemas teóricos propuestos con las características necesarias para probar cada una de las reglas propuestas. Posteriormente se empezó a experimentar con procesos reales. En todos ellos funciono satisfactoriamente el método propuesto incluyendo las reglas desarrolladas, descritas brevemente en los puntos anteriores y reportadas detalladamente en el capitulo 3.

Evaluación de nodos.

Fueron propuestos varios enfoques para evaluar y seleccionar nodos. En su mayoría se probaron con el proceso de la Empresa 1, tabla 5.10. El primer enfoque fue elegir el nodo

con mayor producción pero no se tuvieron buenos resultados, ya que en ocasiones se elegía nodos que en las etapas finales incorporaban demasiado material al proceso, al expandir posteriormente estos nodos todos sus descendientes generaban desperdicio, a estos nodos se les llama nodos ambiciosos. Después se cambio el enfoque por otro que consistía en elegir el nodo de mayor utilización del cuello de botella, este enfoque dio mejores resultados pero el problema de los nodos ambiciosos persistía. Esto se soluciono finalmente dejando el enfoque de la utilización para seleccionar nodos y se definieron horizontes de búsqueda en los cuales después de elegir un nodo con la más alta utilización del cuello de botella se regresaba por sus antecesores para elegir uno que no fuera ambicioso.

En el capitulo 3 se describieron más a detalle los diferentes enfoques propuestos para la evaluación de los nodos. En cada uno de los enfoques se realizaron experimentos que consistían en expandir, elegir, y seguir expandiendo desde el nodo ganador de cada iteración. Estos experimentos se realizaron hasta que las búsquedas terminaban y se encontraba la secuencia de producción para 8 horas de trabajo.

Los resultados obtenidos fueron buenos pero se tenía que comprobar que la selección de nodos por este método fuera realmente eficiente. La forma en que se probó esto es la siguiente. Con las reglas de ramificación, poda y selección se realizaron 2 búsquedas, la primera hasta un tiempo determinado y la segunda al doble de tiempo de la primera, después, en ambas búsquedas, se elegía el mejor nodo con el enfoque de selección propuesto. En la figura 5.1 se muestra la primer búsqueda en donde se marca al nodo ganador y su secuencia.

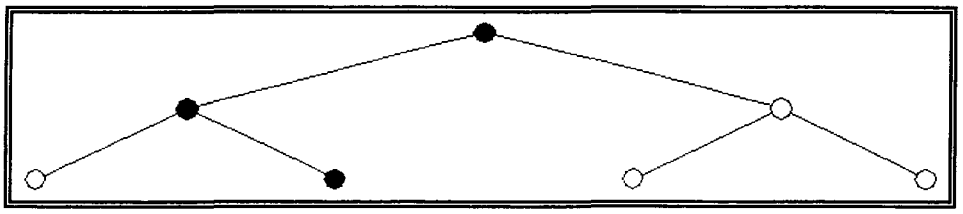


Figura 5.1. Nodo ganador y secuencia de la primer búsqueda.

Después se repetía el procedimiento con la segunda búsqueda, se ramificaba al doble de tiempo de la primera y se elegía el mejor nodo y su secuencia. Para comprobar que el enfoque de selección fuera bueno, el nodo elegido como ganador en la primer búsqueda debe ser parte de la secuencia del nodo ganador de la segunda búsqueda. Si esto ocurre entonces el enfoque de selección de nodos es bueno. Como ejemplo se muestra la figura 5.2, la segunda búsqueda en la que la secuencia del nodo ganador incluye la secuencia del nodo ganador de la primer búsqueda.

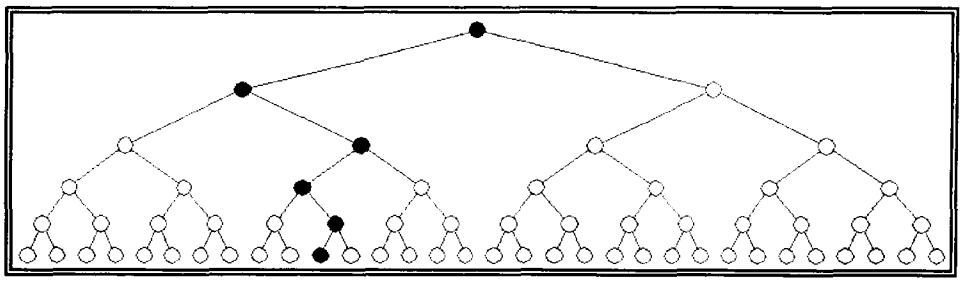


Figura 5.2. Nodo ganador y secuencia de la segunda búsqueda.

Estas pruebas se realizaron varias veces con el proceso de la Empresa 1 y con varios procesos teóricos propuestos. Los resultados fueron satisfactorios ya que en la mayoría de veces las secuencias de las segundas búsquedas incluían a las primeras.

Conclusiones.

Aquí se presentaron las experimentaciones más importantes que se realizaron para desarrollar el algoritmo. Varios sistemas teóricos fueron utilizados para la experimentación, aquí solo se presenta una muestra para mostrar como se fue desarrollando el método.

5.3 Evaluación de la metodología con el proceso *benchmark.*

La forma en que se evaluó la metodología fue simulando el modelo *benchmark* de Phillips y Unger, este proceso es usado de referencia en la mayoría de los artículos relacionados [10][3][4][5]. El problema consiste en 13 estaciones de un solo tanque cada una. El proceso *benchmark* se presenta en la tabla 5.2.

Tanque i	0	1	2	3	4	5	6	7	8	9	10	11	12
mi (seg)	120	150	90	120	90	30	60	60	45	130	120	90	30
Mi (seg)	∞	200	120	180	125	40	120	120	75	∞	∞	120	60
ri (seg)	30	31	22	22	22	25	23	22	22	22	47	27	22

Tabla 5.2. Proceso *benchmark* de Phillips y Unger.

