INSTITUTO TECNOLOGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY
CAMPUS ESTADO DE MEXICO

IMPLEMENTACION DE UN ALGORITMO GENETICO PARA LA CALENDARIZACION DE SISTEMAS DE PRODUCCION TIPO JOB SHOP.

TESIS QUE PRESENTA

DANIEL VILLA CORONA

MAESTRIA EN CIENCIAS EN SISTEMAS DE MANUFACTURA

MSMA 95

AGOSTO, 2003

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

CAMPUS ESTADO DE MÉXICO

IMPLEMENTACIÓN DE UN ALGORITMO GENÉTICO PARA LA CALENDARIZACIÓN DE SISTEMAS DE PRODUCCIÓN TIPO JOB SHOP.

TESIS QUE PARA OPTAR EL GRADO DE
MAESTRO EN CIENCIAS EN SISTEMAS DE MANUFACTURA
PRESENTA

DANIEL VILLA CORONA

Asesor DR. DANTE JORGE DORANTES GONZÁLEZ

Comité de tesis: DR. HUMBERTO VAQUERA HUERTA
M. en C. FRANCISCO J. SANDOVAL PALAFOX

Jurado	DR. HUMBERTO VAQUERA HUERTA	Presidente
	M. en C. FRANCISCO J. SANDOVAL PALAFOX	Secretario
	DR. DANTE JORGE DORANTES GONZÁLEZ	Vocal

Atizapán de Zaragoza, Edo. Méx., Agosto del 2003

DEDICATORIAS

A mis padres Daniel Villa Carrillo y Guadalupe Corona Hernández y a mi hermana Guadalupe Villa Corona por su apoyo incondicional y sus palabras de aliento que me llevaron a realizar este proyecto de la mejor manera.

RECONOCIMIENTOS

Al Instituto Tecnológico y de Estudios Superiores de Monterrey Campus Estado de México por darme la oportunidad de realizar mis estudios de maestría

Al Dr. Dante Jorge Dorantes González por dirigir este proyecto

Al Dr. Humberto Vaquera Huerta por su ayuda y apoyo en todo momento

Al Dr. Carlos Alberto Brizuela Rodríguez por su ayuda y apoyo en todo momento.

Al Dr. Emil Liebermann por todo su apoyo

RESUMEN

La Industria manufacturera en la economía de México reclama la mejora continua de los procesos presentes en esta industria, Una adecuada calendarización puede reducir significativamente los costos de producción y reducir los tiempos de proceso lo que permite cumplir con los compromisos de tiempo de entrega, por tal motivo la búsqueda de métodos de calendarización de tareas a través de un ambiente de producción y la implementación de los mismos en sistemas capaces de obtener una calendarización óptima, son esenciales en los procesos de manufactura.

En este trabajo se presenta la aplicación de los Algoritmos Genéticos como una técnica eficaz para desarrollar un sistema de calendarización, para lo cual, primero se sientan las bases de Algoritmos Genéticos en cuanto a su funcionamiento y desempeño, para luego mostrar su aplicación especifica en calendarización Job Shop partiendo de las diferentes representaciones de calendarización existentes y distintos operadores genéticos (cruzamiento y mutación), que son el punto de partida para el desarrollo del Algoritmo Genético. Esta parte concluye con la presentación de métodos de calendarización Job Shop con Algoritmos Genéticos que se han desarrollado por investigadores en el tema.

Finalmente en esta tesis se muestra el desarrollo de la implementación de un Algoritmo Genético para la calendarización de *n* trabajos en *m* máquinas en un sistema de producción tipo Job Shop. El desarrollo del programa del algoritmo genético se creo en lenguaje C, se planteó que fuera dinámico, esto es, que fuera flexible para calendarizar problemas de *n* trabajos y *m* máquinas con la finalidad de resolver problemas de dimensiones rectangulares, es decir que el número de trabajos sea mayor al número de máquinas o viceversa y así lograr extender el espacio de problemas a resolver. La aplicación se hizo en Visual Basic de una manera muy gráfica y sencilla en función de su manejo, ya que esta creada para facilitar la introducción de datos del problema, así mismo favorece la visualización de la calendarización por medio de una gráfica de Gantt, además muestra un reporte detallado de los resultados obtenidos del algoritmo Genético.

La aplicación que se presenta en esta tesis es robusta, con una interfaz gráfica de fácil manejo para el usuario.

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

1 INTRODUCCIÓN

1.1 ANTECEDENTES

México es uno de los países industrializados de América latina. La industria genera un cuarto del Producto Interno Bruto y proporciona los trabajos para un décimo de la fuerza laboral mexicana.

En el año 2001 el PIB Nominal del Sector Industrial (Minería, Manufacturas, Construcción y Electricidad) sumó 1'399,473 millones de pesos a precios corrientes, con lo cual representó el 26.8% del total. El crecimiento real del Producto Interno Bruto en el año 2002 fue de 1.7%

Dentro del Sector Industrial el Sector Minero fue de 71,795 millones de pesos, lo que equivale al 1.4% del PIB global. La Industria Manufacturera generó un PIB de 1'012,778 millones de pesos a precios corrientes; con ello este sector participó con el 19.4% del producto de la economía en el 2001. En cuanto al PIB de la Industria de la Construcción, éste alcanzó un valor de 254,086 millones de pesos a precios corrientes. De esta manera, esta industria aportó 4.8% del PIB global en el 2001. El PIB Nominal de la Generación de Electricidad, Gas y Agua, ascendió a 60,814 millones de pesos durante el año pasado, aportando 1.2% del producto [1].

Desde 1996 la gran división de la industria manufacturera es la de mayor crecimiento de la rama industrial. Sin embargo no ha logrado recuperarse de la política cambiaria del sexenio 1988-1994, que al sobrevaluar el tipo de cambio provocó el desplazamiento de las manufacturas nacionales por las importaciones, afectando a la industria y las exportaciones. Las consecuencias de la imposibilidad de invertir, financiarse y adquirir tecnología, por parte de la pequeña y mediana empresa, llevaron a México a convertirse en un país, al menos 46% maquilador en el mercado exterior, con todas las desventajas que representa esto [2]. Sin embargo el sector de la manufactura participo con el 19.4% del producto de la economía en el 2001. Este hecho ratifica la importancia de la industria manufacturera en la economía del país.

El Sector manufacturero mexicano está compuesto de nueve divisiones de actividad económica, los cuales se enlistan a continuación:

- Productos alimenticios, bebidas y tabaco
- Textiles, prendas de vestir e industria del cuero
- Industria de la Madera y Productos de Madera
- Papel, Productos de Papel, Imprentas y Editoriales
- Sustancias Químicas, Derivados del Petróleo, Productos de Caucho y Plásticos
- Productos de minerales no metálicos. Excluye los derivados del petróleo y del carbón
- Industrias metálicas básicas
- Productos metálicos, maquinaria y equipo.
- Otras Industrias Manufactureras

La división de "Productos metálicos, maquinaria y equipo" presenta la tasa de crecimiento más dinámica de las nueve divisiones de la actividad económica de la producción manufacturera, pues tiene un crecimiento promedio aproximado de 18% anual desde 1996, y el 2001 participo con el 31.1% de la actividad económica de la industria manufacturera.

La importancia de la Industria manufacturera en la economía de México, reclama la mejora continua de los procesos presentes en esta industria, por tal motivo la búsqueda de métodos de calendarización de tareas a través de un ambiente de producción es esencial en los procesos de manufactura, debido a que cada variación en las condiciones de mercado obliga a la industria hacer frente a cambios dinámicos en la demanda anual de productos. En la industria manufacturera usualmente se manejan un número incontrolable de factores de producción tales como variaciones en los tiempos de arribo de partes, el índice de fallas de máquina y el índice de retrabajo de partes después de la inspección; los efectos de estas variaciones en el sistema de manufactura pueden ser drásticos y las pérdidas causadas por estos factores son significativas si los comparamos con otros factores de la producción [3]. Sin embargo con una adecuada calendarización se pueden reducir significativamente los costos de producción y reducir los tiempos de proceso permitiendo cumplir con los compromisos de tiempo de entrega.

La calendarización de tareas es un problema que puede ser descrito básicamente como la asignación de recursos limitados a ciertas tareas u operaciones a través de cierto período de tiempo. El problema se encuentra en diferentes ambientes de manufactura, por lo que se requiere de sistemas capaces de resolver todas las dificultades encontradas para optimizar el desempeño global de una planta o sistema de producción.

En el ambiente de manufactura las órdenes que son liberadas por el sistema de producción se transforman en tareas con un tiempo de entrega asociado. Esos trabajos tienen que ser procesados en un centro de trabajo donde previamente se les dio una orden o secuencia. La secuencia de tareas se forma en base a las restricciones que relacionan tareas entre sí, recursos con tareas, o actividades con eventos externos al sistema de calendarización [4]. Por ejemplo, puede haber restricciones de precedencia que relacionan a dos actividades determinadas entre sí, que indican que una actividad debe ser realizada antes que otra, pudiendo incluir por cuanto tiempo de anticipación debe precederla. Otra restricción podría ser que dos actividades no puedan utilizar el mismo recurso al mismo tiempo, o que dos recursos no puedan ser utilizados al mismo tiempo durante una parte del proceso, o por una misma actividad. Un caso más sería la imposibilidad de usar cierto recurso durante un lapso de tiempo por actividades de mantenimiento.

La función de calendarización en un sistema de producción debe interactuar con otras funciones de toma de decisión principalmente con el sistema de información de planeación de requerimientos materiales MRP. Después de que la secuencia de tareas se ha generado es necesario que todos los materiales y recursos estén disponibles en el tiempo especificado. Las fechas de inicio de las tareas tienen que ser determinadas conjuntamente por la planeación de la producción, por el sistema de secuencia de tareas y por el sistema MRP. Un sistema de planeación de requerimientos materiales MRP determina el número de partes, componentes y materiales necesarios para producir cada artículo, además mantiene vigilado el inventario de todos estos elementos. La MRP también provee el programa de tiempo que especifica cuando debe ordenarse o producirse cada uno de los materiales partes o componentes [5].

En la figura 1-1 se muestra el flujo de información, y el papel que juega la calendarización en un sistema de manufactura. En la figura se observa que las órdenes liberadas por los clientes, se convierten en trabajos con fechas de entrega al pasar por el bloque de planeación de la producción. Para que estos trabajos sean procesados, se les tiene que dar un orden o secuencia. Esta secuencia empieza a tomar forma en el sistema de planeación de requerimientos materiales (MRP), el cual asigna la fecha de liberación de las órdenes, en base al nivel de inventarios, pronósticos de la demanda y al requerimiento de recursos. Las órdenes con fecha de liberación entran al bloque de calendarización donde se determina el orden de la ejecución de tareas. Esta secuencia entra al bloque de Despacho, en el cual se calendariza la secuencia de tareas, es decir se asignan fechas de inicio y terminación, en base a su tiempo de procesamiento y a las condiciones del sistema de producción. Con la información de la calendarización de tareas, se inicia en el taller la ejecución del trabajo programado.

El problema del calendarización de tareas se hace más difícil si consideramos el hecho de que las máquinas pueden fallar, la materia prima puede faltar, o nuevas tareas pueden llegar. La forma más común de tratar el problema es a través de relajación, esto es, quitar algunas de las restricciones y abordar el problema resultante [6].

Un sistema de programación de tareas que sea flexible a cambios en las suposiciones de estabilidad sería sumamente útil. En la práctica, las tareas no tardan siempre la misma cantidad de tiempo en cada máquina según se programa. Las variaciones en los tiempos pueden depender de fallas de las máquinas, de operadores humanos, faltas de materia prima, o por mantenimiento; estos eventos son típicamente tratados como ocurrencias aleatorias [5].

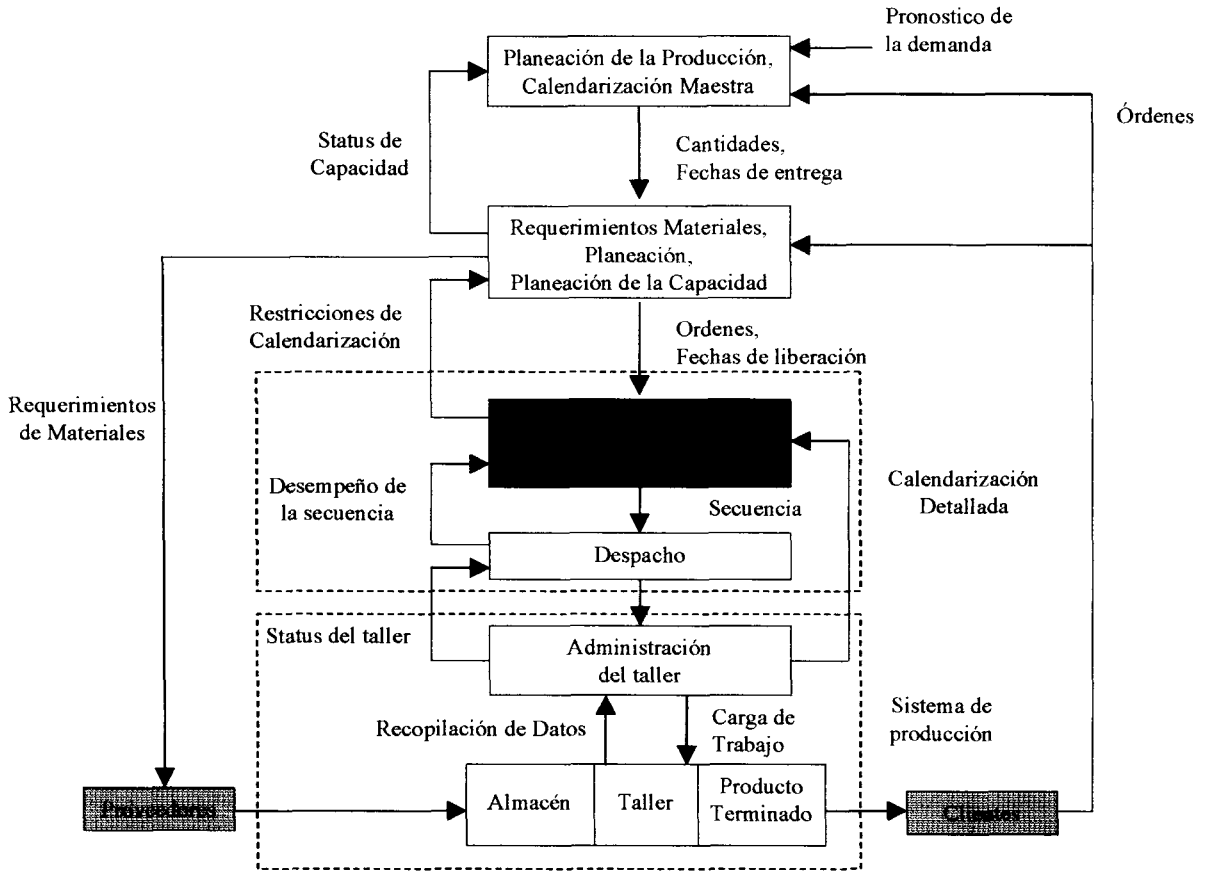


Figura 1-1. Diagrama de Flujo de Información en un Sistema de Manufactura

1.2 CALENDARIZACIÓN

La calendarización de tareas trata sobre la asignación de recursos limitados a ciertas tareas u operaciones a través de cierto período de tiempo. Los recursos y tareas pueden tomar diferentes formas, de acuerdo al dominio del problema que se aborde. El problema de calendarización de tareas existe en casi todos los ambientes y juega un papel importante en el desempeño general de la empresa o sistema de producción.

Los distintos ambientes en un problema de calendarización de tareas son los siguientes:

Una sola máquina. El caso de una sola máquina es el más simple de todos los ambientes posibles y es un caso especial de los demás ambientes.

Máquinas idénticas en paralelo (*Pm*). Esto es cuando hay *m* máquinas idénticas en paralelo. El trabajo *j* requiere una simple operación y puede ser procesada en cualquiera de las *m* máquinas.

Máquinas en paralelo con diferentes velocidades (*Qm*). La velocidad de la máquina *i* es denotada por v_i. El tiempo de procesamiento *pij*, que el trabajo *j* gasta en la maquina *i* es igual a

p_j/v_i (asumiendo que el trabajo *j* recibe todo su procesamiento de la maquina *i*). Este ambiente se utiliza para uniformizar el uso de las máquinas. Si todas las máquinas tienen el mismo tiempo de procesamiento entonces el ambiente sería idéntico al anterior.

Máquinas en paralelo sin relación (*Rm*). Este ambiente es una generalización del anterior. Hay m diferentes máquinas en paralelo. La máquina *i* puede procesar el trabajo *j* a la velocidad v_{ij}. El tiempo de procesamiento p_{ij}, que el trabajo *j* gasta en la maquina *i* es igual a p_j/v_i (asumiendo que el trabajo *j* recibe todo su procesamiento de la maquina *i*). Si las velocidades de las máquinas son independientes de los trabajos, entonces el ambiente es igual al anterior.

Flow Shop (Fm). Hay *m* máquinas en serie. Cada trabajo tiene que ser procesado en cada una de las *m* máquinas. Todos los trabajos tienen que seguir la misma ruta. Después de terminar su proceso en una máquina, ese trabajo se une a la cola de la siguiente máquina. Usualmente todas las colas funcionan con la disciplina, *primero en entrar primero en salir (FIFO First In First Out).*

Flow Shop Flexible (FFc). Un flow shop flexible es una generalización del flow shop y del ambiente de máquinas en paralelo. En lugar de *m* máquinas en serie hay *c* estaciones en serie con el mismo número de máquinas cada estación. Cada trabajo tiene que ser procesado primero en la Estación 1, luego en la Estación 2 y así sucesivamente. Una estación funciona como un banco de máquinas en paralelo; en cada estación un trabajo *j* requiere ser procesado solo en una máquina y cualquier máquina puede hacerlo. La cola en una estación puede o no operar de acuerdo a la disciplina *primero en llegar primero en atenderse (FCFS First Come First Served).*

Job Shop (Jm). En un job shop con *m* máquinas, cada trabajo tiene su propia ruta a seguir. Se hace distinción de entre los job shops en los cuales cada trabajo pasa por cada máquina sólo una vez y en los que cada trabajo puede visitar cada máquina más de una vez, llamando a estos últimos job shop con recirculación, y se debe considerar una variable que considere esta característica.

Job Shop Flexible (FJc). Un job shop flexible es una generalización de un job shop y del ambiente de máquinas en paralelo. En lugar de *m* máquinas, hay *c* centros de trabajo con el mismo número de máquinas en paralelo. Cada trabajo tiene su propia ruta a través del taller; un trabajo *j* requiere procesamiento en un sola máquina de cada centro de trabajo y este puede ser en cualquier máquina del centro de trabajo. Si un trabajo en su ruta a través del taller debe visitar un centro de trabajo más de una vez entonces se debe considerar una variable de recirculación [4].

La mayoría de problemas de calendarización en estos ambientes son problemas de optimización combinatoria y una gran parte de ellos pertenecen a la clase de problemas *NP-hard.* Los problemas *NP-hard* son un subconjunto de la clase *NP* (problemas para los que no se puede tener una solución en tiempo polinomial para todas sus instancias) con la característica de que todos los problemas de ésta clase pueden ser reducidos a él; los problemas para los que se puede encontrar un algoritmo de solución en tiempo polinomial, forman la clase *P*, que es un subconjunto de la clase *NP* [4].