Donde:

mi (seg) Tiempo mínimo de proceso.
Mi (seg) Tiempo máximo de proceso.

ri (seg) Tiempo de viaje con carga hacia el siguiente tanque.
Los viajes de la grúa vacía entre tanque y tanque pueden durar de 1 a 29 segundos.

Para simular el problema de Phillips y Unger se hicieron algunas adaptaciones ya que en ese sistema no se consideran algunas características y restricciones consideradas en esta investigación.

Características especiales del proceso *benchmark*:

- No utiliza una sola velocidad horizontal para la grúa.
- Para los movimientos horizontales sin carga no hay una velocidad fija, estos movimientos de tanque a tanque pueden variar de 1 a 29 segundos.
- Para los movimientos horizontales con carga se define un tiempo para cada tanque del proceso.
- No consideran tiempos de escurrimiento.
- No consideran ancho de tanques.

Adaptaciones del *benchmark* para simularlo con la metodología propuesta:

- Para los movimientos horizontales se definió una sola velocidad. Se realizaron varios experimentos dando a esta velocidad valores diferentes dentro del rango de tiempos para los movimientos sin carga del *benchmark*.
- A cada uno de los tiempos de movimiento horizontal con carga, de tanque a tanque, se les resto el tiempo que llevaría ese movimiento considerando la velocidad del punto anterior. El tiempo resultante fue añadido como tiempo de escurrimiento. De esta forma se consideraron los tiempos de movimiento con carga sin interferir con la velocidad definida para la grúa.
- Los anchos de los tanques se definieron de tal forma que las velocidades horizontales fueran iguales a la velocidad elegida de la grúa sin carga.

Tomando en cuenta estas ultimas consideraciones, de los experimentos realizados con el *benchmark*, los mejores resultados obtenidos fueron con los datos de la tabla 5.3.

Tanque	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Tmin (min)	0	2.5	1.5	2	1.5	0.5	1	1	0.75	2.17	2	1.5	0.5	0
Ttol (min)	-	0.83	0.5	1	0.58	0.17	1	1	0.5	∞	∞	0.5	0.5	-
Tesc (min)	-	0.5	0.35	0.35	0.35	0.4	0.37	0.35	0.35	0.35	0.77	0.43	0.35	-
Dist (m)	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016

Vel. Vertical 0 m/min
Vel. Horizontal 1m/min

Tabla 5.3. Proceso *benchmark* de Phillips y Unger adaptado.

Los resultados obtenidos con el *benchmark* fueron comparados con los de Lamothe [10] debido a que el HSP a solucionar por él es muy parecido al propuesto en esta investigación.

En la investigación de Lamothe [10] se define el *average period* como el tiempo de ciclo dividido por el *cycle length* (el numero de contenedores que entraron durante el tiempo de ciclo).

Los resultados mostrados por Lamothe [10] fueron obtenidos de 6 formas diferentes, en la tabla 5.4 se muestran estos resultados.

simple strategy	criterion	*average period*	*average computation time per DHSP*	*cycle length*
yes	makespan	652	7 cpu sec	1
no	makespan	561	143 cpu sec	3
yes	criterion 1	746	2 cpu	1
no	criterion 1	564	70 cpu sec	3
yes	criterion 2	731	1.1 cpu sec	1
no	criterion 2	564	67 cpu sec	3

Tabla 5.4. Resultados de Lamothe.

En esta tabla puede observarse que su mejor resultado es un *average period* de 561, esto quiere decir que en promedio la línea produce una pieza cada 561 segundos. Los resultados obtenidos con la metodología propuesta se muestran en la tabla 5.5.

average period	tiempo de calculo	cycle length
493	10800 segundos	4

Tabla 5.5. Resultados obtenidos en el *benchmark* con la metodologia propuesta.

El *average period* resultante mejoro los datos presentados por Lamothe [10], ya que con la secuencia generada se producen en promedio 1 contenedor cada 493 segundos; al correr esta simulación la grúa mantuvo en promedio 4 piezas en proceso dentro de la línea, la producción final fue de 46 contenedores en 8 horas de producción.

Para tener otro punto de referencia en la evaluación de la metodología con el proceso *benchmark*, se realizo un estudio de tiempos y movimientos de este proceso, el estudio consistió en determinar cual es la cota máxima de velocidad de la línea de proceso. Este estudio se presenta a continuación.

Análisis de tiempos y movimientos.

En el mejor de los casos la velocidad máxima de producción de una línea de proceso es igual a la velocidad de la estación más lenta de todas. Para determinar esta velocidad en el HSP se debe obtener la velocidad de proceso para cada una de las estaciones, esto se obtiene dividiendo el tiempo de ciclo de cada estación entre el número de tanques de cada estación. En la tabla 5.6 se muestran las velocidades de proceso para cada estación.

No. etapa	Tpo. mínimo	Tpo. tolerancia	Tanques por etapa	*Velocidad*
1	2.5	0.83	1	2.5
2	1.5	0.5	1	1.5
3	2	1	1	2
4	1.5	0.58	1	1.5
5	0.5	0.17	1	0.5
6	1	1	1	1
7	1	1	1	1
8	0.75	0.5	1	0.75
9	2.17	0	1	2.17
10	2	0	1	2
11	1.5	0.5	1	1.5
12	0.5	0.5	1	0.5

Tabla 5.6. Tiempos de proceso para la línea *benchmark*.
(tiempos en minutos, velocidad en minutos /pieza).

En este caso el proceso no tiene estaciones multitanque por lo que la velocidad de cada estación es igual al tiempo de proceso. La estación más lenta del proceso, el cuello de botella, es la numero 1. Para tener el valor total de tiempo que se tardaría el cuello de botella en procesar una pieza, es necesario sumar el tiempo de proceso y el tiempo que tarda la grúa en llevar una pieza de la etapa de entrada a la etapa 2 pasando por la etapa 1, contando los movimientos verticales y horizontales, y sumando el tiempo de escurrimiento de la etapa 1.

$$T = \sum Tmov_V + \sum Tmov_H + \sum Tesc + Tproceso_{CB}$$

$T =$ Tiempo que tarda el cuello de botella en procesar un contenedor.

$\sum Tmov_V =$ Sumatoria de los tiempos de los movimientos verticales para llevar una pieza desde la etapa anterior al cuello de botella hasta la posterior.

$\sum Tmov_H =$ Sumatoria de los tiempos de los movimientos horizontales para llevar una pieza desde la etapa anterior al cuello de botella hasta la posterior.

$$T = 0 + 0.032 + 0.5 + 2.5$$

$$T = 3.032 \min$$

Esto quiere decir que en **el mejor de los casos**, la línea estará produciendo una pieza a cada 3.032 minutos. Este tiempo es la cota máxima del *average period,* que en segundos seria 181.92. La relación entre la cota máxima y los resultados obtenidos con la metodología seria la siguiente:

$$\frac{\text{Cota máxima del avreage period}}{\text{Average period obtenido con la metodologia propuesta}} = \frac{181.92}{493} = 0.369$$

Esto significa que el *average period* de la secuencia generada con la metodología esta al 36.9% de la cota máxima.

El mejor de los casos se considera suponiendo lo siguiente:

- En la estación previa a la etapa cuello de botella siempre hay material disponible.
- En la estación posterior a la etapa cuello de botella siempre hay espacio disponible.
- La grúa siempre esta disponible para atender a la etapa cuello de botella.
- Que el cuello de botella tiene una utilización del 100%

Estas condiciones de trabajo para el cuello de botella son imposibles ya que para que el cuello de botella trabaje con una utilización del 100% es necesario que en cuanto una pieza sale del cuello de botella otra entra inmediatamente. Esto es imposible ya que se cuenta con una sola grúa. Aunque esta velocidad de producción es imposible de alcanzar, puede servir como referencia, ya que mientras más cercana sea la velocidad real a ese dato mejor será la secuencia.

Conclusiones.

Las secuencias obtenidas con la metodología propuesta son mejores que las presentadas por J. Lamothe [10], incluso supero la mejor secuencia cíclica reportada por este mismo autor. Una característica de esta comparación es que en la investigación de Lamothe [10] se supone que la carga y descarga se realiza en el mismo tanque y en este enfoque estos procesos se realizan en estaciones independientes. Para poder realizar una buena comparación se tomo el modelo *benchmark* y se ajusto lo mejor posible para poder simularlo en las condiciones más similares posibles a lo que lo hico Lamothe [10]. Para hacer estos ajustes solo se uso los rangos de valores de velocidades definidos en el *benchmark* de Phillips y Unger. La secuencia generada puede verse en el Anexo 2.

5.4 Evaluación de la metodología con un proceso multitanque con doble cuello de botella.

En el proceso *benchmark* todas las estaciones son de un solo tanque. La metodología propuesta esta diseñada para simular las características reales de producción. Una característica muy importante en la industria es que las estaciones cuello de botella normalmente son multitanque. Para poder probar esta característica varios sistemas teóricos fueron propuestos y simulados. Algunos arrojaron resultados más interesantes que otros debido a que unos tenían mayor complejidad que otros. A continuación se muestran los

resultados obtenidos con uno de los procesos teóricos con mayor dificultad simulados. Los datos del proceso se muestran en la tabla 5.7.

Tanques por etapa	1	1	2	1	3	1	1
Tiempo de proceso	1.5	3	30	3	45	3	1.5
Tiempo de tolerancia	-	0.45	4.5	0.45	6.75	0.45	-
Ancho de tanques (m)	1	1	1	1	1	1	1
Vel. vertical = 0.02 min. por cada movimiento							
Vel. horizontal = 14 m/min.							

Tabla 5.7. Proceso propuesto con 2 cuellos de botella.

Lo que hace que este proceso sea difícil de solucionar es que tiene dos etapas multitanque con tiempos de proceso altos, por lo que se considera que el proceso tiene 2 cuellos de botella, ya que los tiempos de procesamiento de las dos estaciones son iguales si se divide el tiempo de proceso de la estación entre el numero de tanques que la componen. Estos dos cuellos de botella hacen que el factor de ramificación sea más grande.

Para evaluar los resultados obtenidos con este proceso se utilizo los mismos criterios utilizados para evaluar la metodología con el *benchmark*. Los resultados obtenidos se muestran en la tabla 5.8.

Tiempo de cálculo	Producción en 8 Hrs.	*Average period*	*Cycle length*
4500 segundos	13 contenedores	1533	3

Tabla 5.8. Resultados obtenidos con el proceso con dos cuellos de botella.

Los datos de la tabla 5.8 muestran que el algoritmo tardo poco más de 1 hora para encontrar la secuencia de proceso. En promedio produce una pieza cada 1533 segundos. Durante la simulación en promedio se mantuvieron 3 contenedores en proceso. Para este sistema también se realizo un estudio de tiempos y movimientos para determinar cual seria la cota máxima de *average period*. En la tabla 5.9 se muestran las velocidades de proceso para cada estación de la línea.

No. etapa	Tpo. mínimo	Tpo. tolerancia	Tanques por etapa	*Velocidad*
1	3	0.45	1	3
2	30	4.5	2	15
3	3	0.45	1	3
4	45	6.75	3	15
5	3	0.45	1	3

Tabla 5.9. Tiempos de proceso para la línea propuesta con dos cuellos de botella. (tiempos en minutos, velocidad en minutos /pieza).

En este caso se puede ver en la tabla 5.9 que hay dos estaciones cuello de botella la 1 y la 4. Ambas estaciones tienen a sus costados estaciones con características iguales por lo que la cota máxima de velocidad puede determinarse con cualquiera de las dos estaciones, siguiendo el mismo procedimiento de cálculo que con el *benchmark*.

$$T = \sum Tmov_V + \sum Tmov_H + \sum Tesc + Tproceso_{CB}$$

$$T = 0.1 + 0.285 + 0 + 15$$

$$T = 15.385 \min$$

Por lo tanto la velocidad máxima en el mejor de los casos seria de 15.385 minutos por pieza, o un *average period* en segundos de 923.1. Como ya se menciono anteriormente esta velocidad de producción no es posible, ya que dadas las restricciones de la grúa no se puede tener un cuello de botella con utilización del 100%.