1.2.1 REVISIÓN DE TRABAJOS EN CALENDARIZACIÓN

El problema de la calendarización de tareas ha sido ampliamente estudiado y atacado con diferentes técnicas y heurísticas.

En los últimos años se ha realizado mucha investigación abordando el problema de programación de tareas en ambientes determinísticos. Una significativa cantidad de trabajos se han enfocado a tratar de encontrar heurísticas y algoritmos de tiempo polinomial que resuelvan problemas de programación de tareas. Sin embargo, muchos de los problemas encontrados en esta área se clasifican como problemas *NP-hard*, por lo que no es posible encontrar un algoritmo en tiempo polinomial que los resuelva. Se han propuesto un gran número de enfoques para modelar y solucionar los diferentes problemas de programación de tareas, con diferentes grados de éxito. Entre estos enfoques podemos mencionar la programación matemática, reglas de despacho, sistemas expertos, redes neuronales, algoritmos genéticos, búsqueda tabú, lógica difusa, entre otros [6].

En el artículo [7], sus autores Rogers y White Jr. Desarrollan metodologías de calendarización para reducir el tiempo total, mediante modelos algebraicos, de programación matemática y modelos de red. Se establece un modelo que se formula a partir de estas tres diferentes técnicas matemáticas. Con lo cual se pretende determinar un específico grupo de secuencias de máquinas que probablemente es el óptimo. La única manera que esta metodología puede lograr esto, es calculando primero los tiempos totales de una proporción muy grande de las secuencias factibles. Además un formalismo matemático, no toma en cuenta muchas de las restricciones y objetivos de la calendarización, es por esto que el óptimo matemático solo puede aproximarse a la mejor solución en la práctica.

Los investigadores Uzsoy y Ovacik en su artículo [8], usan reglas de despacho para la calendarización de tareas. En el artículo se propone la calendarización en un complejo de pruebas para semiconductores, el cual se considera como un job shop, se usa la información más relevante, y esta se procesa a través de un algoritmo heurístico de despacho, empleando reglas de despacho estáticas y dinámicas, y se realiza la calendarización. Los mismos autores mencionan que las reglas de despacho usan solo información local, por lo cual tienden a ser miopes, al no tomar en cuenta otras variables que influyen en el sistema de producción.

En el artículo [9], de los autores Wang y Luh., se desarrolla una metodología basada en el método de *relajación Lagrangeana* para descomponer el problema en subproblemas simples, donde los requerimientos de tiempo y bajo inventario en proceso, son modelados como objetivos para minimizar la tardanza. Esto se hace agrupando las máquinas de la misma capacidad de procesamiento como "Máquinas tipo". Las partes pueden tener diferentes fechas de entrega para ser calendarizadas sobre un tiempo discretizado K. la parte i consiste de una serie de operaciones sin referencia con operaciones j. Una operación puede iniciar solamente después que sus operaciones precedentes han sido terminadas. Este es un método de optimización, el cual tiene una función objetivo que pretende la liberación de partes a tiempo, y mantener un bajo inventario, se tienen restricciones de capacidad y de tiempo de proceso. Este método esta enfocado a un

centro de maquinado, es decir a una sola máquina de control numérico, y la secuencia de trabajos es solo para las tareas que tiene que procesar esta máquina de control numérico. El trabajo expuesto en este artículo se puede emplear haciendo una analogía con un sistema de manufactura.

En el documento [10], presentan, Byeon, Wu, y Storer, un planteamiento de ponderación de las tardanzas para lo cual se propone un heurístico usando una técnica de descomposición gráfica. El método descompone en una serie de problemas resolviendo una variante del problema de asignación denominada VAP, para los problemas de calendarización de taller tipo Job Shop. La tesis básica es que su calendarización parcial preserva una perspectiva global de los objetivos del sistema sobre el horizonte de planeación, mientras se retiene la flexibilidad local permitiendo la calendarización para manejo de disturbios y limitaciones locales del sistema. Este método heurístico se basa en una descomposición, por medio de teoría gráfica, también asigna los funcionamientos de una serie de subconjuntos resolviendo una variante del problema de asignación generalizada. El programa de búsqueda repetida para disponer de una gráfica disyuntiva en el comienzo de una planeación horizonte y el uso de despacho dinámico hasta el final del límite. El procedimiento consta de tres pasos, el primero genera un nuevo horario que fija el índice de prioridad actual, computa la tardanza del ponderado total. En el paso dos, del horario generado en el paso uno, se procesa el tiempo de espera para cada trabajo y la tardanza de ajuste de fecha especificada. En el paso tres se recalculan índices de prioridad por una regla definida. Finalmente se propone un procedimiento de descomposición óptima para minimizar el ponderado del tiempo de flujo en problemas de máquina simple. Sin embargo, los autores declaran que el acercamiento, se limita a ese objetivo y no podría generalizarse a otros problemas de planificación.

El proceso es más complicado para este tipo de funciones, porque debe efectuarse mediante plazos. Las funciones se construyen al darle valor arbitrario a una de las variables independientes, y se grafica la función. Con varios valores arbitrarios y repitiendo después la misma operación con la otra variable independiente, se encuentra un número suficiente de puntos y trazos que permiten la graficación.

En el artículo [11], de los investigadores Li y Man, Se propone un Modelo de calendarización y planeación de producción anticipada y tardía (earliness/tardiness production scheduling and planning ETPSP) con consideración del tamaño de lote y balance de capacidad. Para su solución se diseñó una metodología usando Algoritmos Genéticos Multiobjetivos (MOGA).

Los autores Li y Man mencionan que aunque la planeación para recursos para la Manufactura (MRP II) y el Justo a Tiempo (JIT) proveen muchas ventajas en un sistema de manufactura, el alto inventario en proceso, el nerviosismo en los sistemas MRP II, el impacto de los cuellos de botella, la susceptibilidad de desbalanceo y la incertidumbre de los sistemas de manufactura JIT, son los problemas que quedan sin resolver y que tienen que atacarse. Para solucionar estos problemas y lograr los mejores resultados en administración de producción y de inventarios, se integran la MRP II con la filosofía JIT. Esta integración se enfoca en como usar la filosofía JIT para mejorar la metodología de Calendarización y Planeación de producción (PSP) del sistema MRP II por medio de un método eficiente Maestro PSP. El método de calendarización y planeación de producción anticipada y tardía ha sido uno de las áreas de investigación más

importantes y activas. Un MPSP juega un papel importante en los sistemas MRP II, al utilizar la capacidad disponible de los sistemas de manufactura. En general los métodos MPSP consideran como objetivo:

- La Minimización del costo de la producción total o la Maximización de la salida de producción.

Sin embargo como para el desarrollo del JIT, la fecha de vencimiento, es el objetivo más importante. Por lo tanto lo ideal es que el MPSP pueda ser ajustado oportunamente a los cambios en requerimientos. Para ser capaz de soportar diversos requerimientos y cambios en el mercado, los sistemas de manufactura deberían manejar los procesos y tipos de productos con diferentes tamaños de lote.

En cada periodo de producción hay por lo menos un proceso clave, en el cual su capacidad es lo mas limitado y restringido para su producción. El objetivo de un ETPSP es encontrar la óptima planeación y calendarización. En este caso las penalizaciones por prontitud y tardanza son minimizadas y las restricciones en la capacidad del proceso de manufactura son confirmadas.

En el análisis, de los artículos se observa que abordan la calendarización de un sistema de producción con diferentes metodologías, ya sea con programación matemática, reglas de despacho, optimización. Sin embargo cada una de estas metodologías tiene ventajas y desventajas, por ejemplo: en la programación matemática se observa que un formalismo matemático, no toma en cuenta muchas de las restricciones y objetivos de la calendarización, es por esto que el óptimo matemático sólo puede aproximarse a la mejor solución en la práctica. Con respecto a las reglas de despacho, estas usan sólo información local, por lo cual tienden a ser miopes, al no tomar en cuenta otras variables que influyen en el sistema de producción. Y por ultimo los métodos de optimización son particulares del entorno en el que se sitúa el problema de calendarización, ya que la función objetivo que se optimiza, se construye en base a las características particulares de la maquina o sistema que se requiere calendarizar.

2 PLANTEAMIENTO DEL PROBLEMA Y OBJETIVOS

2.1 PLANTEAMIENTO DEL PROBLEMA

La Industria manufacturera en la economía de México reclama la mejora continua de los procesos presentes en esta industria. Una manera de optimizar los procesos de producción es logrando una adecuada calendarización de tareas Una adecuada calendarización puede reducir significativamente los costos de producción y reducir los tiempos de proceso lo que permite cumplir con los compromisos de tiempo de entrega, por tal motivo la búsqueda de métodos de calendarización de tareas a través de un ambiente de producción y la implementación de los mismos en sistemas capaces de obtener una calendarización óptima, son esenciales en los procesos de manufactura.

La calendarización de tareas es un problema que puede ser descrito básicamente como la asignación de recursos limitados a ciertas tareas u operaciones a través de cierto período de tiempo. Los recursos y tareas pueden tomar diferentes formas, de acuerdo al dominio del problema que se aborde. El problema de calendarización de tareas existe en casi todos los ambientes y juega un papel importante en el desempeño general de la empresa o sistema de producción.

En este tesis nos enfocamos al problema de calendarización en un ambiente de producción tipo Job Shop.

El problema clásico de calendarización Job-Shop puede establecerse como sigue:

> *Tenemos m diferentes máquinas y n diferentes trabajos a ser calendarizados. Cada trabajo esta compuesto de un grupo de operaciones y el orden de operaciones en las máquinas deben estar definidos. Cada operación esta caracterizada por la máquina requerida y su tiempo de procesamiento.*

Además las restricciones en trabajos y máquinas son:

- Un trabajo no visita la misma máquina dos veces.
- No hay restricción entre operaciones de diferentes trabajos.
- Las operaciones no pueden ser interrumpidas.
- Cada máquina solo puede procesar un trabajo a la vez.
- No se especifican fechas de liberación ni fechas de vencimiento.

El problema es determinar la secuencia de operaciones en las máquinas con el objetivo de minimizar el *makespan* (el tiempo requerido para completar todos los trabajos).

En los últimos años se ha realizado mucha investigación abordando el problema de calendarización de tareas. Una significativa cantidad de trabajos se han enfocado a tratar de encontrar heurísticas y algoritmos de tiempo polinomial que resuelvan problemas de programación de tareas. Sin embargo, muchos de los problemas encontrados en esta área se clasifican como problemas *NP-hard*, por lo que no es posible encontrar un algoritmo en tiempo polinomial que los resuelva. Se han propuesto un gran número de enfoques para modelar y solucionar los diferentes problemas de calendarización de tareas, con diferentes grados de éxito. Entre estos enfoques podemos mencionar la programación matemática, reglas de despacho, sistemas expertos, redes neuronales, algoritmos genéticos, búsqueda tabú, lógica difusa, entre otros

Los Algoritmos Genéticos tiene la cualidad con respecto a otros enfoques, en la introducción de mecanismos de selección de los candidatos a solución en función de sus respectivas aptitudes y de mecanismos de construcción de nuevos candidatos por cruzamiento y mutación de los individuos existentes. Unos y otros mecanismos proporcionan robustez a la búsqueda, esto es, le añaden eficiencia en favor de la búsqueda y obtención de un óptimo que satisfaga la solución del problema de calendarización

Con respecto a efectos prácticos, y especialmente en el momento de desarrollar la aplicación de calendarización, la robustez de los Algoritmos Genéticos se hace patente en las siguientes propiedades:

- Por ser métodos de búsqueda ciega requieren información poco específica para funcionar; concretamente. A consecuencia de ello son procedimientos de búsqueda extremadamente generales.

- Son métodos de aproximación sucesiva, pudiendo parar en cualquier momento para proporcionar buenas soluciones, no necesariamente optimas.

- Trabajan con una población de posibles soluciones, lo que proporciona capacidad de elección de buenas soluciones en función de criterios de aptitud.

- Presentan una complejidad casi lineal, es decir el número de operaciones que realizan crece proporcionalmente con el tamaño del problema. Esto permite resolver aceptablemente problemas con gran número de variables

En resumen, los Algoritmos Genéticos ofrecen una combinación de simplicidad, y rapidez que les permite competir favorablemente con otros procedimientos generales de búsqueda aplicados a la solución del problema de calendarización.

Esta tesis representa la incursión en el uso de Algoritmos Genéticos, en el proyecto interno de investigación del Dr. Dante Jorge Dorantes González, Director del Departamento de Mecatrónica y Automatización del ITESM-CEM, donde se busca obtener un mejor método de calendarización de tareas en sistemas de producción tipo Job Shop.

El planteamiento del problema queda entonces como sigue:

"Se requiere desarrollar un paquete computacional que resuelva el problema de calendarización de sistemas Job Shop tipo N/M/G/Cmáx mediante una técnica de optimización utilizado un Algoritmo Genético con representación de calendarización por operaciones, cruzamiento por partición de conjuntos SPX y con mutación de intercambio y de intercambio segmentado".

2.2 OBJETIVOS.

- Realizar la investigación de las diferentes técnicas de cruzamiento, mutación y representación de calendarización, en los algoritmos genéticos enfocados a la solución de del problema de calendarización Job Shop.

- Implementar un algoritmo genético clásico con representación de calendarización por operaciones, cruzamiento por partición de conjuntos SPX y con mutación de intercambio y de intercambio segmentado para la calendarización de un sistema de producción tipo Job Shop clásico con la función objetivo de minimizar el tiempo requerido para completar todos los trabajos, y con vias de incursionar en el desarrollo de posteriores algoritmos híbridos.

- Desarrollar un programa computacional en base a la implementación del algoritmo genético para la calendarización de un sistema de producción tipo Job Shop, que presente una interfaz gráfica y amigable para el usuario, tanto en la entrada de datos como al mostrar resultados y parámetros de desempeño.

3 ALGORITMOS GENÉTICOS

Los algoritmos genéticos simulan el proceso de evolución natural y son algoritmos de búsqueda estocástica basada en los principios de la selección natural y la genética poblacional, han sido aplicados con éxito en diferentes dominios de problemas, tales como ingeniería, negocios y ciencias. Un algoritmo genético procesa una población de individuos, normalmente representados por una cadena binaria, encontrando los más aptos por medio de la evaluación de cada uno de estos individuos en una función objetivo. Los individuos más aptos son los que tienen más oportunidad de sobrevivir a la siguiente generación de individuos, de aquí la analogía con la teoría de selección natural. La población inicial es comúnmente generada de forma aleatoria, aunque en algunos casos puede inicializarse con individuos que se piense que representan soluciones viables, lo cual reduciría notablemente el tiempo de búsqueda del óptimo.

3.1 ORÍGENES

En los últimos años, la comunidad científica internacional ha mostrado un creciente interés en una nueva técnica de búsqueda basada en la teoría de la evolución y que se conoce como el algoritmo genético. Esta técnica se basa en los mecanismos de selección que utiliza la naturaleza, de acuerdo a los cuales los individuos más aptos de una población son los que sobreviven, al adaptarse más fácilmente a los cambios que se producen en su entorno. Hoy en día se sabe que estos cambios se efectúan en los genes (unidad básica de codificación de cada uno de los atributos de un ser vivo) de un individuo, y que los atributos más deseables (i.e., los que le permiten a un individuo adaptarse mejor a su entorno) del mismo se transmiten a sus descendientes, cuando éste se reproduce sexualmente [12].

Un investigador de la Universidad de Michigan llamado John Holland estaba consciente de la importancia de la selección natural, y a fines de los 60s desarrolló una técnica que permitió incorporarla en un programa de computadora. Su objetivo era lograr que las computadoras aprendieran por sí mismas. A la técnica que inventó Holland se le llamó originalmente "planes

reproductivos", pero se hizo popular bajo el nombre "algoritmo genético" tras la publicación de su Libro [13] en 1975.

Una definición bastante completa de un algoritmo genético es la propuesta por John Koza [13]:

Es un algoritmo matemático altamente paralelo que transforma un conjunto de objetos matemáticos individuales con respecto al tiempo usando operaciones modeladas de acuerdo al principio Darwiniano de reproducción y supervivencia del más apto, y tras haberse presentado de forma natural una serie de operaciones genéticas de entre las que destaca la recombinación sexual. Cada uno de estos objetos matemáticos suele ser una cadena de caracteres (letras o números) de longitud fija que se ajusta al modelo de las cadenas de cromosomas, y se les asocia con una cierta función matemática que refleja su aptitud.

3.2 USO DE ALGORITMOS GENÉTICOS

La aplicación más común de los algoritmos genéticos ha sido la solución de problemas de optimización, en donde han mostrado ser muy eficientes y confiables. Sin embargo, no todos los problemas pudieran ser apropiados para la técnica, y se recomienda en general tomar en cuenta las siguientes características del mismo antes de intentar usarla:

- Su espacio de búsqueda y sus posibles soluciones deben estar delimitados dentro de un cierto rango.

- Debe poderse definir una función de aptitud que nos indique qué tan buena o mala es una cierta respuesta.

El primer punto es muy importante, y lo más recomendable es intentar resolver problemas que tengan espacios de búsqueda discretos -aunque éstos sean muy grandes-. Sin embargo, también podrá intentarse usar la técnica con espacios de búsqueda continuos, pero preferentemente cuando exista un rango de soluciones relativamente pequeño.

La **función de aptitud** no es más que la función objetivo de nuestro problema de optimización. El algoritmo genético únicamente maximiza, pero la minimización puede realizarse fácilmente utilizando el recíproco de la función maximizante (debe cuidarse, por supuesto, que el recíproco de la función no genere una división por cero). Una característica que debe tener esta función es que debe ser capaz de "castigar" a las malas soluciones, y de "premiar" a las buenas, de forma que sean estas últimas las que se propaguen con mayor rapidez.

La codificación más común de las respuestas es a través de cadenas binarias, aunque se han utilizado también números reales y letras. El primero de estos esquemas ha gozado de mucha popularidad debido a que es el que propuso originalmente Holland, y además porque resulta muy sencillo de implementar.