La relación entre la cota máxima y los resultados obtenidos con la metodología seria la siguiente:

$$\frac{\text{Cota máxima del avreage period}}{\text{Average period obtenido con la metodologia propuesta}} = \frac{923.1}{1533} = 0.602$$

Esto significa que el *average period* de la secuencia generada con la metodología esta al 60.2% de la cota máxima.

Conclusiones.

Ante simulaciones con procesos teóricos propuestos con etapas multitanque la metodología tuvo una respuesta muy favorable, los tiempos de cálculo varían dependiendo de la dificultad del proceso. También es importante recalcar que algunos procesos teóricos no tienen solución, esto es debido a que la relación entre tiempos de proceso y tiempos de movimientos no es coherente.

5.5 Evaluación de la metodología con procesos reales.

Para experimentar con los procesos reales ninguna de las dos empresas proporciono datos reales de producción por lo que se realizo un análisis como el desarrollado para el *benchmark*, para poder tener un punto de evaluación.

Proceso de la Empresa 1.

Cuando el algoritmo de búsqueda estaba en una etapa más avanzada se decidió empezar a experimentar con algún proceso real. El proceso en el que se enfoco fue uno no muy complejo, los datos de este proceso fueron proporcionados por la Empresa 1. Este proceso se presenta en la tabla 5.10.

Tanques por etapa	1	1	1	1	1	1	1	1	2	1	1	1
Tiempo mínimo (min.)	1.5	3	3	0.75	0.75	1	0.75	0.75	30	0.75	0.75	1.5
Tiempo de tolerancia (min.)	∞	0.45	0.45	0.112	0.112	0.15	0.112	0.115	4.5	0.112	0.112	∞
Tiempo de escurrimiento (min.)	∞	0.116	0.116	0.25	0.25	0.333	0.25	0.25	0.333	0.333	0.166	∞
Ancho de tanque (m)	1	0.92	1.12	1.72	1.72	0.92	1.72	1.72	1.12	1.72	1.72	0.8

Velocidad Horizontal (m/min.)	14
Tiempo de Mov. Vertical (min.)	0.02

Tabla 5.10. Proceso de galvanizado de la Empresa 1.

Esta línea de producción es atendida por una sola grúa, la carga y descarga se realizan en estaciones independientes. Este es un proceso que actualmente esta trabajando; la Empresa 1 no proporciono resultados de producción ni secuencias de movimientos contra las cuales se pudiera hacer una comparación directa.
En particular, el sistema proporcionado por la Empresa 1 fue con el que se probo la mayoría de reglas y heurísticos propuestos.

Análisis de tiempos y movimientos.

Se identifico el cuello de botella, usando el mismo procedimiento desarrollado con el proceso *benchmark*. Las velocidades de proceso para cada estación se muestran en la tabla 5.11

No. etapa	Tpo. mínimo	Tpo. tolerancia	Tanques por etapa	*Throughput*
1	3	0.45	1	3
2	3	0.45	1	3
3	0.75	0.1125	1	0.75
4	0.75	0.1125	1	0.75
5	1	0.15	1	1
6	0.75	0.1125	1	0.75
7	0.75	0.1125	1	0.75
8	30	4.5	2	15
9	0.75	0.1125	1	0.75
10	0.75	0.1125	1	0.75

Tabla 5.11. Tiempos de proceso para la línea de la Empresa 1.
(tiempos en minutos).

En la tabla 5.11 se puede observar que la estación más lenta es la 8, por tener el tiempo de proceso más grande. La velocidad de la línea en el mejor de los casos, incluyendo los tiempos de movimientos de la grúa, se calcula a continuación utilizando los datos de la tabla 5.10.

$$T = \sum Tmov_V + \sum Tmov_H + \sum Tesc + Tproceso_{CB}$$

$$T = 0.1 + 0.282 + 0.583 + 15$$

$$T = 15.965 \min$$

Esto quiere decir que la estación cuello de botella se tarda 15.965 minutos en procesar cada contenedor. Esta velocidad es la cota máxima para el *average period* que en segundos seria 957.9, en piezas por hora la velocidad seria 3.75 piezas/hr. Esta velocidad de producción no es posible, ya que dadas las restricciones de la grúa no se puede tener un cuello de botella con utilización del 100%.

Después de simular el proceso en el algoritmo propuesto se obtuvieron los resultados presentados en la tabla 5.12.

Proceso	Tiempo de cálculo	Producción en 8 Hrs.	*Average period*	*Cycle length*
Empresa 1	<10 segundos	16 contenedores	1455	2

Tabla 5.12. Resultados obtenidos con el proceso de la Empresa 1.

La relación entre la cota máxima y los resultados obtenidos con la metodología seria la siguiente:

$$\frac{\text{Cota máxima del avreage period}}{\text{Average period obtenido con la metodologia propuesta}} = \frac{957.9}{1455} = 0.658$$

Esto significa que el *average period* de la secuencia generada con la metodología esta al 65.8% de la cota máxima.

Los resultados obtenidos son buenos ya que el tiempo de calculo es muy corto en comparación con el tiempo que tardaría realizar estas secuencias por los métodos convencionales. Este es un punto de evaluación para la metodología propuesta, ya que el proceso es real. Desde el punto de vista producción no se puede realizar una evaluación directa ya que la empresa 1 no proporciono este tipo de datos. Pero la cota máxima de velocidad calculada y el *average period* sirven como referencia. Este proceso en la simulación mantuvo en promedio 2 piezas en proceso.

Proceso de la Empresa 2.

El proceso proporcionado por la Empresa 2 es un poco más complicado que el de la Empresa 1, ya que el factor de ramificación es más grande por tener más estaciones multitanque. El proceso es una línea de fosfatado y al igual que la Empresa 1 no proporcionaron datos de producción ni secuencias de movimientos. Los datos proporcionados por la Empresa 2 se muestran en la tabla 5.13. En este proceso los movimientos son realizados por una sola grúa y la carga y descarga se realizan en estaciones independientes.

Tanques por etapa	1	2	1	1	1	1	1	1	1	4	1	1
Tiempo mínimo (min.)	3	12.5	0.5	0.5	5	0.5	0.5	0.5	0.5	30	0.5	2
Tiempo de tolerancia (min.)	∞	1.875	0.075	0.075	0.75	0.075	0.075	0.075	0.075	4.5	0.075	∞
Tiempo de escurrimiento (min.)	∞	0.2	0.2	0.2	0.2	0	0	0	0	0	0	∞
Ancho de tanque (m)	1.079	1.079	1.079	1.079	1.079	1.079	1.079	1.079	1.079	1.079	1.079	1.079

Velocidad Horizontal (m/min.)	10.32
Tiempo de Mov. Vertical (min.)	0.2

Tabla 5.13. Proceso de fosfatado de la Empresa 2.

El proceso presentado en la tabla 5.13 actualmente se encuentra en operación, dado que no se tienen resultados reales de proceso, se realizo la misma evaluación que con el proceso de la Empresa 1.

Análisis de tiempos y movimientos.

Para este proceso también se realizo un estudio de tiempos y movimientos. En la tabla 5.14 se pueden ver los tiempos de proceso de las etapas de la línea de la Empresa 1.

No. etapa	Tpo. mínimo	Tpo. tolerancia	Tanques por etapa	*Throughput*
1	12.5	1.875	2	6.25
2	0.5	0.075	1	0.5
3	0.5	0.075	1	0.5
4	5	0.75	1	5
5	0.5	0.075	1	0.5
6	0.5	0.075	1	0.5
7	0.5	0.075	1	0.5
8	0.5	0.075	1	0.5
9	30	4.5	4	7.5
10	0.5	0.075	1	0.5

Tabla 5.14. Tiempos de proceso para la línea de la Empresa 1.
(tiempos en minutos).

En este caso la estación más lenta es la numero 9, con un tiempo de proceso de 7.5 minutos.

Usando los datos de la tabla 5.13 se calcula el tiempo total que le lleva a la estación cuello de botella, **en el mejor de los casos**, procesar una pieza.

$$T = \sum Tmov_V + \sum Tmov_H + \sum Tesc + Tproceso_{CB}$$

$$T = 1 + 0.522 + 0 + 7.5$$

$$T = 9.022 \min$$

Por lo tanto el tiempo que se lleva la estación cuello de botella en procesar una pieza es de 9.022 minutos. Esta velocidad es la cota máxima para el *average period* que en segundos seria 541.32, en piezas por hora la velocidad seria 6.65 piezas/hr. Esta velocidad de producción no es posible, ya que dadas las restricciones de la grúa no se puede tener un cuello de botella con utilización del 100%.

Para generar la secuencia para este proceso, el algoritmo tardo un poco más debido al factor de ramificación que provocan las estaciones multitanque. Los resultados obtenidos se muestran en la tabla 5.15.

Proceso	Tiempo de cálculo	Producción en 8 Hrs.	Average period	*Cycle length*
Empresa 2	7200 segundos	11	1811	2

Tabla 5.15. Resultados obtenidos con el proceso de la Empresa 2.

La relación entre la cota máxima y los resultados obtenidos con la metodología seria la siguiente:

$$\frac{\text{Cota máxima del avreage period}}{\text{Average period obtenido con la metodologia propuesta}} = \frac{541.32}{1811} = 0.298$$

Esto significa que el *average period* de la secuencia generada con la metodología esta al 29.8% de la cota máxima.

Al igual que con la Empresa 1, no se puede hacer una comparación directa con la producción, ya que estos datos no fueron proporcionados, pero se puede afirmar que el tiempo que tardo el algoritmo en generar la secuencia es mucho más corto de lo que se tardaría con métodos convencionales. La cota máxima de velocidad calculada y el *average period* pueden servir como referencia para analizar los resultados de la simulación.

Conclusiones sobre procesos reales.

El algoritmo genera buenas secuencias para los proceso industriales con los que se probo, el tiempo de calculo es corto en comparación con los tiempos que lleva hacer cambio a un producto nuevo en las empresas que se dedican al negocio de la electrodeposición. Una de las principales ventajas de la metodología propuesta es que se desarrollo pensando en los procesos reales de tal forma que se toma en cuenta las restricciones reales y pueden ser configurables.

Las personas contactadas en la industria tuvieron una buena opinión acerca de la metodología y de lo útil que podría ser en sus procesos, estuvieron muy interesados en la investigación y ambas empresas visitadas propusieron continuar el trabajo de investigación con ellos enfocándose principalmente a sus procesos.

En este caso en particular no se tienen datos reales de producción de estas líneas, pero el análisis de los procesos para calcular la cota máxima de *average period* es muy útil para poder realizar una evaluación de las secuencias generadas por la metodología propuesta.

Uno de los principales problemas que muestra la industria de la electrodeposición en México, es que el producto no es uniforme antes de ser procesado. En otros países cuando el producto no es uniforme el trabajo se rechaza. En México esto no pasa, aunque el producto no este uniforme se acepta el trabajo, dadas estas circunstancias no se puede procesar todo el material con una misma secuencia de tiempos y movimientos. Este problema actualmente es solucionado a tiempo real, cada vez que se introduce un nuevo contenedor o *rack* a la línea se determinan con reglas empíricas los tiempos de proceso dependiendo de las condiciones del material, lo cual hace la producción mucho más lenta de lo normal. Una solución propuesta para este problema es agrupar los *racks* o contenedores con las mismas características y usar el algoritmo para generar las secuencias de los grupos formados de material.

5.6 Conclusiones generales e investigaciones futuras.

Sobre la metodología propuesta

1) Las secuencias obtenidas por la metodología propuesta supera los resultados mostrados por Lamothe [10] en su investigación.

2) La metodología se adapta a las restricciones de procesos industriales reales. En particular aplica a sistemas multitanque tan comunes en la industria.

3) Puede ser ejecutada en computadoras personales.