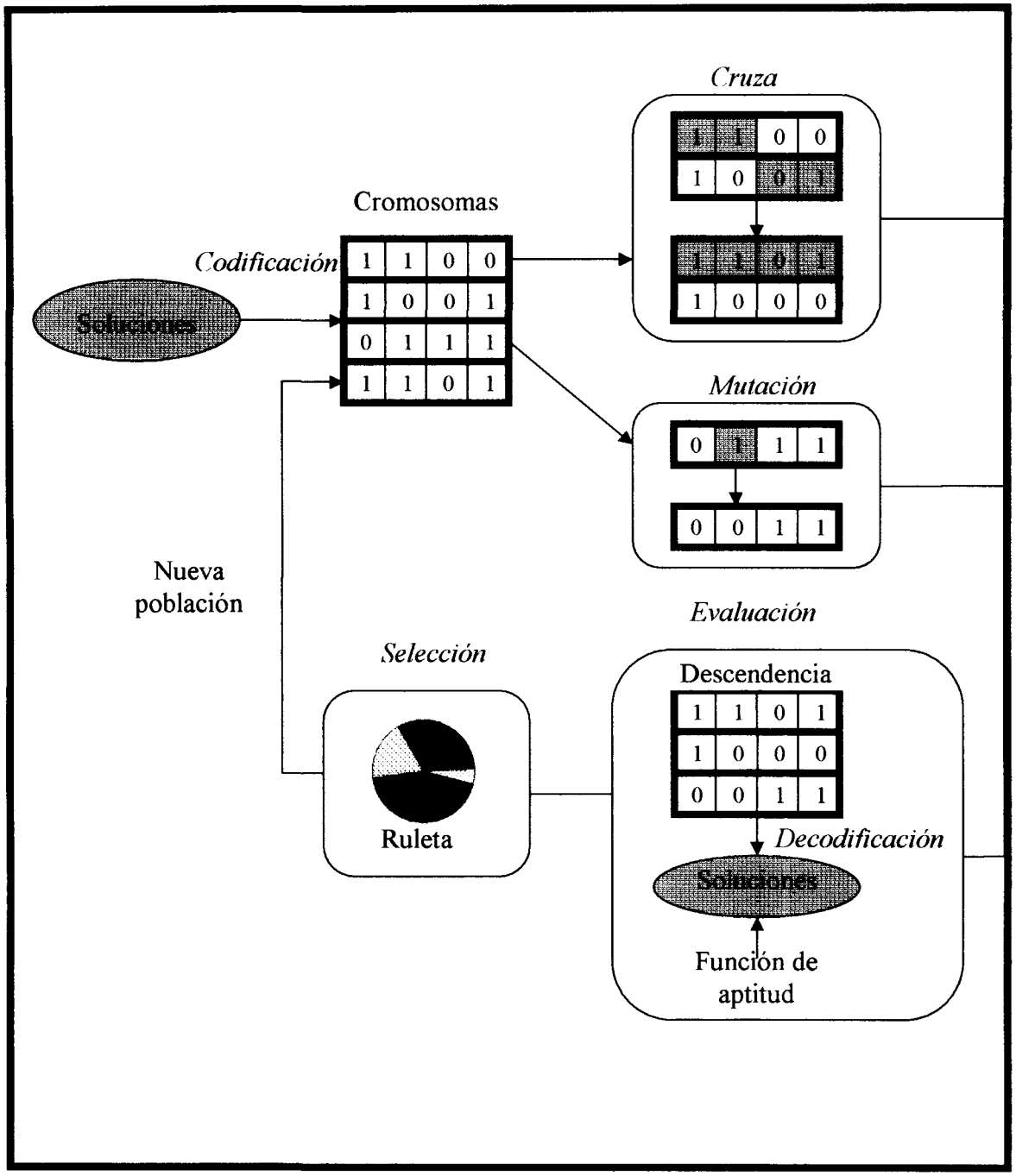


Figura 3-1. Estructura general de los Algoritmos Genéticos

3.3 FUNCIONAMIENTO DE UN ALGORITMO GENÉTICO SIMPLE

La operación de un algoritmo genético se describe a continuación

```
generar población inicial, G(0);
evaluar G(0);
t:=0;
repetir
        t:=t+1;
        generar G(t) usando G(t-1);
        evaluar G(t);
hasta encontrar una solución;
```

En los Algoritmos Genéticos hay solo dos tipos de operaciones

- Operaciones Genéticas: Cruza y Mutación
- Operaciones de Evolución: Selección [14].

Primero, se genera aleatoriamente la población inicial, que estará constituida por un conjunto de **cromosomas**, o cadenas de caracteres que representan las soluciones posibles del problema. A cada uno de los cromosomas de esta población se le aplicará la función de aptitud a fin de saber qué tan buena es la solución que está codificando.

Sabiendo la aptitud de cada cromosoma, se procede a la **selección** de los que se cruzarán en la siguiente generación (presumiblemente, se escogerá a los "mejores"). Dos son los métodos de selección más comunes:

La Ruleta: Es el usado por Goldberg en su libro [15]. Este método es muy simple, y consiste en crear una ruleta en la que cada cromosoma tiene asignada una fracción proporcional a su aptitud. Sin que nos refiramos a una función de aptitud en particular, supongamos que se tiene una población de cinco cromosomas cuyas aptitudes están dadas por los valores mostrados en la Tabla 3-1.

Con los porcentajes mostrados en la cuarta columna de la Tabla 3-1 podemos elaborar la ruleta de la Figura 3-2. Esta ruleta se gira cinco veces para determinar qué individuos se seleccionarán. Debido a que a los individuos más aptos se les asignó un área mayor de la ruleta, se espera que sean seleccionados más veces que los menos aptos.

El torneo: La idea de este método es muy simple. Se baraja la población y después se hace competir a los cromosomas que la integran en grupos de tamaño predefinido (normalmente compiten en parejas) en un torneo del que resultarán ganadores aquéllos que tengan valores de aptitud más altos. Si se efectúa un torneo binario (i.e., competencia por parejas), entonces la población se debe barajar dos veces. Nótese que esta técnica garantiza la obtención de múltiples copias del mejor individuo entre los progenitores de la siguiente generación (si se efectúa un torneo binario, el mejor individuo será seleccionado dos veces).

Tabla 3-1. Valores de ejemplo para ilustrar la selección mediante ruleta

Cromosoma No.	Cadena	Aptitud	% del Total
1	11010110	254	24.5
2	10100111	47	4.5
3	00110110	457	44.1
4	01110010	194	18.7
5	11110010	85	8.2
Total		1037	100

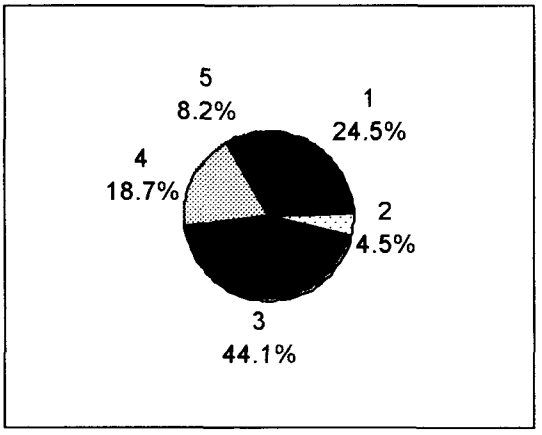


Figura 3-2. Ruleta que representa los valores de aptitud de la Tabla 3-1

Una vez realizada la selección, se procede a la **reproducción sexual** o **cruza** de los individuos seleccionados. En esta etapa, los sobrevivientes intercambiarán material cromosómico y sus descendientes formarán la población de la siguiente generación. Las dos formas más comunes de reproducción sexual son: la de un punto único de cruza y la de dos puntos de cruza.

Cuando se usa **un solo punto de cruza**, éste se escoge de forma aleatoria sobre la longitud de la cadena que representa el cromosoma, y a partir de él se realiza el intercambio de material de los dos individuos, tal y como se muestra en la Figura 3-3.

Cuando se usan **2 puntos de cruza**, se procede de manera similar, pero en este caso el intercambio se realiza en la forma mostrada en la Figura 3-4.

Normalmente la cruza se maneja dentro de la implementación del algoritmo genético como un porcentaje que indica con qué frecuencia se efectuará. Esto significa que no todas las parejas de cromosomas se cruzarán, sino que habrá algunas que pasarán intactas a la siguiente generación. De hecho existe una técnica desarrollada hace algunos años en la que el individuo más apto a lo largo de las distintas generaciones no se cruza con nadie, y se mantiene intacto hasta que surge

otro individuo mejor que él, que lo desplazará. Dicha técnica es llamada **elitismo**, y se desarrollo en Alemania.

Uso de un solo punto de cruza entre 2 individuos. Cada pareja de cromosomas da origen a 2 descendientes para la siguiente generación. El punto de cruza puede ser cualquiera de los 2 extremos de la cadena, en cuyo caso no se realiza la cruza

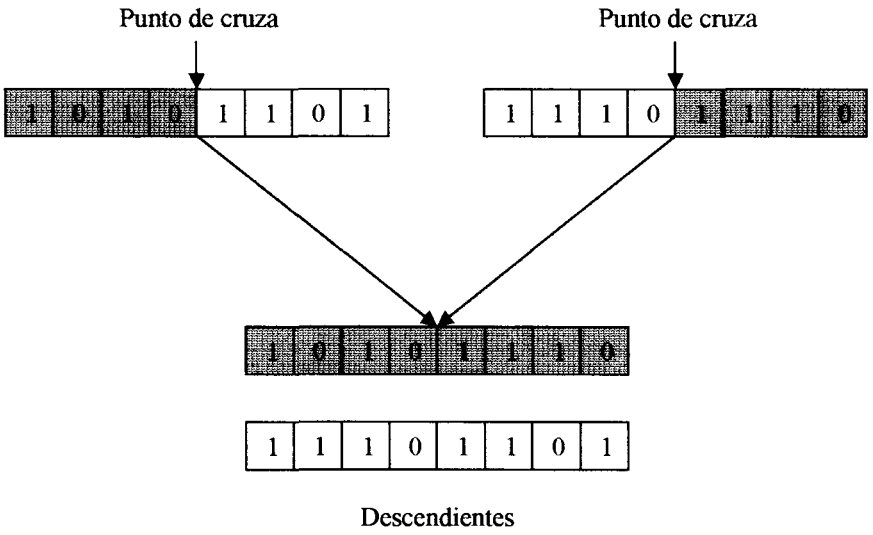


Figura 3-3. Cruzamiento

En el uso de 2 puntos de cruza entre 2 individuos se mantienen los genes de los extremos, y se intercambian los del centro. También aquí existe la posibilidad de que uno o ambos puntos de cruza se encuentren en los extremos de la cadena, en cuyo caso se hará una cruza usando un solo punto, o ninguna cruza, según corresponda.

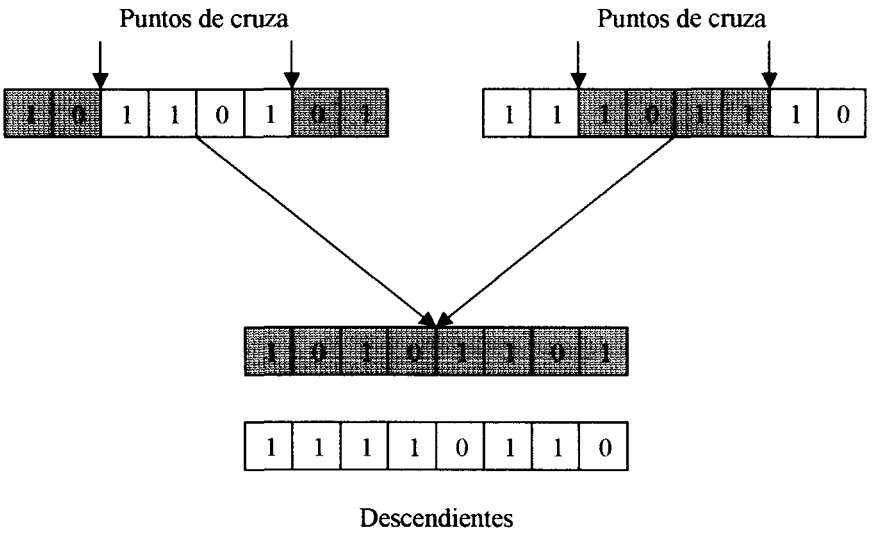


Figura 3-4. Uso de 2 puntos de cruza entre 2 individuos

Además de la selección y la cruza, existe otro operador llamado **mutación**, el cual realiza un cambio a uno de los genes de un cromosoma elegido aleatoriamente.

Cuando se usa una representación binaria, el gen seleccionado se sustituye por su complemento (un cero cambia en uno y viceversa). Este operador permite la introducción de nuevo material cromosómico en la población, tal y como sucede con sus equivalentes biológicos.

Al igual que la cruza, la mutación se maneja como un porcentaje que indica con qué frecuencia se efectuará, aunque se distingue de la primera por ocurrir mucho más esporádicamente (el porcentaje de cruza normalmente es de más del 60%, mientras que el de mutación normalmente nunca supera el 5%).

Si supiéramos la respuesta a la que debemos llegar de antemano, entonces detener el algoritmo genético sería algo trivial. Sin embargo, eso casi nunca es posible, por lo que normalmente se usan 2 criterios principales de detención:

- Correr el algoritmo genético durante un número máximo de generaciones.
- Detener el algoritmo genético cuando la población se haya estabilizado (i.e., cuando todos o la mayoría de los individuos tengan la misma aptitud).

En resumen, se puede explicar el funcionamiento de los tres operadores del Algoritmo Genético Simple de la siguiente manera:

1. ***Selección***: La selección se encarga de incrementar la presencia de los mejores esquemas y reducir la presencia de los retrasados. No obstante, la selección no introduce nuevos esquemas.

2. ***Cruce***: El cruce permite el intercambio estructurado de información útil (esquemas cortos, de bajo orden y aventajados) entre individuos. Por lo tanto, el cruce es un operador esencial para el eficaz funcionamiento de un AG.

3. ***Mutación***: La mutación introduce variedad en el juego de esquemas de la población y proporciona un mecanismo de seguridad frente a posibles pérdidas de información valiosa. Visto de esta manera, la mutación es un operador secundario; de ahí que se aplique con bastante menor frecuencia que el cruce.

3.4 VENTAJAS Y DESVENTAJAS DE LOS ALGORITMOS GENÉTICOS CON RESPECTO A OTRAS TÉCNICAS DE BÚSQUEDA

- No necesitan conocimientos específicos sobre el problema que intentan resolver.
- Operan de forma simultánea con varias soluciones, en vez de trabajar de forma secuencial como las técnicas tradicionales.

- Cuando se usan para problemas de optimización -maximizar una función objetivo- resultan menos afectados por los máximos locales (falsas soluciones) que las técnicas tradicionales.
- Usan operadores probabilísticos, en vez de los típicos operadores determinísticos de las otras técnicas.
- Pueden tardar mucho en converger, o no converger en absoluto, dependiendo en cierta medida de los parámetros que se utilicen -tamaño de la población, número de generaciones, etc.-.
- Pueden converger prematuramente debido a una serie de problemas de diversa índole.

3.5 AMBIENTES DE PROGRAMACIÓN

En la actualidad existe un gran número de ambientes de programación disponibles en el mercado para experimentar con los algoritmos genéticos. De acuerdo a la taxonomía sugerida en [15], pueden distinguirse tres clases de ambientes de programación:

1) Sistemas Orientados a las aplicaciones: Son esencialmente "cajas negras" para el usuario, pues ocultan todos los detalles de implementación. Sus usuarios -normalmente neófitos en el área- los utilizan para un cierto rango de aplicaciones diversas, pero no se interesan en conocer la forma en qué estos operan. Ejemplos de este tipo de sistemas son: **Evolver** (Axcelis, Inc.) y **XpertRule GenAsys** (Attar Software).

2) Sistemas Orientados a los algoritmos: Soportan algoritmos genéticos específicos, y suelen subdividirse en:
Sistemas de uso específico: Contienen un solo algoritmo genético, y se dirigen a una aplicación en particular. Algunos ejemplos son: **Escapade** (Frank Hoffmeister), **GAGA** (Jon Crowcroft) y **Genesis** (John Grefenstette).
Bibliotecas: Agrupan varios tipos de algoritmos genéticos, y diversos operadores (e.g. distintas formas de realizar la cruza y la selección). **Evolution Machine** (H. M. Voigt y J. Born) y **OOGA** (Lawrence Davis) constituyen 2 ejemplos representativos de este grupo. En estos sistemas se proporciona el código fuente para que el usuario -normalmente un programador- pueda incluir el algoritmo genético en sus propias aplicaciones.

3) Cajas de Herramientas: Proporcionan muchas herramientas de programación, algoritmos y operadores genéticos que pueden aplicarse en una enorme gama de problemas. Normalmente se subdividen en:
Sistemas Educativos: Ayudan a los usuarios novatos a introducirse de forma amigable a los conceptos de los algoritmos genéticos. **GA Workbench** (Mark Hughes) es un buen ejemplo de este tipo de ambiente.
Sistemas de Propósito General: Proporcionan un conjunto de herramientas para programar cualquier algoritmo genético y desarrollar cualquier aplicación. Tal vez el sistema más conocido de este tipo es **Splicer** (NASA).

3.6 EJEMPLO DEL FUNCIONAMIENTO DE UN ALGORITMO GENÉTICO

Supóngase que se pretende encontrar el punto del intervalo [0, 0.875] donde la función:

$$f(x)=\left[1-\left(\frac{11}{2}x-\frac{7}{2}\right)^2\right]\left[\cos\left(\frac{11}{2}x-\frac{7}{2}\right)+1\right]+2$$

tiene su máximo, usando para ello un algoritmo genético simple con probabilidad de cruza p_c = 0.70 y probabilidad de mutación p_m = 0.0666... = 1/15. En la Figura 3-5 se muestra la gráfica de la función.

El primer paso del algoritmo genético simple es codificar el dominio de la función, que en nuestro caso es el intervalo [0, 0.875]. Hay que tomar en cuenta que siempre estos códigos serán almacenados en la memoria de la computadora donde se ejecutará el algoritmo genético. Se pueden pensar muchas maneras de codificar los elementos del dominio. Es posible, por ejemplo, colocar cada uno de los primeros *k* dígitos decimales del número (recuérdese que el dominio está constituido por números entre cero y 0.875) fijando el valor de *k* arbitrariamente; se puede pensar también en guardar estos números simplemente de la forma en que son representados los números reales por los compiladores de lenguajes de alto nivel, es decir, en el formato de punto flotante. No importa cuál mecanismo de codificación se utilice siempre se tendrá, por supuesto, un número finito de códigos disponibles para representar un conjunto infinito y no numerable de elementos del dominio, así que hay siempre una infinidad de elementos del dominio que no son representados. Con el fin de simplificar el ejemplo se codificarán los elementos del dominio en binario, usando 3 bits en la representación. Evidentemente se tienen entonces sólo 2^3=8 números representados (la codificación se muestra en la Tabla 3-2), y se puede pensar en que cada número binario de tres bits es, en realidad, la expansión en notación binaria pesada de la parte decimal de un número. Así por ejemplo, 101 representa $1\times2^{-1}+0\times2^{-2}+1\times2^{-3}$ = ½+1/8 = 0.625.

El máximo de la función en el intervalo se obtiene en:

$$x_m=\frac{7}{11}=0.6363$$

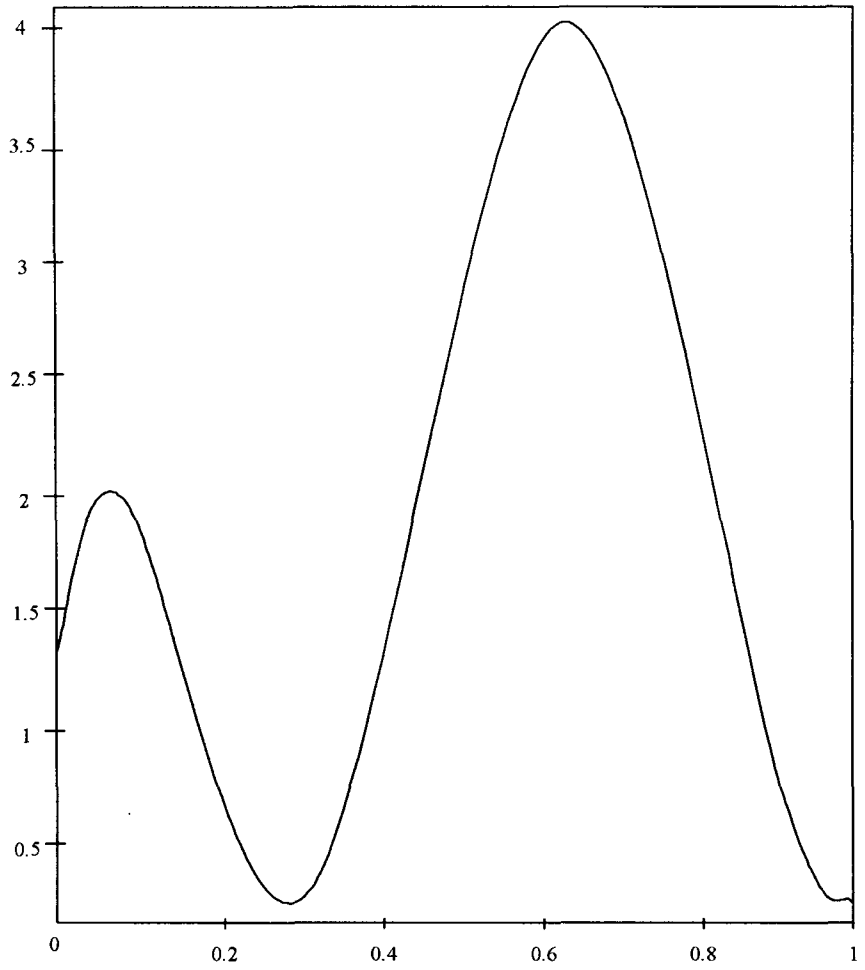


Figura 3-5. Gráfica de la función del ejemplo

Tabla 3-2. Codificación y evaluación de los elementos del dominio de la función f(x) del ejemplo.