4) Dada la forma en que opera (simula el proceso en condiciones reales) puede ser extendida a sistemas multiproducto y multigrua.

5) Su ejecución sin necesidad de expertos en optimización la hace una alternativa excelente para uso en la industria maquiladora mexicana en la cual las condiciones de operación no son constantes sino que cambian de cliente a cliente y en muchas ocasiones entre lotes de un mismo cliente.

Sobre la herramienta computacional.

Permite la simulación del HSP con las siguientes características.

a) Se puede configurar fácilmente cualquier sistema de producción real en forma de "I".

b) Genera secuencias de movimiento con la metodología propuesta sin necesidad de experto en optimización o Inteligencia Artificial.

c) Permite simular visualmente la ejecución de la secuencia generada así como la ejecución de secuencias manuales introducidas por el usuario. Esta característica de simulación visual fue muy apreciada por los industriales entrevistados.

Investigaciones futuras.

1) Extender la metodología a otras configuraciones, especialmente:

- Entradas y salidas en un mismo extremo de la línea.
- Procesos multiproducto.
- Procesos multigrua.
- Procesos con tanques intercalados.

2) Generar mejores heurísticos de poda y limitación de ramificación para reducir los tiempos de búsqueda.

3) Proponer mejores reglas para selección del horizonte de búsqueda.

4) Integrarla al sistema de control de movimientos de una grúa real.

5) Mejoras a la herramienta computacional.

Anexo 1

Procesos en general.

1. **Baños previos a la electrodeposición de metal.**

    1. Desengrase por inmersión (3 – 30min)
    2. Desengrase electrolítico (3 – 30min)
    3. Enjuague (1 – 2min)
    4. Enjuague (1 – 2min)
    5. Activado (1 – 15min)
    6. Enjuague (1 – 2min)
    7. Enjuague (1 – 2min)

2. **Electrodeposición de material. (Puede ser uno de los siguientes procesos)**

    Cobrizado
    1. Cobrizado (5 – 60min)
    2. Enjuague (1 – 2min)
    3. Enjuague (1 – 2min)
    4. Activado (1 – 15min)
    5. Enjuague (1 – 2min)
    6. Enjuague (1 – 2min)

    Niquelado
    1. Niquelado (1 – 30min)
    2. Recuperado de níquel (2 – 4min)
    3. Enjuague (1 – 2min)
    4. Enjuague (1 – 2min)

    Latonado
    1. Latonado (2 – 60min)
    2. Enjuague (1 – 2min)
    3. Enjuague (1 – 2min)

    Cromado
    1. Cromado (30seg – 15min)
    2. Recuperado de cromo (2 – 4min)
    3. Enjuague (1 – 2min)
    4. Enjuague (1 – 2min)
    5. Enjuague (1 – 2min)
    6. Enjuague (1 – 2min)

3. Etapa final del proceso.

1. Sellado (1 – 4min)
2. Enjuague (1 – 2min)
3. Secado (5 – 30min)

Estos procesos y sus tiempos, no necesariamente tienen que seguir el orden en que se presentan, o ser todos aplicados, la secuencia de procesos y los tiempos pueden variar dependiendo de la pieza a recubrir y del recubrimiento deseado.

Anexo 2

Secuencia generada con la metodología propuesta para el *benchmark* de Phillips y Unger.

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
1	0	1
2	1	2
3	0	1
4	2	3
5	1	2
6	3	4
7	2	3
8	4	5
9	5	6
10	3	4
11	6	7
12	0	1
13	4	5
14	7	8
15	5	6
16	8	9
17	1	2
18	6	7
19	7	8
20	2	3
21	9	10
22	8	9
23	3	4
24	0	1
25	4	5
26	5	6
27	1	2
28	10	11
29	6	7
30	9	10
31	2	3
32	7	8
33	11	12
34	8	9
35	12	13
36	3	4
37	0	1

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
38	4	5
39	5	6
40	1	2
41	10	11
42	6	7
43	9	10
44	2	3
45	7	8
46	11	12
47	8	9
48	12	13
49	3	4
50	10	11
51	9	10
52	4	5
53	5	6
54	11	12
55	6	7
56	12	13
57	7	8
58	0	1
59	8	9
60	1	2
61	2	3
62	3	4
63	0	1
64	4	5
65	5	6
66	1	2
67	10	11
68	6	7
69	9	10
70	2	3
71	7	8
72	11	12
73	8	9
74	12	13

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
75	3	4
76	0	1
77	4	5
78	5	6
79	1	2
80	10	11
81	6	7
82	9	10
83	2	3
84	7	8
85	11	12
86	8	9
87	12	13
88	3	4
89	0	1
90	4	5
91	5	6
92	1	2
93	10	11
94	6	7
95	9	10
96	2	3
97	7	8
98	11	12
99	8	9
100	12	13
101	3	4
102	0	1
103	4	5
104	5	6
105	1	2
106	10	11
107	6	7
108	9	10
109	2	3
110	7	8
111	11	12
112	8	9
113	12	13
114	3	4
115	10	11
116	9	10
117	4	5
118	5	6
119	11	12
120	6	7
121	12	13

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
122	7	8
123	0	1
124	8	9
125	1	2
126	2	3
127	3	4
128	0	1
129	4	5
130	5	6
131	1	2
132	10	11
133	6	7
134	9	10
135	2	3
136	7	8
137	11	12
138	8	9
139	12	13
140	3	4
141	0	1
142	4	5
143	5	6
144	1	2
145	10	11
146	6	7
147	9	10
148	2	3
149	7	8
150	11	12
151	8	9
152	12	13
153	3	4
154	0	1
155	4	5
156	5	6
157	1	2
158	10	11
159	6	7
160	9	10
161	2	3
162	7	8
163	11	12
164	8	9
165	12	13
166	3	4
167	0	1
168	4	5