No.	Código	E. Dominio	*f(x)*
0	000	0.000	1.285
1	001	0.125	1.629
2	010	0.250	0.335
3	011	0.375	0.792
4	100	0.500	2.000
5	101	0.625	3.990
6	110	0.750	3.104
7	111	0.875	1.093

El valor de la función en el máximo es justamente 4. Evidentemente *Xm* no está representado en la codificación del dominio. El elemento más cercano representado es 0.625, así que se esperaría que el algoritmo genético encontrara la solución aproximada 101 que codifica al 0.625.

El siguiente paso es generar de manera aleatoria un conjunto de posibles soluciones. Nuevamente hay muchas formas de hacer esto: generar códigos válidos uniformemente distribuidos en el conjunto de posibles códigos, o generarlos con una distribución normal o de Poisson, etc. Para que el proceso sea lo más imparcial posible, se ha elegido aquí una distribución uniforme (aunque la población es tan pequeña que esto no es notorio). Se ha seleccionado un tamaño de población de 5. En la Tabla 3-3 se muestra la población inicial.

Se procede a calificar a cada uno de los elementos de la población actual (en este caso la población inicial). En la Tabla 3-3 se muestra la calificación en la tercera columna (de izquierda a derecha). En el caso que nos ocupa se pretende maximizar una función. Ésta siempre toma valores no negativos en el intervalo que se codificó, así que un buen candidato para función de calificación, adaptación o *fitness* es, justamente, el valor mismo de la función. Es decir, la calificación de cada individuo será el valor de la función en el elemento del dominio que es codificado por dicho individuo.

En este momento se debe traducir esta calificación en una cierta medida de la oportunidad para reproducirse, es decir, de ser seleccionado para pasar a la siguiente generación o, al menos, pasar algunos descendientes. En la cuarta columna de la Tabla 3-3 se muestra la probabilidad de selección de cada individuo de la población inicial. Para calcular la probabilidad de selección del i-ésimo individuo de la población basta dividir la calificación de *i* entre la suma total de las calificaciones de toda la población. Esta operación lleva implícita el mecanismo de selección uniforme o por ruleta que ya se mencionó. Pero no es necesario calcular esta probabilidad de selección.

Tabla 3-3. Población inicial P_0 generada uniformemente para el ejemplo

Índice	Individuos	Calificación	Prob. de selección	Calif. acumulada
0	000	1.285	0.200	1.285
1	001	1.629	0.253	2.914
2	001	1.629	0.253	4.543
3	111	1.093	0.170	5.636
4	011	0.729	0.123	6.428
	Suma	*6.428*	*0.999*	
	Promedio	*1.286*		

Para efectuar la selección se hace lo siguiente:

1. Se genera un número aleatorio *r* de la distribución uniforme (0 ,1).
2. Se multiplica *r* por la suma de las calificaciones de la población (S), obteniéndose $c = rS$
3. Se establece la calificación acumulada (C_a) y el índice actual en cero: $C_a = 0$, i = 0
4. A la calificación acumulada se le suma la calificación del i-ésimo individuo: $C_a = C_a + calif(i)$
5. Si $C_a > c$ entonces el i-ésimo individuo es seleccionado
6. Si no, entonces se incrementa *i* y se regresa al paso 4

En nuestro caso, en la primera selección efectuada:

r = 0.213
c = 0.213 × 6.428 = 1.369

Dado que 1.369 está entre 1.285 y 2.914, calificaciones acumuladas para el individuo de índice 0 y el de índice 1, respectivamente, entonces el individuo seleccionado es precisamente el de índice 1.

En la segunda selección:

r = 0.314
c = 0.314 × 6.428 = 2.018

que nuevamente cae entre las calificaciones acumuladas para el individuo de índice 0 y el de índice 1, así que nuevamente es seleccionado el individuo de índice 1. Cabe enfatizar que la selección se hace *con reemplazo*, es decir, un mismo individuo puede ser seleccionado varias veces. Cada vez que se selecciona un individuo *no se quita de la población* de la que se está seleccionando, permanece en ella para que pueda ser elegido de nuevo.

Ahora que ya se poseen dos individuos seleccionados (que en realidad son el mismo), se procede a determinar si deben o no ser cruzados. Para ello:

1. Se genera un número aleatorio $d \in [0, 1]$.
2. Si $d < p_c$ entonces la pareja seleccionada (A, B) debe cruzarse.
3. Sea *l* la longitud de la cadena que constituye el código genético de cada individuo. Se genera un número aleatorio (x) entero entre 1 y *l*-1. Se indexan los bits de los individuos seleccionados de izquierda a derecha: 1, 2,..., *l*.
4. Se pegan los bits de índices 1, ..., x-1 de A y los de índices x, ... *l* de B para formar un individuo híbrido llamado C y los bits 1, ..., x-1 de B se pegan con los de índices x, ..., *l* de A para formar un individuo híbrido D.
5. C y D son los nuevos individuos.
6. Si $d \geq p_c$ entonces la pareja seleccionada (A, B) no se debe cruzar. A y B son los *nuevos individuos.*

En el ejemplo que nos ocupa, dado que A y B son ambos iguales a 001, entonces no importa si se cruzan o no, los nuevos individuos serán nuevamente 001 ambos.

Pero en una repetición posterior del ciclo resultaron seleccionados el individuo 1 (001) y el individuo 3 (111), y en este caso al llevar a cabo el procedimiento descrito arriba:

$$d = 0.47$$

dado que $d < 0.70 = p_c$ entonces 001 y 111 deben cruzarse $x = 1$, así que A = 001 queda dividido en dos subcadenas: $A_1 = 0$, $A_2 = 01$ y B = 111 queda dividido en: $B_1 = 1$, $B_2 = 11$
Pegando A_1 y B_2 queda C = 011 y pegando B_1 y A_2 queda D = 101 C y D son los nuevos individuos

Cada vez que se tienen dos nuevos individuos se procede a ejecutar el procedimiento de mutación:

1. Se indexan nuevamente los bits de cada individuo de izquierda a derecha: 1,..., l y para cada uno de los bits:
2. Se elige un número aleatorio $q \in [0, 1]$
3. Si $q < p_m$ entonces el bit en turno se invierte (si valía 0 se transforma en 1 y viceversa)
4. Si $q \geq p_m$ entonces el bit en turno permanece sin cambio.

Tabla 3-4. Resultados del proceso

P_0	Calif.	P_1	Calif.	P_2	Calif.	P_3	Calif.
000	1.285	001	1.629	101	3.99	101	3.99
001	1.629	001	1.629	000	1.285	101	3.99
001	1.629	100	2	001	1.629	101	3.99
111	1.093	011	0.792	001	1.629	100	2
011	0.792	001	1.629	110	3.104	101	3.99
	1.285		*1.536*		*2.327*		*3.592*

La Tabla 3-4 muestra los Resultados del proceso genético aplicado a una cierta población inicial P_0 para resolver el problema de ejemplo. Los números en cursivas son el promedio de la columna.

En nuestro caso $p_m = 1/15$, es decir, en promedio sólo habrá un bit cambiado en cada generación. De hecho, en nuestro ejemplo el bit que resultó cambiado al obtener la primera generación fue el último bit de D = 101 que se acaba de obtener por cruzamiento, como resultado de esto se obtiene el 100 que aparece como individuo de índice 2 en la generación 1 mostrada en la Tabla 3-4. En esta tabla se muestra el proceso completo, generación a generación, y puede observarse cómo la calificación promedio de la población (números en cursivas al final de cada columna de calificación), se va incrementando conforme se avanza en el tiempo. Si al principio se tenía una

población dispersa con una calificación promedio de 1.285, al final se obtiene una población donde casi todos los individuos son 101 (el código más cercano al óptimo real) y la calificación promedio es de 3.592. De hecho, la población tiende a poseer sólo el individuo 101 (es el único punto óptimo) y solamente debido al operador de mutación hay algún individuo diferente (100 en el caso de P_3). Nótese también que, en las primeras etapas del proceso se tiende a reproducir con más frecuencia el individuo 001, donde la función tiene un máximo local. Esta situación se mantiene hasta que se encuentra el individuo 101, donde la función alcanza (aproximadamente) su máximo global.

3.7 DISCUSIÓN

En definitiva cabe concluir que, desde un punto de vista teórico, la principal cualidad de los Algoritmos Genéticos con respecto a otras técnicas de búsqueda es la introducción de mecanismos de selección de los candidatos a solución en función de sus respectivas aptitudes y de mecanismos de construcción de nuevos candidatos por cruzamiento y mutación de los individuos existentes. Unos y otros mecanismos proporcionan robustez a la búsqueda, esto es, le añaden eficiencia en favor de la búsqueda y obtención de la solución que satisfaga el problema de calendarización

Con respecto a efectos prácticos, y especialmente en el momento de desarrollar la aplicación de calendarización, la robustez de los Algoritmos Genéticos se hace patente en estas propiedades:

- Por ser métodos de búsqueda ciega requieren información poco específica para funcionar; concretamente. A consecuencia de ello son procedimientos de búsqueda extremadamente generales.

- Son métodos de aproximación sucesiva, pudiendo parar en cualquier momento para proporcionar buenas soluciones, no necesariamente optimas.

- Trabajan con una población de posibles soluciones, lo que proporciona capacidad de elección de buenas soluciones en función de criterios de aptitud.

- Presentan una complejidad casi lineal, es decir el número de operaciones que realizan crece proporcionalmente con el tamaño del problema. Esto permite resolver aceptablemente problemas con gran número de variables

En resumen, los Algoritmos Genéticos ofrecen una combinación de simplicidad, y rapidez que les permite competir favorablemente con otros procedimientos generales de búsqueda.

4 CALENDARIZACIÓN JOB-SHOP

Los problemas de calendarización de máquinas están presentes en diversas áreas, tales como los sistemas flexibles de manufactura, la planeación de producción, logística, comunicaciones entre otros. Una característica común de estos problemas es que no existe aún un algoritmo de solución eficiente para resolver óptimamente el problema en tiempo polinomial.

El problema de calendarización Job-Shop puede describirse informalmente como sigue: *Hay un conjunto de trabajos y un conjunto de máquinas. Cada trabajo consiste de una cadena de operaciones, cada una de estas operaciones necesita ser procesada durante un determinado periodo de tiempo sin interrupciones, en una maquina dada. Cada máquina solo puede procesar una operación a la vez.*

La calendarización de un sistema tipo Job-Shop es de los problemas más difíciles de optimización combinatoria. Debido a la inherente dificultad para lograr una calendarización optima, los procedimientos heurísticos son una alternativa atractiva, sin embargo en años recientes ha surgido el interés por resolver problemas Job-Shop por medio de búsqueda local, esto ha ocurrido por el desarrollo de métodos probabilísticos de búsqueda local, tales como: recocido simulado, búsqueda tabú y algoritmos genéticos.

4.1 MODELO CLÁSICO JOB-SHOP

El problema clásico de calendarización Job-Shop puede establecerse como sigue:

> *Tenemos m diferentes máquinas y n diferentes trabajos a ser calendarizados. Cada trabajo esta compuesto de un grupo de operaciones y el orden de operaciones en las máquinas deben estar definidos. Cada operación esta caracterizada por la máquina requerida y su tiempo de procesamiento.*

A continuación se enumeran las restricciones en trabajos y máquinas:

- Un trabajo no visita la misma máquina dos veces.
- No hay restricción de operaciones entre operaciones de diferentes trabajos.
- Las operaciones no pueden ser interrumpidas.
- Cada máquina solo puede procesar un trabajo a la vez.
- No se especifican fechas de liberación ni fechas de vencimiento.

El problema es determinar la secuencia de operaciones en las máquinas con el objetivo de minimizar el *makespan* (el tiempo requerido para completar todos los trabajos).

Un ejemplo de tres trabajos tres máquinas se presenta en la siguiente tabla

Tabla 4-1. Problema de tres máquinas, tres trabajos

	Operaciones (máquina, tiempo proc.)		
Trabajos	1	2	3
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

Se usan diagramas de Gantt para ilustrar las secuencias factibles. Hay dos formas de representar las secuencias factibles en los diagramas de Gantt, estas son: En base a las maquinas y en base a trabajos. Se utiliza el índice ***jom*** para nombrar cada operación, donde ***j*** indica el trabajo, ***o*** indica la operación y ***m*** indica la maquina a realizar esa operación

Las dos distintas representaciones de las soluciones factibles por diagramas de Gantt se muestran a continuación, expresando las operaciones por el índice *trabajo, operación, máquina.*

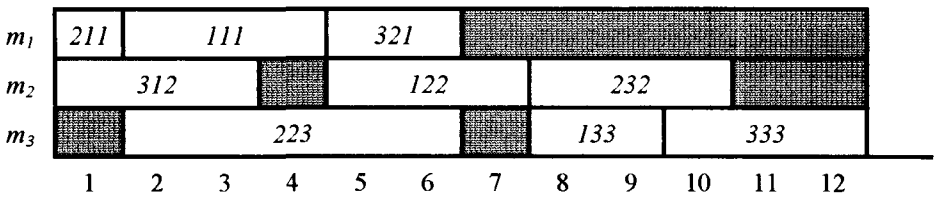


Figura 4-1. Diagrama de Gantt por maquinas

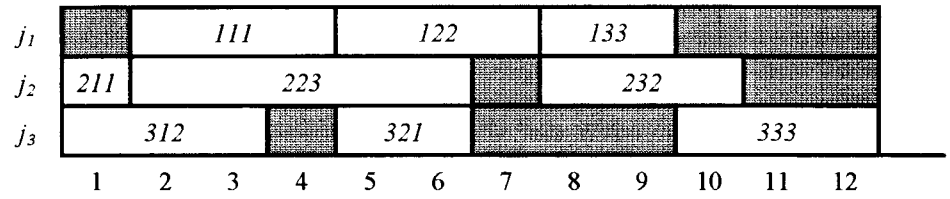


Figura 4-2. Diagrama de Gantt por trabajos

4.2 MODELO DE PROGRAMACIÓN ENTERA

Esta formulación confía en un indicador para especificar la secuencia de operación.

Dos tipos de restricciones necesitan ser considerados para el problema de calendarización Job-Shop.

- Restricción de precedencia de operación para un trabajo dado
- Restricción de no-traslape de operaciones en una maquina dada

Se va a denotar c_{jk} como el tiempo de terminación del trabajo j en la máquina k y a t_{jk} como el tiempo de procesamiento del trabajo j en la máquina k. Primero se va a considerar la restricción de precedencia de operaciones para una secuencia dada. Para un trabajo i, si el procesamiento en una máquina h precede al de una máquina k, se necesita seguir la siguiente restricción:

$$c_{ik} - t_{ik} \geq c_{ih}$$

Si en cambio, el procesamiento en la máquina k es primero, entonces necesitamos seguir la siguiente restricción

$$c_{ih} - t_{ih} \geq c_{ik}$$

Estas restricciones son llamadas restricciones disyuntivas. Es útil definir un coeficiente indicador a_{ihk} como sigue:

$$a_{ihk} \begin{cases} 1, & \text{el procesamiento en la máquina h precede al de la máquina } k \text{ para el job } i \\ 0, & \text{el procesamiento en la máquina k precede al de la máquina } h \text{ para el job } i \end{cases}$$

Entonces se puede reescribir las restricciones anteriores como sigue:

$$c_{ik} - t_{ik} + M(1 - a_{ihk}) \geq c_{ih}, \qquad i = 1, 2, \ldots, n, \qquad h, k = 1, 2, \ldots, \text{m}$$

Donde M es un número positivo grande. La desigualdad anterior representa claramente la restricción de precedencia. Si el procesamiento en la maquina h se debe hacer primero, $M(1- a_{ihk})$ es cero, quedando la desigualdad verdadera. De otra manera si el procesamiento en la máquina k es primero $M(1- a_{ihk})$ se hace un número muy grande haciendo la desigualdad verdadera.

Ahora se considerara la restricción de no-traslape de operaciones para una máquina dada. Para dos trabajos *i* y *j*, donde ambos necesitan ser procesados en la máquina *k*, si el trabajo *i* se procesa primero que el trabajo *j*, necesitamos la siguiente restricción

$$c_{jk} - c_{ik} \geq t_{jk}$$

Si por otro lado, el trabajo *j* se procesa primero que el trabajo *i*, necesitamos la siguiente restricción

$$c_{ik} - c_{jk} \geq t_{ik}$$

Entonces definiremos una variable indicador x_{ijk} como sigue:

$$x_{ijk} \begin{cases} 1, & \text{si el trabajo } i \text{ precede al trabajo } j \text{ en la máquina } k \\ 0, & \text{si el trabajo } j \text{ precede al trabajo } i \text{ en la máquina } k \end{cases}$$

Entonces se puede reescribir las restricciones anteriores como sigue:

$$c_{jk} - c_{ik} + M(1- x_{ijk}) \geq t_{jk}, \qquad i,j = 1, 2, \ldots, n, \qquad k = 1, 2, \ldots, \text{m}$$

El problema de calendarización Job-Shop con el objetivo de minimizar el makespan puede ser formulado como sigue:

$$\min \quad \max_{1\leq k\leq m}\left\{\max_{1\leq i\leq n}\{c_{ik}\}\right\} \tag{1}$$

sujeto a

$$c_{ik} - t_{ik} + M(1- a_{ihk}) \geq c_{ih}, \qquad i = 1, 2, \ldots, n, \qquad h, k = 1, 2, \ldots, \text{m} \tag{2}$$

$$c_{jk} - c_{ik} + M(1- x_{ijk}) \geq t_{jk}, \qquad i, j = 1, 2, \ldots, n, \qquad k = 1, 2, \ldots, \text{m} \tag{3}$$

$$c_{ik} \geq 0, \qquad i = 1, 2, \ldots, n, \qquad k = 1, 2, \ldots, \text{m} \tag{4}$$

$$x_{ijk} = 0 \text{ ó } 1, \qquad i, j = 1, 2, \ldots, n, \qquad k = 1, 2, \ldots, \text{m} \tag{5}$$

La restricción (2) asegura que la secuencia de procesamiento de las operaciones de cada trabajo corresponda a la orden determinada.

La restricción (3) asegura que cada máquina pueda procesar solo un trabajo a la vez.

Para el ejemplo dado por la tabla 3-1

Tabla 3-1

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

Tenemos las restricciones siguientes:

$$C_{12} - t_{12} + M(1\text{-}a_{112}) \geq C_{11}$$
$$C_{13} - t_{13} + M(1\text{-}a_{123}) \geq C_{12}$$
$$C_{11} - t_{11} + M(1\text{-}a_{131}) \geq C_{13}$$
$$C_{22} - t_{22} + M(1\text{-}a_{212}) \geq C_{21}$$
$$C_{23} - t_{23} + M(1\text{-}a_{223}) \geq C_{22}$$
$$C_{21} - t_{21} + M(1\text{-}a_{231}) \geq C_{23}$$
$$C_{32} - t_{32} + M(1\text{-}a_{312}) \geq C_{31}$$
$$C_{33} - t_{33} + M(1\text{-}a_{323}) \geq C_{32}$$
$$C_{31} - t_{31} + M(1\text{-}a_{331}) \geq C_{33}$$

Restricciones de no traslape de operaciones en una máquina dada

$$C_{21} - C_{11} + M(1\text{-}X_{112}) \geq t_{21}$$
$$C_{31} - C_{21} + M(1\text{-}X_{231}) \geq t_{31}$$
$$C_{31} - C_{11} + M(1\text{-}X_{131}) \geq t_{31}$$
$$C_{22} - C_{12} + M(1\text{-}X_{122}) \geq t_{22}$$
$$C_{32} - C_{22} + M(1\text{-}X_{232}) \geq t_{32}$$
$$C_{32} - C_{12} + M(1\text{-}X_{132}) \geq t_{32}$$
$$C_{23} - C_{13} + M(1\text{-}X_{123}) \geq t_{23}$$
$$C_{33} - C_{23} + M(1\text{-}X_{233}) \geq t_{33}$$
$$C_{33} - C_{13} + M(1\text{-}X_{133}) \geq t_{33}$$

4.3 MODELO GRÁFICO

El problema de calendarización Job-Shop puede ser representado también por medio de un gráfico disyuntivo.

El gráfico disyuntivo G = (***N, A, E***) se define como sigue:

- ***N*** contiene los nodos que representan todas las operaciones
- ***A*** contiene los arcos que conectan a operaciones consecutivas del mismo trabajo
- ***E*** contiene los arcos disyuntivos que conectan a las operaciones a ser procesadas por la misma máquina

Un arco disyuntivo puede ser fijado por cualquiera de sus dos posibles orientaciones. La calendarización establecerá la orientación de todos los arcos disyuntivos así mismo determinará la secuencia de las operaciones en las mismas máquinas. Una vez que una secuencia es determinada para una máquina, los arcos disyuntivos que conectaban a las operaciones a ser procesadas, son reemplazados por una flecha orientada de precedencia o arco conjuntivo. El conjunto de arcos disyuntivos E puede ser descompuesto en grupos Ek, por ejemplo

$$E = E_1 \cup E_2 \cup E_2 \cup E_3 \cdots\cdots E_m$$

La Figura 4-3 ilustra el gráfico disyuntivo para tres trabajos, tres máquinas, donde cada trabajo consiste de tres operaciones. Los nodos de N = {0,1,2,3,4,5,6,7,8,9,10} corresponden a las operaciones, donde los nodos 0 y 10 son operaciones sin valor y sirven para representar el "inicio" y el "termino" del gráfico.

Los arcos conjuntivos de A = {1,2),(2,3),(4,5),(5,6),(7,8),(8,9)} corresponden a las restricciones de precedencia en las operaciones del mismo trabajo.

Los arcos disyuntivos (las líneas punteadas) de E_1 = {(1,5),(5,9),(9,1)} corresponde a las operaciones a realizarse en la máquina 1, los arcos disyuntivos de E_2 = {(4,2),(2,8),(8,4)} corresponde a las operaciones a realizarse en la máquina 2 y los arcos disyuntivos de E_3 corresponde a las operaciones a realizarse en la máquina 3, E_3 = {(7,3),(3,6),(6,7)}.

El problema de Job-Shop se resuelve al encontrar el orden de las operaciones en cada máquina, esto es fijar la orientación de los arcos disyuntivos de manera que el gráfico resultante sea no cíclico, (esto es que no halla conflictos de precedencia entre las operaciones) y que la longitud de la ruta más larga entre los nodos de inicio y terminación sea mínima. La longitud de la ruta más larga determina el makespan

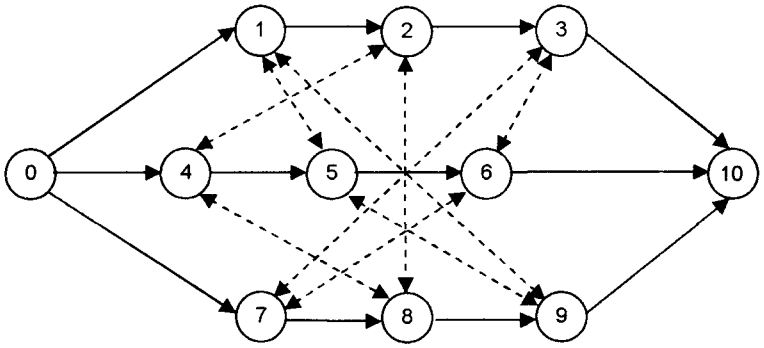


Figura 4-3. Grafico disyuntivo para un problema de tres máquinas tres trabajos

4.3.1 EJEMPLO

Para el ejemplo dado por la Tabla 3-1, su representación gráfica se muestra a continuación

Tabla 3-1

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

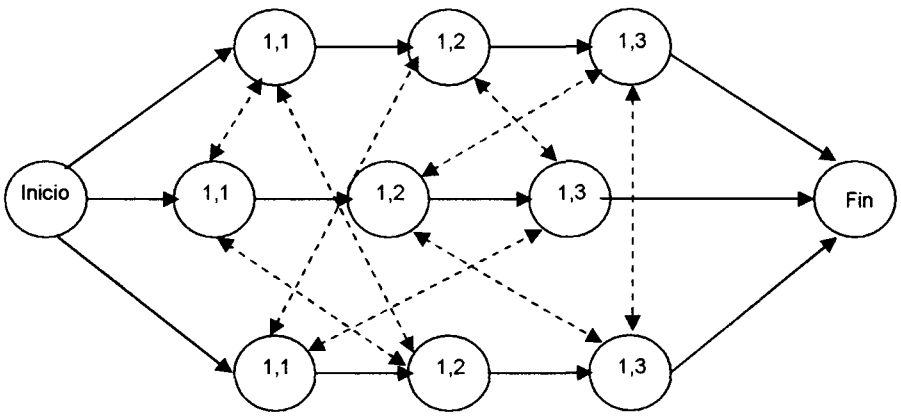


Figura 4-4. Representación gráfica del problema de la tabla 3-1

Solución factible

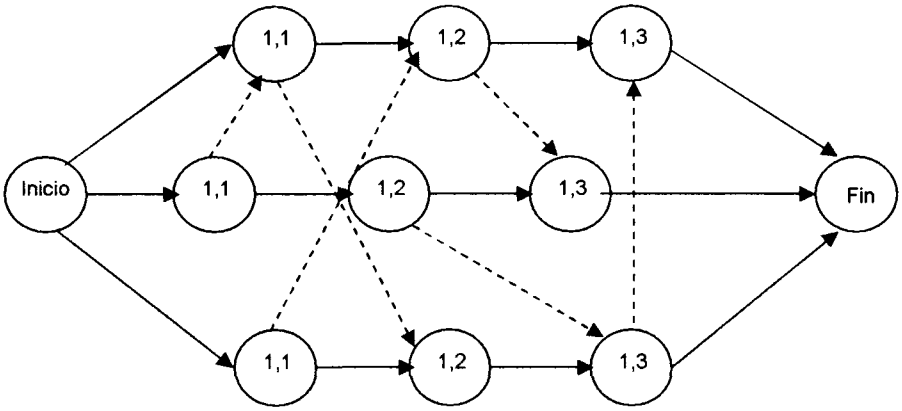


Figura 4-5. Representación gráfica de una solución factible del problema de la tabla 3-1

5 APLICACIÓN DE ALGORITMOS GENÉTICOS PARA EL PROBLEMA DE CALENDARIZACIÓN JOB-SHOP

5.1 REPRESENTACIÓN DE SOLUCIONES DEL PROBLEMA DE CALENDARIZACIÓN JOB SHOP PARA LA APLICACIÓN DE ALGORITMOS GENÉTICOS.

En el problema de calendarización Job-Shop, debido a la existencia de restricciones de precedencia entre operaciones, no es fácil la formación de una representación natural, esto quiere decir, que no se puede formar una buena representación de las restricciones de precedencia mediante un sistema de desigualdades.

Una cuestión muy importante en la implementación de un algoritmo genético para el problema de Job-Shop es diseñar una apropiada representación de soluciones y al mismo tiempo diseñar los operadores genéticos para el problema en específico con el fin de que los cromosomas generados en cualquiera de las fases iniciales o procesos de evolución generen secuencias factibles. Esta es la fase crucial que afecta a todos los pasos siguientes del algoritmo genético.

Durante los años en que se han estudiado los algoritmos genéticos para el problema de calendarización Job-Shop se han propuesto las siguientes representaciones:

- Representación basada en operaciones
- Representación basada en los trabajos
- Representación basada en lista de preferencia
- Representación basada en relaciones de pares de trabajos
- Representación basada en reglas de despacho
- Representación basada en gráficos disyuntivos
- Representación basada en tiempos de terminación
- Representación basada en las máquinas
- Representación por clave aleatoria

Estas representaciones pueden ser clasificadas dentro de dos estrategias básicas de codificación: Directa e Indirecta

En la *estrategia directa* la calendarización es codificada en el cromosoma, y el algoritmo genético es usado para evolucionar este cromosoma con el fin de obtener la mejor calendarización. Las representaciones que pertenecen a esta clase son: la representación basada en operaciones, la representación basada en los trabajos, la representación basada en relaciones de pares de trabajos, la representación basada en tiempos de terminación y la representación aleatoria.

En la *estrategia indirecta*, tal como en la representación basada en reglas de prioridad, una secuencia de reglas de despacho, es codificada en un cromosoma, y el algoritmo genético es usado para evolucionar dicho cromosoma y de esta manera obtener una mejor secuencia de reglas de despacho. Por lo tanto la solución óptima se construye con la secuencia de reglas de despacho obtenida por el algoritmo genético. Las representaciones que pertenecen a esta clase son: la representación basada en lista de preferencia, la representación basada en reglas de despacho, la representación basada en gráficos disyuntivos y la representación basada en las máquinas.

A continuación se discutirá cada una de las representaciones en base al ejemplo dado por la Tabla 3-1

Tabla 3-1

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

5.1.1 REPRESENTACIÓN BASADA EN OPERACIONES.

Esta representación codifica la calendarización como una secuencia de operaciones y cada gen representa una operación. Una forma para nombrar cada operación es usando números naturales y combinándolos por medio de permutaciones, desafortunadamente debido a la existencia de restricciones de precedencia no todas las permutaciones de números naturales definen secuencias factibles. En los artículos [16], [17], se propone la siguiente alternativa: nombrar todas las operaciones de un trabajo con el mismo símbolo y entonces interpretar las operaciones de acuerdo al orden en que aparecen en la secuencia para un cromosoma dado. Para un problema de n trabajos y m máquinas, un cromosoma debe contener $n \times m$ genes. Cada trabajo aparece en el cromosoma exactamente m veces y cada gen representa una operación dependiente del trabajo al que pertenece. Por lo tanto cualquier permutación del cromosoma produce una calendarización factible.

Considerando el problema de *tres máquinas tres trabajos* mostrado en la Tabla 3-1, suponga el cromosoma formado como sigue [3 2 2 1 1 2 3 1 3], donde 1 es la representación para el trabajo j_1, 2 es la representación para el trabajo j_2 y el 3 es la representación para el trabajo j_3. Debido a que cada trabajo tiene tres operaciones, este debe aparecer tres veces en el cromosoma. Por ejemplo hay tres 2's, en el cromosoma, los cuales representan las tres operaciones del trabajo j_2, el primer 2 corresponde a la primer operación del trabajo j_2, la cual será procesada en la máquina 1, el segundo 2 corresponde a la segunda operación del trabajo j_2, la cual será procesada en la máquina 3 y el tercer 2 corresponde a la tercera operación del trabajo j_2, la cual será procesada en la máquina 2. Se puede observar que todas las operaciones para el trabajo j_2 se representan con el mismo símbolo, un 2 y se interpretan de acuerdo al orden de aparición en la secuencia del cromosoma. La siguiente figura muestra gráficamente lo que se ha estado explicando

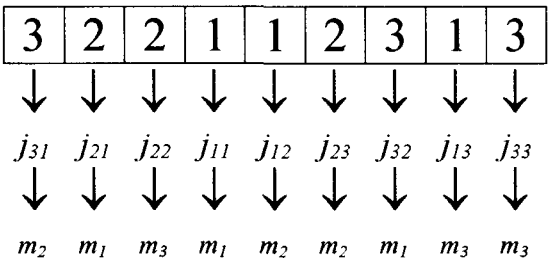


Figura 5-1. Representación basada en operaciones

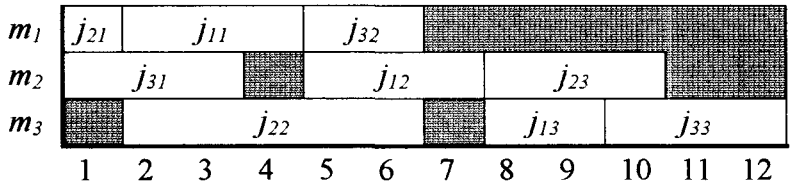


Figura 5-2 Decodificación del cromosoma [3 2 2 1 1 2 3 1 3]

5.1.2 REPRESENTACIÓN BASADA EN TRABAJOS

Esta representación consiste de una lista de n trabajos. La calendarización se construye de acuerdo a la secuencia de estos trabajos en el cromosoma. Para una secuencia de trabajos dada, todas las operaciones del primer trabajo en la lista se calendarizan primero y posteriormente las operaciones del segundo trabajo en la lista son consideradas. La primera operación del trabajo en cuestión se coloca en la mejor posición disponible de la cola de espera de la máquina que realizará está operación; después la segunda operación es colocada; y así sucesivamente hasta que todas las operaciones del trabajo son calendarizadas. Este proceso se repite con cada uno de los trabajos considerando la lista con la secuencia adecuada. Cualquier permutación de trabajos genera una calendarización factible. En el artículo [20] se usa esta representación para tratar con un problema de calendarización estática en el entorno de Manufactura Flexible

Para ejemplificar la representación basada en trabajos usaremos el problema mostrado en la Tabla 3-1.

Supóngase el cromosoma

[2 3 1]

El primer trabajo que será procesado es el trabajo j_2. La restricción de precedencia de operaciones para j_2 es [m_1 m_2 m_3], y los tiempos de procesamiento correspondientes a cada máquina son [1, 5, 3]. El primer trabajo j_2 es calendarizado como se muestra en la figura

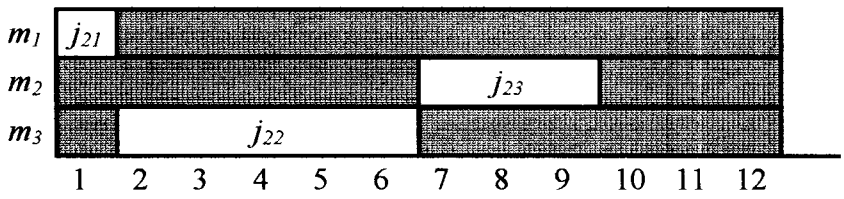


Figura 5-3. Calendarización para el primer trabajo j_2

El siguiente trabajo a ser procesado es el trabajo j_3, la precedencia de operaciones entre máquinas es [m_2 m_1 m_3], y el tiempo de procesamiento correspondiente en cada máquina es [3 2 3]. Cada una de las operaciones es calendarizada en la mejor posición disponible, como se muestra en la siguiente figura.

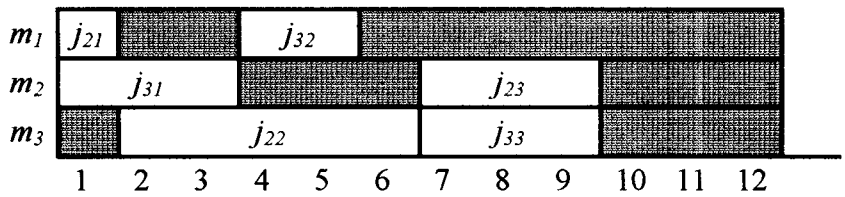


Figura 5-4. Calendarización para el segundo trabajo j_3

Finalmente el trabajo j_1 es calendarizado como se muestra en la siguiente figura

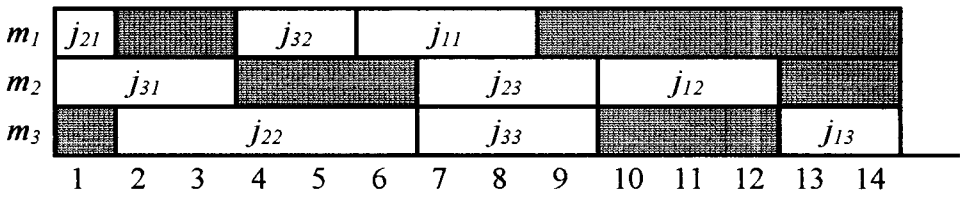


Figura 5-5. Calendarización para el tercer trabajo j_1

5.1.3 REPRESENTACIÓN BASADA EN LISTA DE PREFERENCIA

Esta representación fue propuesta por primera vez por primera vez por Lawrence Davis en [18]. Falkenauer and Bouffouix para tratar a un problema de calendarización Job-Shop con tiempos de liberación y fechas de vencimiento [19]. Kobayashi también adopta este tipo de representación en su artículo [21]

Para un problema de calendarización Job-Shop con n trabajos y m máquinas, se forma un cromosoma que consiste de:

- m subcromosomas (un subcromosoma para cada máquina).
- Cada subcromosoma es una cadena de símbolos con longitud n,
- Cada símbolo representa una operación que tiene que ser procesada en una máquina determinada.

Las listas de preferencia o de prioridad se forman a partir de reglas de despacho, asignadas a cada máquina. Hay una lista de preferencia por estación en el cromosoma. Una lista de preferencia es simplemente una lista de todas las operaciones a realizarse en determinada estación.

Los subcromosomas no describen la secuencia de operaciones en la máquina, ya que estos son listas de preferencia; cada máquina tiene su propia lista de preferencia. La calendarización actual es deducida del cromosoma a través de una simulación, la cual analiza el estado de la cola de espera de cada máquina y si es necesario, usa la lista de preferencia para determinar la calendarización; es decir, la operación que aparece primero en la lista de preferencia será seleccionada.

A continuación se mostrara un ejemplo para deducir la calendarización para un cromosoma dado. Considerando el ejemplo de la Tabla 3-1, supongamos el cromosoma

$$[(2\ 3\ 1)(1\ 3\ 2)(2\ 1\ 3)]$$

- (2 3 1) es la lista de preferencia para máquina m_1
- (1 3 2) es la lista para la máquina m_2
- (2 1 3) es la lista para la máquina m_3.