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
169	5	6
170	1	2
171	10	11
172	6	7
173	9	10
174	2	3
175	7	8
176	11	12
177	8	9
178	12	13
179	3	4
180	10	11
181	9	10
182	4	5
183	5	6
184	11	12
185	6	7
186	12	13
187	7	8
188	0	1
189	8	9
190	1	2
191	2	3
192	3	4
193	0	1
194	4	5
195	5	6
196	1	2
197	10	11
198	6	7
199	9	10
200	2	3
201	7	8
202	11	12
203	8	9
204	12	13
205	3	4
206	0	1
207	4	5
208	5	6
209	1	2
210	10	11
211	6	7
212	9	10
213	2	3
214	7	8
215	11	12

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
216	8	9
217	12	13
218	3	4
219	0	1
220	4	5
221	5	6
222	1	2
223	10	11
224	6	7
225	9	10
226	2	3
227	7	8
228	11	12
229	8	9
230	12	13
231	3	4
232	0	1
233	4	5
234	5	6
235	1	2
236	10	11
237	6	7
238	9	10
239	2	3
240	7	8
241	11	12
242	8	9
243	12	13
244	3	4
245	10	11
246	9	10
247	4	5
248	5	6
249	11	12
250	6	7
251	12	13
252	7	8
253	0	1
254	8	9
255	1	2
256	2	3
257	3	4
258	0	1
259	4	5
260	5	6
261	1	2
262	10	11

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
263	6	7
264	9	10
265	2	3
266	7	8
267	11	12
268	8	9
269	12	13
270	3	4
271	0	1
272	4	5
273	5	6
274	1	2
275	10	11
276	6	7
277	9	10
278	2	3
279	7	8
280	11	12
281	8	9
282	12	13
283	3	4
284	0	1
285	4	5
286	5	6
287	1	2
288	10	11
289	6	7
290	9	10
291	2	3
292	7	8
293	11	12
294	8	9
295	12	13
296	3	4
297	10	11
298	9	10
299	4	5
300	5	6
301	11	12
302	6	7
303	12	13
304	7	8
305	0	1
306	8	9
307	1	2
308	0	1
309	2	3

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
310	1	2
311	3	4
312	2	3
313	4	5
314	5	6
315	3	4
316	6	7
317	10	11
318	9	10
319	4	5
320	7	8
321	5	6
322	11	12
323	8	9
324	6	7
325	12	13
326	10	11
327	7	8
328	9	10
329	8	9
330	11	12
331	12	13
332	10	11
333	11	12
334	9	10
335	12	13
336	0	1
337	1	2
338	0	1
339	2	3
340	1	2
341	3	4
342	2	3
343	4	5
344	5	6
345	3	4
346	6	7
347	0	1
348	4	5
349	7	8
350	5	6
351	8	9
352	1	2
353	6	7
354	10	11
355	7	8
356	2	3

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
357	9	10
358	8	9
359	11	12
360	12	13
361	3	4
362	0	1
363	4	5
364	5	6
365	1	2
366	10	11
367	6	7
368	9	10
369	2	3
370	7	8
371	11	12
372	8	9
373	12	13
374	3	4
375	0	1
376	4	5
377	5	6
378	1	2
379	10	11
380	6	7
381	9	10
382	2	3
383	7	8
384	11	12
385	8	9
386	12	13
387	3	4
388	0	1
389	4	5
390	5	6
391	1	2
392	10	11
393	6	7
394	9	10
395	2	3
396	7	8
397	11	12
398	8	9
399	12	13
400	3	4
401	0	1
402	4	5
403	5	6

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
404	1	2
405	10	11
406	6	7
407	9	10
408	2	3
409	7	8
410	11	12
411	8	9
412	12	13
413	3	4
414	10	11
415	9	10
416	4	5
417	5	6
418	11	12
419	6	7
420	12	13
421	7	8
422	0	1
423	8	9
424	1	2
425	2	3
426	3	4
427	0	1
428	4	5
429	5	6
430	1	2
431	10	11
432	6	7
433	9	10
434	2	3
435	7	8
436	11	12
437	8	9
438	12	13
439	3	4
440	0	1
441	4	5
442	5	6
443	1	2
444	10	11
445	6	7
446	9	10
447	2	3
448	7	8
449	11	12
450	8	9

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
451	12	13
452	3	4
453	0	1
454	4	5
455	5	6
456	1	2
457	10	11
458	6	7
459	9	10
460	2	3
461	7	8
462	11	12
463	8	9
464	12	13
465	3	4
466	10	11
467	9	10
468	4	5
469	5	6
470	11	12
471	6	7
472	12	13
473	7	8
474	0	1
475	8	9
476	1	2
477	0	1
478	2	3
479	1	2
480	3	4
481	2	3
482	4	5
483	5	6
484	3	4
485	6	7
486	10	11
487	9	10
488	4	5
489	7	8
490	5	6
491	11	12
492	8	9
493	6	7
494	12	13
495	7	8
496	10	11
497	9	10

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
498	8	9
499	11	12
500	12	13
501	10	11
502	11	12
503	12	13
504	0	1
505	1	2
506	0	1
507	2	3
508	1	2
509	3	4
510	9	10
511	2	3
512	4	5
513	5	6
514	3	4
515	6	7
516	0	1
517	4	5
518	7	8
519	5	6
520	8	9
521	1	2
522	6	7
523	10	11
524	7	8
525	2	3
526	9	10
527	8	9
528	11	12
529	12	13
530	3	4
531	0	1
532	4	5
533	5	6
534	1	2
535	10	11
536	6	7
537	9	10
538	2	3
539	7	8
540	11	12
541	8	9
542	12	13
543	3	4
544	0	1

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
545	4	5
546	5	6
547	1	2
548	10	11
549	6	7
550	9	10
551	2	3
552	7	8
553	11	12
554	8	9
555	12	13
556	3	4
557	0	1
558	4	5
559	5	6
560	1	2
561	10	11
562	6	7
563	9	10
564	2	3
565	7	8
566	11	12
567	8	9
568	12	13
569	3	4
570	0	1
571	4	5
572	5	6
573	1	2
574	10	11
575	6	7
576	9	10
577	2	3
578	7	8
579	11	12
580	8	9
581	12	13
582	3	4
583	10	11
584	9	10
585	4	5
586	5	6
587	11	12
588	6	7
589	12	13
590	7	8
591	0	1