De la lista de preferencia se puede observar que las primeras operaciones son el j_2 en m_1, j_1 en m_2 y j_2 en m_3.

$$[(\mathbf{2}\ 3\ 1)(\mathbf{1}\ 3\ 2)(\mathbf{2}\ 1\ 3)]$$

De acuerdo con las restricciones de precedencia de operaciones, solo j_2 en m_1 es calendarizable, además es la primera operación en la secuencia de m_1. Esto se muestra en la siguiente figura.

Tabla 5-1. Calendarización del trabajo 2 en la máquina 1

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

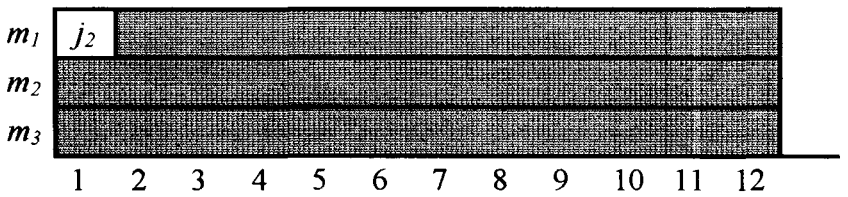


Figura 5-6. Calendarizando el trabajo 2 en la máquina 1

La siguiente operación calendarizable es j_2 en m_3.

$$[(\cancel{2}\ 3\ 1)(\mathbf{1}\ 3\ 2)(\mathbf{2}\ 1\ 3)]$$

Tabla 5-2 Calendarización del trabajo 2 en la máquina 3

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

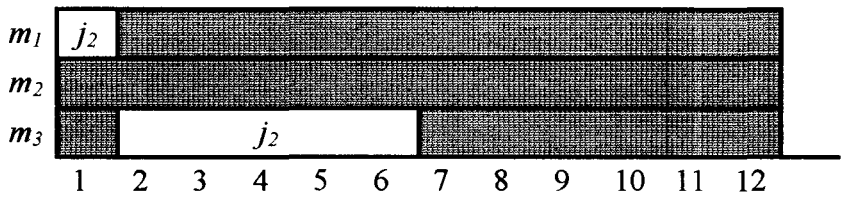


Figura 5-7. Calendarizando el trabajo 2 en la máquina 3

Ahora la actual preferencia en operaciones son j_3 en m_1 y j_1 en m_2 y m_3, ya que las demás operaciones no son calendarizables en el tiempo actual. Entonces las operaciones en segundo lugar de preferencia son j_1 en m_1 y j_3 en m_2 y m_3, pero las operaciones calendarizables son j_1 en m_1 y j_3 en m_2.

[(~~2~~ **3** 1)(**1** 3 2)(~~2~~ **1** 3)]

Tabla 5-3. Calendarización del trabajo 1 en la máquina 3 y del trabajo 2 en la máquina 1

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

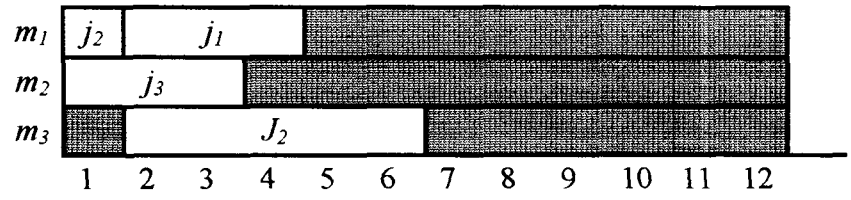


Figura 5-8. Calendarizando el trabajo 1 en la máquina 3 y el trabajo 2 en la máquina

Las siguientes operaciones calendarizables son j_3 en m_1 y j_1 en m_2.

[(~~2~~ **3** ~~1~~)(**1** ~~3~~ 2)(~~2~~ **1** 3)]

Tabla 5-4. Calendarización del trabajo 2 en la máquina 3 y del trabajo 1 en la máquina 2

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

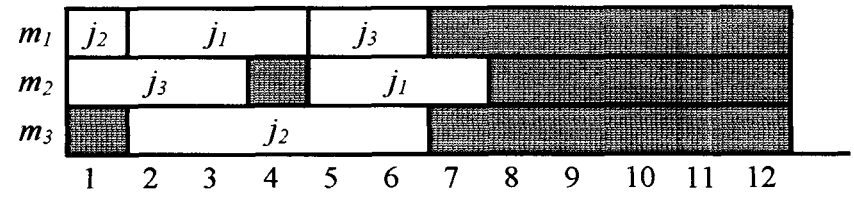


Figura 5-9. Calendarizando el trabajo 1 en la máquina 3 y el trabajo 2 en la máquina 1

Ahora las operaciones calendarizables en este momento son j_2 en m_2 y j_1 en m_3.

[(2 3 1)(1 3 **2**)(2 **1** 3)]

Tabla 5-5. Calendarización del trabajo 3 en la máquina 2 y del trabajo 2 en la máquina 3

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

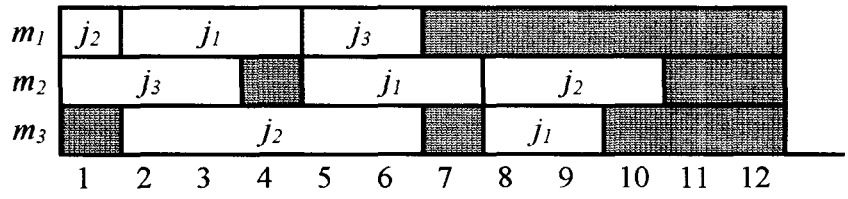


Figura 5-10. Calendarizando el trabajo 1 en la máquina 3 y el trabajo 2 en la máquina 1

La última operación que se va a calendarizar es j_3 en m_3. Con esto se ha completado la deducción de la secuencia de un cromosoma y se ha obtenido una calendarización factible con un makespan de12 unidades.

[(2 3 1)(1 3 **2**)(2 **1** 3)]

Tabla 5-6. Calendarización del trabajo 3 en la máquina 3.

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

Con este proceso de decodificación, todos los cromosomas generados, siempre producen una calendarización factible.

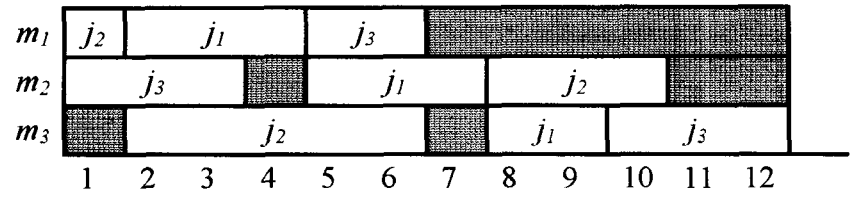


Figura 5-11. Calendarizando el trabajo 3 en la máquina 3.

5.1.4 REPRESENTACIÓN BASADA EN RELACIONES DE PARES DE TRABAJOS

Los investigadores Nakano y Yamada en su trabajo [22] usaron una matriz binaria para codificar la calendarización. Este matriz se determina de acuerdo a la relación de precedencia entre un par de trabajos que son procesados por la misma máquina.

Se define una variable binaria, para indicar la relación de precedencia entre una pareja de trabajos. La variable binaria se define como sigue:

$$x_{ijm} \begin{cases} 1, & \text{si el trabajo } i \text{ precede al trabajo } j \text{ en la máquina } m \\ 0, & \text{de otra forma} \end{cases}$$

Considerando el problema de calendarización de tres máquinas tres trabajos de la Tabla 5-7, donde también se muestran las restricciones de precedencia de trabajos y una calendarización factible, se mostrará la representación basad en relaciones de pares de trabajos.

Tabla 5-7. Problema de calendarización de tres-trabajos, tres-máquinas.

Precedencia de operaciones					Calendarización factible			
Trabajos	Secuencia de máquinas				Máquinas	Secuencia de trabajos		
j_1	m_1	m_2	m_3		m_1	j_2	j_1	j_3
j_2	m_1	m_3	m_2		m_2	j_3	j_1	j_2
j_3	m_2	m_1	m_3		m_3	j_2	j_1	j_3

Examinaremos la relación de precedencia entre las parejas de trabajos. Para (j_1, j_2) de acuerdo a la secuencia dada en las máquinas $[m_1, m_2, m_3]$, tenemos $(x_{121}\ x_{122}\ x_{123}) = (0\ 1\ 0)$. Para el par de trabajos (j_1, j_3), tenemos $(x_{131}\ x_{132}\ x_{133}) = (1\ 0\ 1)$. Para el par de trabajos (j_2, j_3), la relación de precedencia en las máquinas (m_1, m_3, m_2) son $(x_{231}\ x_{233}\ x_{232}) = (1\ 1\ 0)$. La secuencia de la variable x_{ijm} para un par de trabajos debe ser consistente con la secuencia de operaciones del primer trabajo i. Por ejemplo para el par de trabajos (j_2,j_3), la secuencia de operaciones para el trabajo j_2 es (m_1, m_3, m_2), entonces las variables binarias son ordenadas como $(x_{231}\ x_{233}\ x_{232})$ en vez de $(x_{231}\ x_{232}\ x_{233})$. Resumiendo estos resultados, obtenemos la matriz binaria que representa una calendarización factible

$$\begin{array}{ll} (j_1,j_2) \text{ en } (m_1, m_2, m_3): & \\ (j_1,j_3) \text{ en } (m_1, m_2, m_3): & \begin{pmatrix} x_{121} & x_{122} & x_{123} \\ x_{131} & x_{132} & x_{133} \\ x_{231} & x_{233} & x_{232} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \\ (j_1,j_2) \text{ en } (m_1, m_3, m_2): & \end{array}$$

Cabe señalar que esta es la representación más compleja, entre todas las representaciones mostradas.

5.1.5 REPRESENTACIÓN BASADA EN REGLAS DE DESPACHO

En su trabajo Dorndorf y Pesch proponen un algoritmo genético basado en reglas de despacho, donde un cromosoma es codificado como una secuencia de reglas de despacho para la asignación de trabajos, entonces la calendarización se construye con una heurística basada en la secuencia de las reglas de despacho [23]. Los algoritmos genéticos son usados aquí para hacer evolucionar a los cromosomas que ayudaran a producir una mejor secuencia de reglas de despacho.

Las reglas de prioridad de despacho son probablemente las reglas más aplicadas en "heurísticas" para la solución de problemas de calendarización, debido a su fácil implementación. El algoritmo de Giffter y Thompson puede ser considerado como la base de todas las heurísticas basadas en reglas de despacho, donde el problema principal es encontrar una regla de despacho efectiva.

Para un problema de n trabajos m máquinas, un cromosoma es una cadena de longitud n × m, representada como:

$$(p_1, p_2, p_3, \ldots, p_{nm})$$

Donde el dato p_i representa una de las reglas despacho de un conjunto de reglas previamente especificado. La entrada en la ith posición indica que un conflicto en la ith iteración del algoritmo de Giffter y Thompson podría ser resuelto usando la regla de prioridad p_i.

Antes de definir el procedimiento para realizar la calendarización se definirán ciertos parámetros útiles en la explicación de este procedimiento:

PS_t Una secuencia parcial que contiene t operaciones calendarizables

S_t El conjunto de operaciones calendarizables en la iteración t, correspondientes a PS_t

σ_i El tiempo mas temprano en el que la operación i $\in S_t$ puede empezar

ϕ_i El tiempo mas temprano en el que la operación i $\in S_t$ puede ser completada

C_t El conjunto de las operaciones en conflicto en la iteración t

El procedimiento para deducir la calendarización de un cromosoma dado $(p_1, p_2, \ldots, p_{nm})$ es el siguiente:}

Paso 1 Inicializar t =1 (es decir iteración 1) y empezar con PS_t como una secuencia parcial nula y formar a S_t con las operaciones que no tienen predecesores.

Paso 2 Determinar el $\phi_t^* = \min_{i \in St} \{\phi_i\}$ y la máquina m* en la cual puede ser realizado. Si se obtiene más de una máquina, se escoge una de ellas aleatoriamente.

Paso 3 Formar un conjunto de operaciones en conflicto C_t que incluya todas las operaciones i $\in$ S_t con $\sigma_i < \phi_i^*$ que sean procesadas en la máquina m*. Seleccionar una operación de por la regla de prioridad p_t y añadir esta operación a PS_t y de esta manera crear una nueva secuencia parcial PS_{t+1}. Si existe más de una operación que cumpla con la regla de prioridad p_t, entonces se escoge una de estas operaciones aleatoriamente.

Paso 4 Actualizar PS_{t+1} removiendo la operación seleccionada de S_t y añadiendo el sucesor directo de la operación en cuestión a S_t. Incrementar t en uno.

Paso 5 Regresar al paso 2 hasta que se genere la calendarización completa

Ejemplo. Se usaran las cuatro reglas de prioridad mostradas en la tabla siguiente:

Tabla 5-8. Reglas de despacho seleccionadas

Número	Regla	Descripción
1	SPT	Selecciona una operación con el menor tiempo de procesamiento
2	LPT	Selecciona una operación con el mayor tiempo de procesamiento
3	MWR	Selecciona una operación de el trabajo con el mayor tiempo de procesamiento restante
4	LWR	Selecciona una operación de el trabajo con el menor tiempo de procesamiento restante

Considerar el siguiente cromosoma [1 2 2 1 4 4 2 1 3], donde 1 representa a la regla **SPT**, 2 a la regla **LPT**, 3 a la regla **MWR** y 4 a la regla **LWR**. En el paso inicial tenemos

$$S_1 = \{o_{111}, o_{211}, o_{312}\}$$
$$\phi_i^* = \min\{3, 1, 3\} = 1$$
$$m^* = 1$$
$$C_1 = \{o_{111}, o_{211}\}$$

Se utiliza el índice *jom* para nombrar cada operación, donde j indica el trabajo, o indica la operación y m indica la maquina a realizar esa operación

Tabla 5-9. Operaciones factibles de calendarización

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

Regresando al ejemplo, las operaciones o_{111} y o_{211} corresponden a la máquina m_1. Debido a que el primer gen en el cromosoma es 1, el cual representa a la regla de prioridad SPT, entonces la operación 211 (índice trabajo, operación, máquina) es calendarizada en la máquina m_1 como se muestra en la siguiente figura

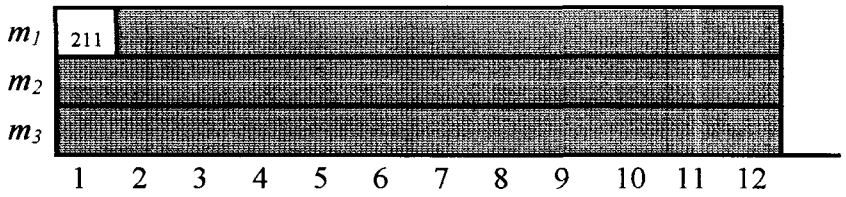


Figura 5-12. Calendarizando la operación 211 en m1

Después de la actualización de datos, tenemos

$$S_2 = \{o_{111}, o_{223}, o_{312}\}$$
$$\phi_2^* = \min\{4, 6, 3\} = 3$$
$$m^* = 2$$
$$C_2 = \{o_{312}\}$$

Tabla 5-10. Calendarización de la operación 312 en m_2

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

La operación o_{312} es calendarizada en la máquina m2 como se muestra en la siguiente figura.

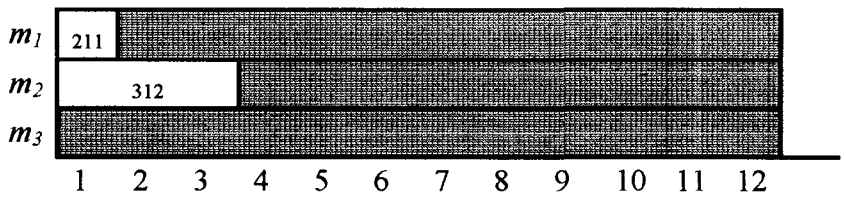


Figura 5-13. Calendarizando la operación 312 en m2

Después de actualizar los datos, tenemos

$$S_3 = \{o_{111}, o_{223}, o_{321}\}$$
$$\phi_2^* = \min\{4, 6, 3\} = 3$$
$$m^* = 1$$
$$C_2 = \{o_{111}, o_{321}\}$$

Las operaciones o_{111} y o_{312} corresponden a la maquina m1. debido a que el tercer gen en el cromosoma dado es 2, el cual representa a la regla de prioridad LPT, la operación es calendarizada en la máquina m1 como se muestra en la figura siguiente.

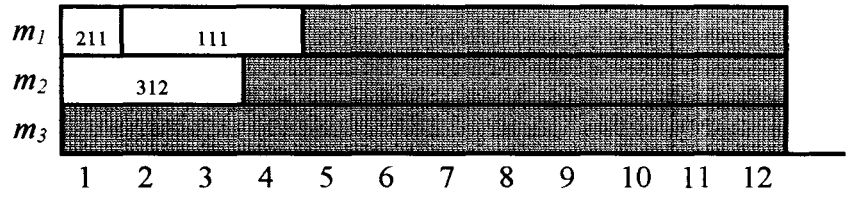


Figura 5-14. Calendarización de la operación 111 en m1

Se repiten estos pasos hasta que la calendarización completa es deducida del cromosoma dado como se muestra en la figura

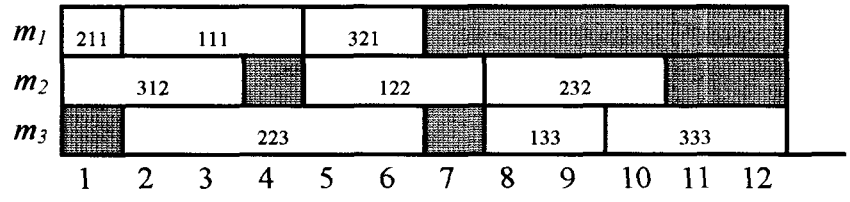


Figura 5-15. Calendarización completa

5.1.6 REPRESENTACIÓN BASADA EN GRÁFICO DISYUNTIVO

Tamaki y Nishikawa proponen una representación basada en un gráfico disyuntivo, el cual también puede ser visto como un tipo de representación basada en la relación de pares de trabajos. El problema de calendarización Job Shop puede ser representado con un gráfico disyuntivo. El gráfico disyuntivo G = (***N, A, E***) se define como sigue: ***N*** contiene los nodos que representan todas las operaciones, ***A*** contiene los arcos que conectan las operaciones consecutivas del mismo trabajo, y ***E*** contiene los arcos disyuntivos que conectan a las operaciones que tienen que ser procesadas por la misma máquina. Las restricciones disyuntivas son representadas por una esquina en ***E***. Un arco disyuntivo puede ser fijado por cualquiera de sus dos posibles orientaciones. La calendarización establecerá la orientación de todos los arcos disyuntivos, así mismo determinara la secuencia de las operaciones en las mismas máquinas. Una vez que una secuencia es determinada para una máquina, los arcos disyuntivos que conectaban

las operaciones que debe procesar la máquina, son reemplazados por una flecha orientada de precedencia o arco conjuntivo. La siguiente figura ilustra el gráfico disyuntivo para un ejemplo de tres máquinas, tres trabajos.

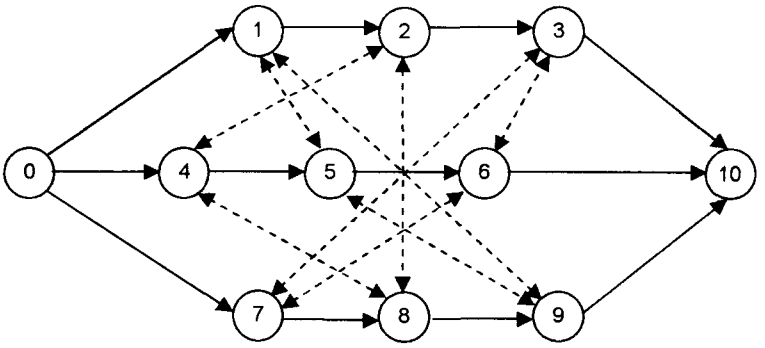


Figura 5-16. Gráfico disyuntivo para un problema de tres máquinas tres trabajos

Un cromosoma consiste de una cadena binaria que corresponde a una lista del orden de arcos disyuntivos en $\boldsymbol{E}$ como se muestra

Lista del orden de los arcos disyuntivos:	e_{15}	e_{19}	e_{59}	e_{24}	e_{28}	e_{48}	e_{36}	e_{37}	e_{67}
Cromosoma:	0	0	1	1	0	0	0	1	1

Figura 5-17. Representación basada en Gráfico disyuntivo

Donde e_{ij} representa a un arco disyuntivo entre los nodos i, j y esta definido como sigue:

$$e_{ij} = \begin{cases} 1, & \text{establece la orientación del arco disyuntivo del nodo } j \text{ al nodo } i \\ 0, & \text{establece la orientación del arco disyuntivo del nodo } i \text{ al nodo } j \end{cases}$$

El problema de calendarización Job-Shop consiste en encontrar un orden de operaciones en cada máquina, esto es establecer la orientación de los arcos disyuntivos tal que el gráfico resultante no sea cíclico para garantizar que no hay conflictos de precedencia entre operaciones. Es fácil ver que un cromosoma arbitrario puede producir un gráfico cíclico, lo cual significa que esa calendarización no es factible. De esta manera este cromosoma no es usado para representar la calendarización, pero es usado como referencia de decisión. Para deducir la calendarización se usa un procedimiento basado en la ruta crítica. Durante el proceso de deducción, cuando un conflicto de dos nodos (operaciones) ocurre en una máquina, el correspondiente bit del cromosoma es usado para establecer el orden de procesamiento de las dos operaciones, esto es para establecer la dirección del arco disyuntivo entre dos nodos.

5.1.7 REPRESENTACIÓN BASADA EN TIEMPOS DE TERMINACIÓN

Yamada y Nakano proponen una representación basada en tiempos de terminación. Un cromosoma es una lista ordenada de los tiempos de terminación de operación. Para el ejemplo de la Tabla 3-1.

Tabla 3-1.

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

El cromosoma puede ser representado como sigue:

$$[c_{111}\ c_{122}\ c_{133}\ c_{211}\ c_{223}\ c_{232}\ c_{312}\ c_{321}\ c_{333}]$$

Donde c_{jir} denota el tiempo de terminación para la operación i del trabajo j en la máquina r. Es fácil ver que tal representación no es adecuada para la mayoría de los operadores genéticos porque esto producirá una calendarización ilegal. Los investigadores mencionados anteriormente diseñaron un operador de cruce especial para este caso.

5.1.8 REPRESENTACIÓN BASADA EN LAS MÁQUINAS

Dorndorf y Pesh en su trabajo [23] propusieron un algoritmo genético basado en máquinas, donde un cromosoma es codificado como una secuencia de máquinas y la calendarización es construida mediante una secuencia basada en la heurística del cuello de botella.

La heurística del cuello de botella, propuesta por Adams, Balas y Zawack, es probablemente el procedimiento mas poderoso entre las heurísticas para el problema de calendarización Job-Shop [24].

Este procedimiento ordena las máquinas una por una, sucesivamente toma cada vez la máquina identificada como un cuello de botella entre las máquinas que no han sido secuenciadas todavía. En cualquier momento que una nueva máquina sea secuenciada, las secuencias previamente establecidas se reoptimizan localmente. La identificación del cuello de botella y los procedimientos de reoptimización local son basados en la solución repetida de un problema de calendarización de una máquina, que representa la versión simplificada del problema original. La principal contribución de esta metodología es la forma en que usa esta simplificación para decidir sobre el orden en el cual las máquinas podrían ser secuenciadas.

La heurística del cuello de botella esta basada en la clásica idea de dar prioridad a las máquinas que representan el cuello de botella. Diferentes medidas de calidad del cuello de botella redituará en diferentes secuencias de máquinas que representan cuellos de botella. La calidad de la calendarización obtenida por la heurística de cuello de botella depende en gran manera de la secuencia de las máquinas de cuello de botella. Adams, Balas y Zawack también proponen una heurística para considerar diferentes secuencias de máquina.

Dorndorf y Pesh proponen una estrategia genética para determinar la mejor secuencia de máquinas para la heurística del cuello de botella. Un cromosoma es una lista de máquinas ordenadas. Los Algoritmos Genéticos son usados en esta representación para evolucionar a los cromosomas que encontrarán una mejor secuencia de máquinas para la heurística de cuello de botella. La diferencia entre la heurística del cuello de botella y los algoritmos genéticos es que el cuello de botella no es un criterio de decisión para la selección de la siguiente máquina, la cual es controlada por el cromosoma dado.

Vamos a definir a M_0 como el conjunto de máquinas secuenciadas, y se define al cromosoma como $[m_1, m_2, \ldots, m_m]$. El procedimiento para deducir la calendarización trabaja como sigue:

5.1.8.1 Procedimiento

Paso 1 Inicializar M_0 como conjunto vació y i $\leftarrow$ 1 formar al cromosoma como $[m_1, m_2, \ldots, m_m]$.

Paso 2 Formar la secuencia de la máquina m_i Actualizar el conjunto $M_0 \leftarrow M_0 \cup \{m_i\}$

Paso 3 Reoptimizar la secuencia de cada máquina critica $mi \in M_0$, mientras se mantienen las otras secuencias fijas.

Paso 4 Actualizar i $\leftarrow$ i + 1. Entonces si i > m, se detiene el procedimiento; de otra manera ir al paso 2.

5.1.9 REPRESENTACIÓN POR CLAVE ALEATORIA

La representación por clave aleatoria fue introducida por Bean [25]. Esta representación codifica una solución con números aleatorios, estos valores son usados como claves de clasificación para decodificar la solución.

Para un problema de calendarización de n trabajos, m máquinas, cada gen, (es decir una clave aleatoria) consiste de dos partes: un entero {1, 2,..., m} y una fracción generada aleatoriamente de 0 a 1. La parte entera de la clave aleatoria se interpreta como la asignación de máquina de cada trabajo. Al ordenar las claves aleatorias, de menor a mayor, se obtiene la secuencia de trabajos en cada máquina

Considerando el ejemplo de la **Tabla 3-1.**

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

Supóngase el siguiente cromosoma

[1.34 1.09 1.88 2.66 2.91 2.01 3.23.3.21 3.44]

Ordenando las claves para la máquina 1 de manera ascendente la secuencia de trabajos es: 2→1→3, para la máquina 2 la secuencia de trabajos es: 3→1→2, y para la máquina 3 la secuencia de trabajos es 2→1→3. Se pueden representar las operaciones de la siguiente forma O_{jm}, siendo j el trabajo en la máquina m. De esta manera nuestro cromosoma quedaría de la siguiente forma:

$$[O_{21}\ O_{11}\ O_{31}\ O_{32}\ O_{12}\ O_{22}\ O_{23}\ O_{13}\ O_{33}]$$

5.2 OPERADORES DE CRUZAMIENTO PARA PROBLEMAS DE CALENDARIZACIÓN JOB SHOP.

Diversos operadores de cruzamiento han sido propuestos para mejorar los espacios de búsqueda en los algoritmos genéticos. A continuación se describen algunos de los mecanismos de cruzamiento más representativos.

5.2.1 CRUZAMIENTO PARCIALMENTE MAPEADO (PMX).

Este cruzamiento fue propuesto por Goldberg y Lingle [27]. Este mecanismo de cruce, puede ser visto como una variación del cruzamiento de dos puntos de corte. Los pasos de este mecanismo de cruce son los siguientes:

1. Seleccionar aleatoriamente dos puntos de corte a lo largo de la cadena o cromosoma. A la subcadena definida por los dos puntos de corte se le llama sección mapeada.

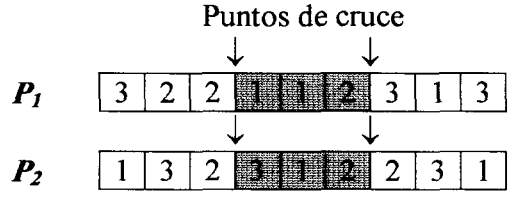


Figura 5-18. Puntos de cruce

2. Se intercambian entre padres las subcadenas para producir descendencia.

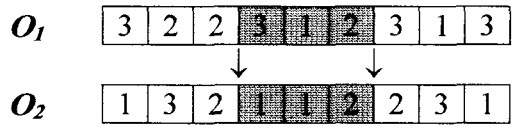


Figura 5-19. Descendencia no legalizada

3. Se determinan las relaciones de mapeo entre las dos secciones mapeadas.

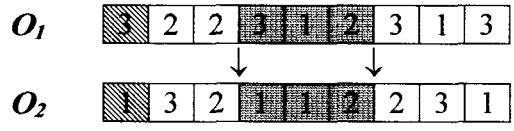


Figura 5-20. Relaciones de mapeo

4. Se legaliza la descendencia usando las relaciones de mapeo.

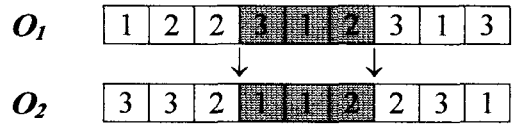


Figura 5-21. Descendencia legalizada

5.2.2 CRUZAMIENTO CON ORDEN (OX).

Este cruzamiento fue propuesto por Davis [27]. Este cruzamiento es similar al PMX con la diferencia de que usa un procedimiento distinto para la legalización de la descendencia.

Tomando el ejemplo anterior se describirán los pasos de este mecanismo de cruzamiento

1. Se seleccionan aleatoriamente dos puntos de corte a lo largo de la cadena o cromosoma, definiendo de esta manera una sección o subcadena de selección.

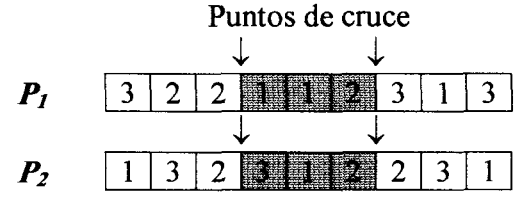


Figura 5-22. Puntos de cruce

2. Se van dejando huecos en el cromosoma padre P_2, en donde halla los mismos genes contenidos en la subcadena de selección P_1, haciendo esto de izquierda a derecha. Lo mismo se hace con el otro padre de forma recíproca.

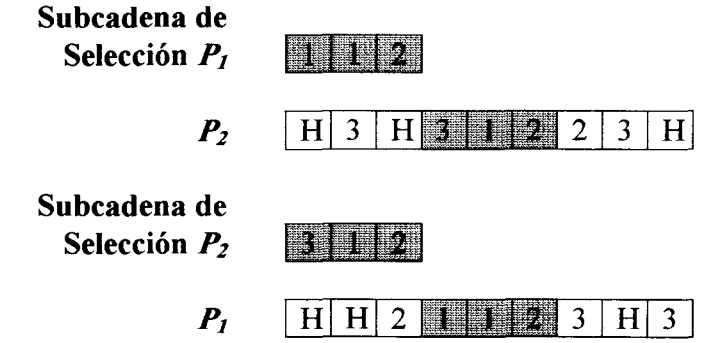


Figura 5-23. Generación de huecos en el cromosoma Padre

3. Los huecos se desplazan de manera que ocupen el lugar que delimitaron los puntos de cruce, y la subcadena de selección se desplaza al inicio de la cadena moviendo los genes que quedaron entre huecos en la subcadena inicial al final de la cadena.

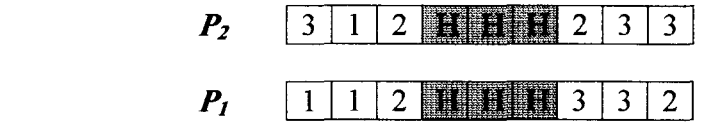


Figura 5-24. Desplazamiento de huecos

4. Los huecos son llenados por la subcadena de selección del otro padre, formando así la descendencia.

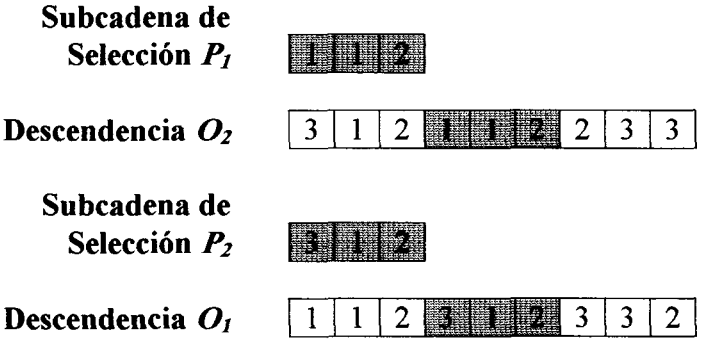


Figura 5-25. Formación de la descendencia

5.2.3 CRUZAMIENTO CON ORDEN EN LÍNEA (LOX).

Falkenauer y Bouffoix propusieron una versión modificada del cruzamiento en orden (OX), el cruzamiento con orden en línea. El cruzamiento en orden tiende a trasmitir en la descendencia la posición relativa de los genes de los padres mas que las posiciones absolutas de los genes, también en el cruzamiento en orden el cromosoma es considerado circular ya que este operador fue diseñado para el problema del agente viajero (TSP). En el problema de Job Shop, el cromosoma no puede ser considerado circular, por esta razón Falkenauer y Bouffoix desarrollaron una variante del cruzamiento (OX), el llamado cruzamiento con orden en línea (LOX) en el cual el cromosoma es considerado lineal en vez de circular.

Es importante mencionar que este operador de Cruzamiento fue diseñado para ser utilizado en la representación de listas de preferencia. Para un problema de *n* trabajos *m* máquinas un cromosoma codificado en la representación de listas de preferencia está formado de *m* subcromosomas y cada subcromosoma consiste de *n* genes, cada subcromosoma es una lista de preferencia de operaciones para una máquina.

Para describir el mecanismo de este operador de cruzamiento se considerará el subcromosoma como padre. El cruzamiento (LOX) funciona como se describe a continuación.

Se selecciona aleatoriamente una sublista en los padres como se describió en el cruzamiento (OX)

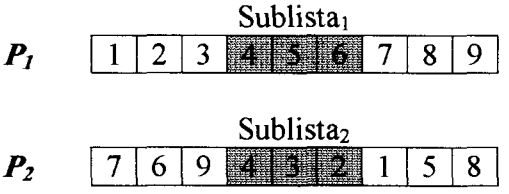


Figura 5-26. Selección de sublistas

Se remueve la sublista$_2$ del padre P_1, dejando algunos huecos y después se desplazan estos huecos hacia el centro hasta alcanzar la sección de cruce. De la misma forma se remueve la sublista$_1$ del padre P_2 y se desplazan los huecos a la sección de huecos.

Se remueve la sublista$_2$ de P_1:

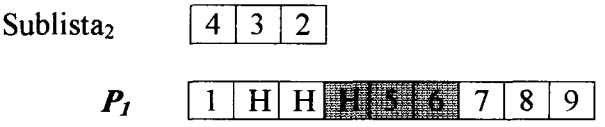


Figura 5-27. Remoción de sublista$_2$ de P_1

Se desplazan los huecos a la sección de cruce:

Figura 5-28. Desplazamiento de huecos a la sección de cruce

Se remueve la sublista$_1$ de P_2:

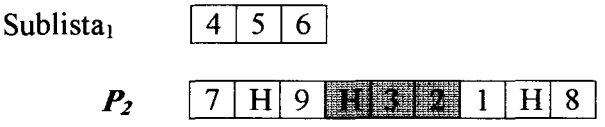


Figura 5-29. Remoción de sublista$_1$ de P_2

Se desplazan los huecos a la sección de cruce:

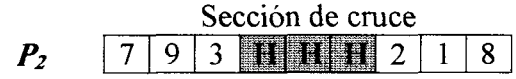


Figura 5-30. Desplazamiento de huecos a la sección de cruce

Se inserta la sublista$_1$ en los huecos del padre P_2 para formar la descendencia O_1 y se inserta la sublista$_2$ en los huecos del padre P_1 para formar la descendencia O_2.

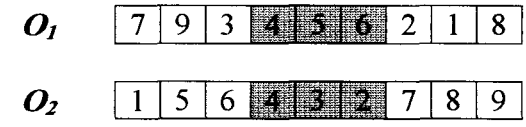


Figura 5-31. Descendencia formada

Este operador de cruzamiento puede conservar en medida de lo posible las posiciones relativas entre genes y las posiciones absolutas entre los extremos de los cromosomas padres. Los extremos de los padres corresponden a las operaciones de prioridad alta y baja.

5.2.4 CRUZAMIENTO DE INTERCAMBIO DE SUBSECUENCIAS.

Kobayashi, Ono y Yamamura propusieron el método de cruzamiento de intercambio de subsecuencias, inspirados en ideas similares aplicadas a la solución del problema del agente viajero (TSP). Una matriz de secuencias de trabajos es usada como representación. Para un problema de n trabajos m máquinas la codificación es una matriz de tamaño m x n, cada fila representa la secuencia de operaciones para una máquina. Una subsecuencia es definida como un conjunto de trabajos que son procesados consecutivamente en una máquina para ambos padres pero no necesariamente en el mismo orden.

Paso 1. Identificar las subsecuencias que existen en cada fila es decir en cada máquina en las secuencias de los padres.

Paso 2 Intercambiar estas subsecuencias máquina a máquina entre padres para crear descendencia

En la figura 5-32 se muestra un ejemplo del cruzamiento de intercambio de subsecuencias llamado también SXX por las iniciales en inglés. Este es un ejemplo de un problema de 6 trabajos 6 máquinas.

En esta figura se muestra que el cruzamiento SXX intercambia subsecuencias entre padres por cada máquina, cuando esta subsecuencia consiste de los mismos trabajos.