Número de movimiento	Número de tanque del que se saca contenedor	Número de tanque al que se mete contenedor
592	8	9
593	1	2
594	2	3
595	3	4
596	0	1
597	4	5
598	5	6
599	1	2
600	10	11
601	6	7
602	9	10
603	2	3
604	7	8
605	11	12
606	8	9
607	12	13
608	3	4
609	0	1
610	4	5
611	5	6
612	1	2
613	10	11
614	6	7
615	9	10
616	2	3
617	7	8
618	11	12
619	8	9
620	12	13
621	3	4

REFERENCIAS

[1] **Luis Martín Cavazos Salazar.** *TESIS Desarrollo de técnicas de cambio rápido de producción en plantas de galvanizado de tuberías.* Diciembre de 1995.

[2] **Centro Mexicano para la producción más limpia.** http://www.cmpl.ipn.mx/Area_Tecnica/Mumo.PDF

[3] **J. Lamothe, M. Correge and J. Delmas.** *A dynamic heuristic for the real time hoist scheduling problem.* Emerging Technologies and Factory Automation, 1995. ETFA '95, Proceedings., 1995 INRIA/IEEE Symposium on , Volume: 2 , 1995 Page(s): 161 -168 vol.2

[4] **Rodosek R., Wallace M. G.** *A Generic Model and Hybrid Algorithm for Hoist Scheduling Problems* Proceedings of the 4th International Conference on Principles and Practice of Constant Programming, pg 385-399, LNCS 1520, Pisa, 1998

[5] **J. Lamothe, C. Thierry and J. Delmas.** *A Multihoist model for the real time Hoist Scheduling problem.* International Multiconference on Computational Engineering in Systems Applications. CESA'96 Lille Juillet 1996.

[6] **Haoxun Chen, Chengbin Chu, Proth J. M.** *Cyclic hoist scheduling based on graph Theory.* Emerging Technologies and Factory Automation, 1995. ETFA '95, Proceedings., 1995 INRIA/IEEE Symposium on, Volume: 1, 1995 Page(s): 451 -459 vol.1

[7] **Varnier C., Grunder O., Baptiste P.** *Improving the productivity of electroplating lines by changing the layout of the tanks.* Emerging Technologies and Factory Automation, 1995. ETFA '95, Proceedings., 1995 INRIA/IEEE Symposium on , Volume: 1 , 1995 Page(s): 441 -450 vol.1

[8] **Bloch C., Manier M. A.** *Notation and typology for the hoist scheduling problem.* Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on, Volume: 6 , 1999 Page(s): 475 -480 vol.6

[9] **Baptiste P., Legeard B., Varnier C.** *Hoist Scheduling Problem: an Approach Based on Constraints Logic Programming.* Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on, 1992 Page(s): 1139 -1144 vol.2

[10] **Lamothe J., Correge M., Delmas J.** *Hoist Scheduling Problem in a real time context.* Conference internationale sur l'analyse et l'optimisation des systèmes, Sophia antipolis, 1994.

[11] **C. Cheng and S. Smith.** *A Constraint-Posting Framework for Scheduling Under Complex Constraints.* Emerging Technologies and Factory Automation, 1995. ETFA '95, Proceedings., 1995 INRIA/IEEE Symposium on, Volume: 1, 1995 Page(s): 269 - 280 vol.1.

[12] **Collart Dutilleul S., Denat J. P.** *P-time Petri nets and the hoist scheduling problem.* Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on, Volume: 1, 1998 Page(s): 558 -563 vol.1.

[13] **Kats V., Levner E, Meyzin L.** *Multiple-part cyclic hoist scheduling using a sieve method.* Robotics and Automation, IEEE Transactions on, Volume: 15 Issue: 4, Aug. 1999 Page(s): 704 -713

[14] **Jegou D., Baptiste P., Lee K.H.** *Multiagent Systems for the hoist scheduling Problem.* Fuzzy Systems, 2001. The 10th IEEE International Conference on, Volume: 2, 2001 Page(s): 581 -585.

[15] **Tesis Digitales.** Universidad de las Américas-Puebla Este sitio forma parte de la iniciativa NDLTD www.ndltd.org Actualizado por: Lourdes Fernández Ramírez 9 de septiembre de 2002

[16] **D'ALOIA** http://www.daloia.com.ar/galvanotecnia/index.htm. Bermúdez 1271-73, 1407, Buenos Aires. Tel.: (54-11) 4567-4884, Fax: (54-11) 4567-2008.

[17] **Sawyer and Smith Cororation.** *Electroplating and electrophoresis.* http://www.electroplate.cc/ Last modified: September 06, 2002 10:36:28 AM.

[18] **Stuart J. Russell and Peter Norvig.** *Artificial intelligence: a modern Approach.* Englewood Cliffs, N.J.: Prentice Hall, c1995.

[19] **Universidad Fasta** De la fraternidad de agrupaciones Santo Tomas de Aquino http://www.ufasta.edu.ar/.

[20] **The University of Liverpool.** Department of Computer Science, http://www.csc.liv.ac.uk. Chadwick Building, Peach Street, Liverpool L69 7ZF last updated 29 April 2002 13:47

[21] **National Centre for Software Technology.** http://www.ncst.ernet.in/ KBCS-98 Secretariat Wed May 5 11:50:48 GMT 1999 Last Updated: Tuesday, October 22, 2002

[22] **Frontline Systems.** Frontline Systems Inc. http://www.frontsys.com/algomip.htm. Last modified: December 01, 1996.

[23] **Universidad de Granada.** Escuela Técnica Superior de Ingeniería Informática Departamento de Ciencias de la Computación e Inteligencia Artificial. http://decsai.ugr.es/~castro/CA/CA.html

[24] **Mokhtar S. Bazaraa, John J. Jarvis, Hanif D. Sherali.** *Linear programming and network flows*. Second edition. Ed. WIE WILEY.