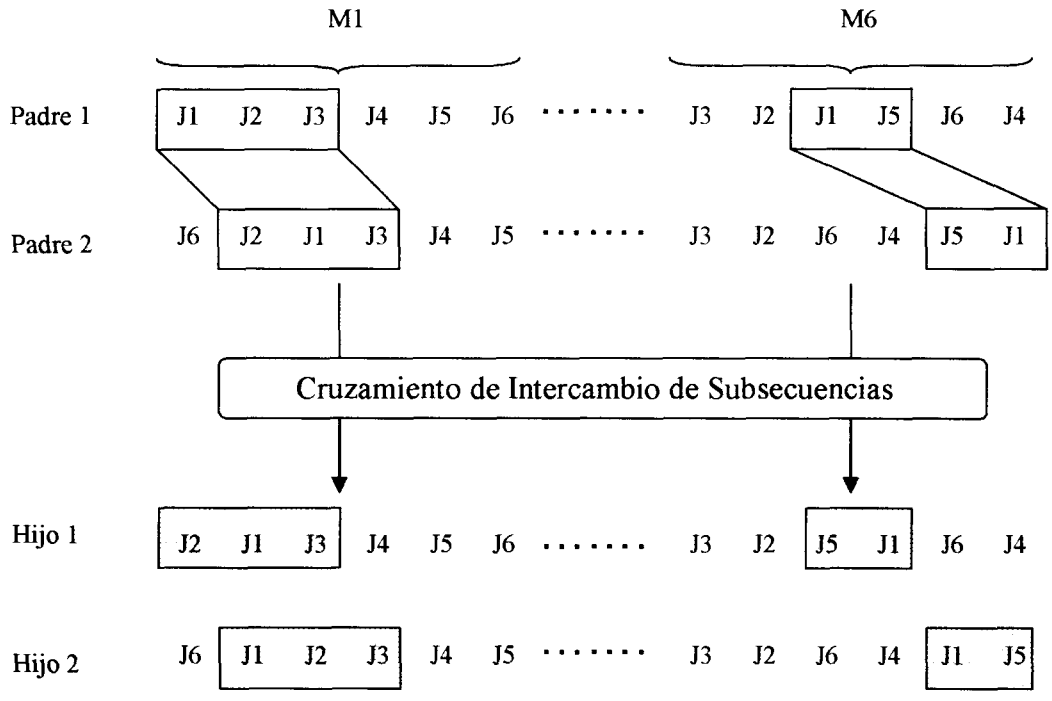


Figura 5-32. Cruzamiento de intercambio de sub-secuencias (SXX)

5.3 BÚSQUEDA GENÉTICA HÍBRIDA

Los problemas de optimización combinatoria están al alcance de los algoritmos genéticos. Comparado con la heurística convencional, los algoritmos genéticos no están bien definidos para las estructuras de sintonización fina que están muy cerca de soluciones óptimas, por lo tanto es esencial incorporar heurística convencional (como por ejemplo la búsqueda local) dentro de los algoritmos genéticos para construir un algoritmo genético más competitivo. Varios métodos de hibridización han sido propuestos para los problemas de calendarización Job-Shop, los cuales pueden ser clasificados en dos grupos.

- Operadores genéticos adaptados
- Heurística convencional incorporada dentro de algoritmos genéticos

El método por Operadores Genéticos Adaptados intenta inventar nuevos operadores genéticos inspirados de la heurística convencional, como por ejemplo el diseño de un nuevo cruzamiento basado en el Algoritmo de Giffter y Thompson o diseñando un nuevo operador de mutación basado en el mecanismo de búsqueda en vecindad. El segundo método implica la combinación de la heurística convencional dentro de los algoritmos genéticos donde sea posible. Esto puede hacerse de varias formas, incluyendo las siguientes:

- Incorporando la heurística en la inicialización para generar una población inicial bien adaptada, de esta manera, un algoritmo genético híbrido con elitismo puede garantizar tener un mejor desempeño que la heurística convencional.

- Incorporando heurística en la función de evaluación para decodificar cromosomas en sus respectivas secuencias de calendarización.

- Incorporando heurística de búsqueda local como una parte adicional al bucle básico del algoritmo genético, trabajando en conjunto con los operadores de mutación y cruzamiento, para un desempeño rápido en la optimización para mejorar la descendencia antes de empezar a evaluarlas.

Con el método híbrido, los algoritmos Genéticos son usados para realizar una exploración global en la población, mientras que los métodos heurísticos son usados para realizar la exploración local entre los cromosomas. Debido a las propiedades complementarias de los algoritmos genéticos y las heurísticas, el método híbrido supera a menudo cualquier método que funciona solamente con una de estas metodologías.

5.3.1 CRUZAMIENTO BASADO EN EL ALGORITMO GIFFLER Y THOMPSON

El cruzamiento basado en el Algoritmo Giffler y Thompson, fue propuesto por Yamada y Nakano [1]. El procedimiento de la generación del algoritmo Giffler y Thompson es una técnica de búsqueda de árbol. En cada paso, se identifican todos los conflictos de procesamiento (las operaciones que corresponden a la misma máquina) y un procedimiento de enumeración se usa para resolver estos conflictos en todas las formas posibles. En el operador de cruzamiento cuando se genera la descendencia, en cada, paso se identifican todos los conflictos de procesamiento de manera similar a lo realizado en el algoritmo de Giffler y Thompson, y entonces se escoge una operación del conjunto de operaciones en conflicto de acuerdo a una de las calendarizaciones de los padres.

Se nombraran ciertas variables para explicar el procedimiento de este cruzamiento.

o_{ji} = La iésima operación del trabajo j

S = El conjunto de operaciones calendarizables para una secuencia parcial de operaciones dada

Φ_{ji} = El tiempo de terminación mas temprano de la iésima operación del trabajo j en S

G_r = El conjunto de operaciones en conflicto en S, en la máquina r

El procedimiento para generar descendencia de dos padres es el siguiente:

Procedimiento: Cruzamiento basado en el Algoritmo Giffler y Thompson

Paso1. Se inicializará S de tal manera que incluya a todas las operaciones sin predecesor.

Paso2. Se determinará $\Phi^* = \min \{\Phi_{ji} | o_{ji} \in S\}$ y la máquina r* en la cual Φ^* puede ser realizada.

Paso 3. Se formará G_{r*} de manera que incluya todas las operaciones $o_{ji} \in S$ que se requieren en la máquina $r*$.

Paso 4. Se escoge una de las operaciones de G_{r*} como sigue:
1. Se genera un número aleatorio $\epsilon \in [0, 1)$ y se compara con la tasa de mutación p_m si ($\epsilon < p_m$), y entonces se escoge una operación arbitraria de G_{r*} como o_{ji*}.
2. De otra manera se selecciona un padre con igual probabilidad, y lo nombraremos p_s; se encuentra una operación o_{ji*}, la cual fue la primera en ser calendarizada en p_s a través de todas las operaciones en G_{r*}.
3. Se calendariza o_{ji*} en el descendiente de acuerdo a Φ_{ji}.

Paso 5. Se actualiza S como sigue:
1. Se remueve la operación o_{ji*} de S.
2. Se añade directamente el sucesor de la operación o_{ji*} en S.

Paso 6. Continuamos en el paso 2 hasta que generamos la calendarización completa.

En el paso 4.1, el conflicto se resuelve escogiendo una operación aleatoria, mientras que en el paso 4.2, el conflicto se resuelve dando prioridad a la operación que primero fue calendarizada en uno de los padres p_s, entre todas las operaciones en conflicto en G_{r*}. El padre p_s es seleccionado aleatoriamente con igual probabilidad. De esta manera la propuesta de Nakano y Yamada es esencialmente basada en la heurística de reglas de despacho y no un algoritmo Giffler y Thompson puro. En cada paso una operación es seleccionada para añadirse a la calendarización parcial de la descendencia y los conflictos entre operaciones son resueltos especificando prioridades en las operaciones de acuerdo a la calendarización de sus padres.

La figura 5-33 demuestra la selección de la operación en el paso 4.2 para el descendiente o "hijo", las operaciones o_{11}, o_{21}, o_{31}, y o_{32} son calendarizables. Las operaciones siguientes a calendarizar son S = $\{o_{12}, o_{22}, o_{33}\}$. Asumiendo que el conjunto de operaciones en conflicto son S = $\{o_{12}, o_{22}\}$, el padre p_1 es seleccionado, debido a que la operación o_{12} fue la primera en ser calendarizada en el padre p_1, entonces esta operación es calendarizada en el descendiente.

5.3.2 MUTACIÓN BASADA EN LA BÚSQUEDA EN LA VECINDAD

En algoritmos genéticos convencionales la mutación es un operador complementario, el cual solo es usado para producir pequeñas perturbaciones en los cromosomas para mantener la diversidad de la población. Cuando se diseña un algoritmo genético híbrido, un principio fundamental es hibridizar donde sea posible. La mutación que se explicará a continuación se diseñada con la técnica de búsqueda en el entrono de vecindad. Este tipo de mutación no es un operador complementario, sino que es usado para realizar una búsqueda intensa con el objetivo de encontrar una descendencia mejorada.

Muchas definiciones pueden ser consideradas para definir el entorno de vecindad de un cromosoma. Para la representación basada en operaciones, la vecindad de un cromosoma dado puede considerarse como un conjunto de cromosomas (schedules) derivados del cromosoma original por intercambio de posición de genes λ (genes no idénticos y seleccionados aleatoriamente). Un cromosoma es λ-optimo si es mejor que cualquier otro de la vecindad. Por ejemplo en un problema de cuatro trabajos cuatro máquinas, supóngase que los genes en las

posiciones 4, 8 y 12 son aleatoriamente seleccionados. Estos genes son (4 3 2) y sus posibles permutaciones son (3 4 2), (3 2 4), (2 3 4), (2 4 3) y (4 2 3). Las permutaciones de los genes junto con los genes remanentes que forman los cromosomas vecinos se muestran en la figura 5-34. Entonces evaluamos todos los cromosomas vecinos, y el mejor es seleccionado como descendiente de la mutación.

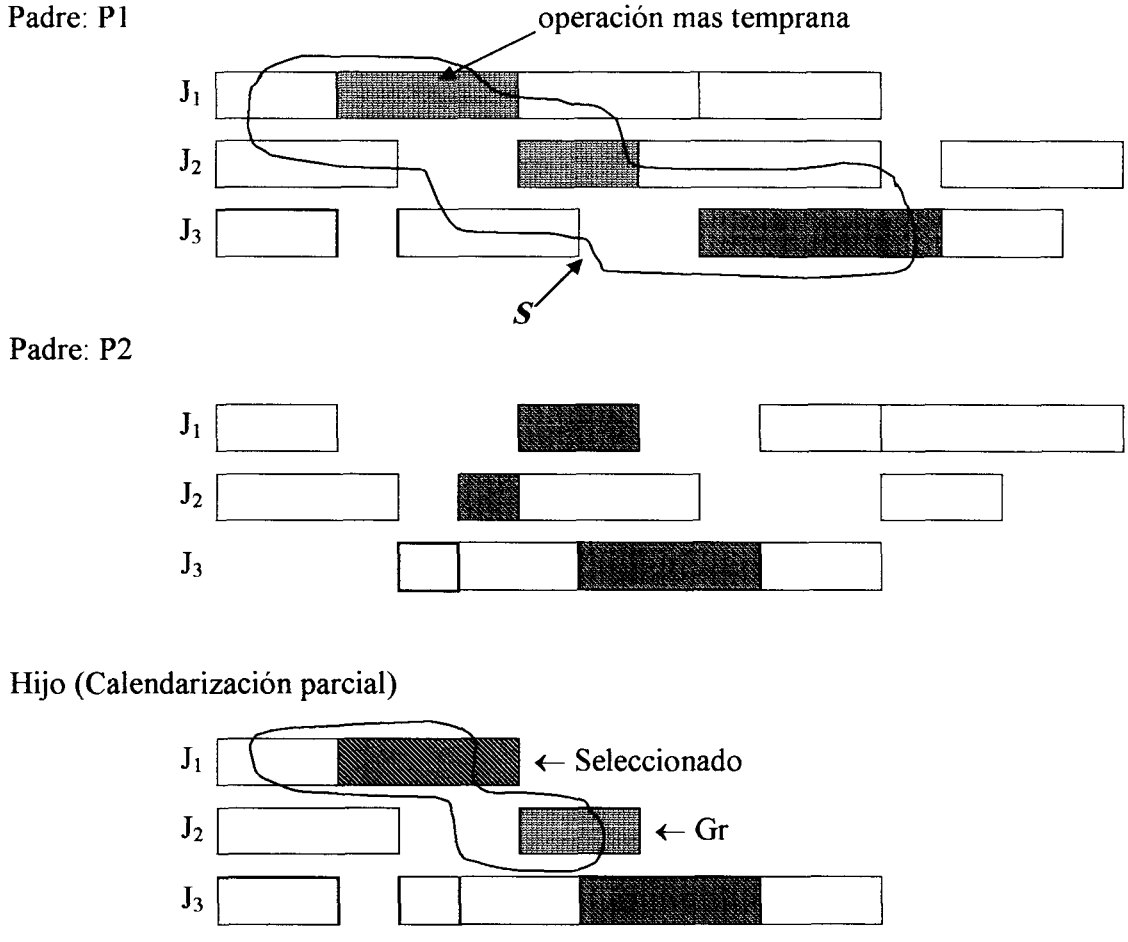


Figura 5-33. Cruzamiento basado en el algoritmo Giffler y Thompson

Cromosoma Padre

Cromosomas Vecinos

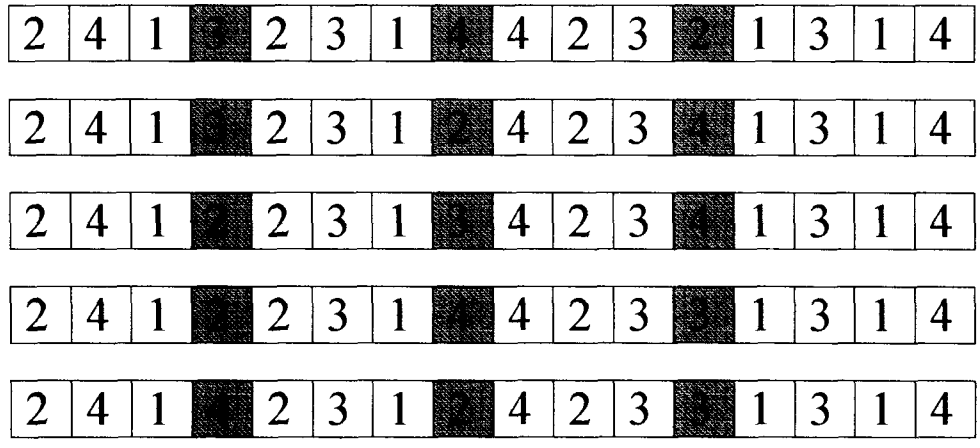


Figura 5-34. Secuencias vecinas

A continuación se muestra el pseudocodigo de la mutación basada en la búsqueda en la vecindad

Pseudocódigo: Mutación basada en la búsqueda en la vecindad

Begin
    i ← 0;
    while (i≤pop_size × pm) **do**
        escoger aleatoriamente un cromosoma que no se ha mutado;
        tomar aleatoriamente λ numero de genes diferentes del cromosoma;
        hacer vecinos en base a todas las permutaciones de los genes;
        evaluar la calendarización de todos los vecinos;
        seleccionar el mejor vecino como descendencia;
        i ← i + 1;
    end
end

6 MÉTODOS DE CALENDARIZACIÓN JOB SHOP CON ALGORITMOS GENÉTICOS

6.1 MÉTODO GEN, TSUJIMURA Y KUBOTA

Hay dos métodos diferentes para representar la calendarización como cromosoma: representación directa e indirecta.

En la representación indirecta la calendarización misma es usada como un individuo. Esta representación requiere un proceso de decodificación, pero necesita una complicada recombinación de operadores, lo cual podría asegurar que los individuos generados por esta representación sean viables.

La representación indirecta necesita un programa para traducirla en una secuencia valida, la ventaja de este esquema es la simplicidad de la estructura individual y de los operadores y el inconveniente es que el algoritmo genético esta restringido a realizar la búsqueda solo en el espacio de las posibles permutaciones de los genes individuales.

El método Gen, Tsujimura y Kubota utiliza la representación directa. Esta representación es una lista de operaciones, que contiene $N \times M$ genes, donde:

N: Número de trabajos
M: Número de Máquinas

Se asume que cada trabajo debe ser procesado sólo una vez en cada máquina, por esto es que cada trabajo solo aparece M veces en la lista de trabajos.

Por ejemplo; Un problema 4×4, esta descrito por las siguientes tablas:

Entonces la lista de trabajos (3,1,2,4,2,1,2,4,3,1,3,4,3,2,1,4) significa que la asignación de máquinas es (3,3,2,1,3,1,1,2,4,2,1,4,2,4,4,3)

Tabla 5-7

Trabajos	Ordenamiento de máquinas		Máquinas	Ordenamiento de Trabajos
j_1	3 1 2 4		m_1	4 1 2 3
j_2	2 3 1 4		m_2	2 4 1 3
j_3	3 4 1 2		m_3	3 1 2 4
j_4	1 2 4 3		m_4	3 4 2 1

6.1.1 FUNCIÓN DE EVALUACIÓN

La aptitud de cada individuo es evaluada como el tiempo total transcurrido de la calendarización correspondiente.

$$f(S_i) = \max_{1 \le j \le N} \{ft_{(j)}\}$$

Donde $ft_{(j)}$ es el tiempo de finalización del trabajo j.

6.1.2 CRUZAMIENTO

Los autores proponen un operador de intercambio de secuencia parcial. La secuencia parcial se identifica con el mismo trabajo en la primera y ultima posición del mismo. Por ejemplo se tienen dos cromosomas padres p_1, p_2.

Las secuencias parciales se seleccionan aleatoriamente, como sigue

1. Se escoge aleatoriamente una posición en el cromosoma padre p_1. Supóngase la sexta posición en la cual se localiza el trabajo 4.

2. Se busca el siguiente trabajo 4 más cercano. Ahora tenemos el secuencia parcial$_1$, definida como (4 1 2 4).

3. La siguiente secuencia parcial no se genera aleatoriamente, este debe empezar y terminar con el trabajo 4 como en la primera secuencia parcial.

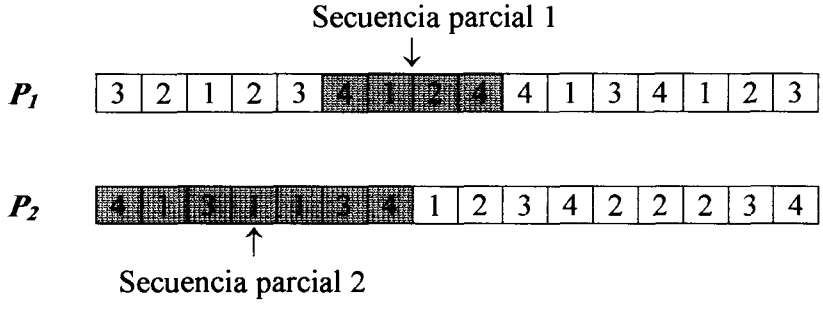


Figura 6-1. Selección de Secuencias parciales

Se hace el intercambio de secuencias parciales. Como se muestra en la figura 6-2. Usualmente las secuencias parciales contienen un número diferente de genes, por lo que los descendientes generados después del intercambio pueden ser ilegales

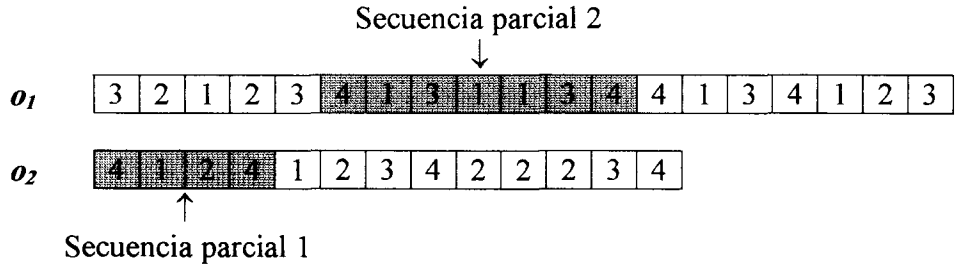
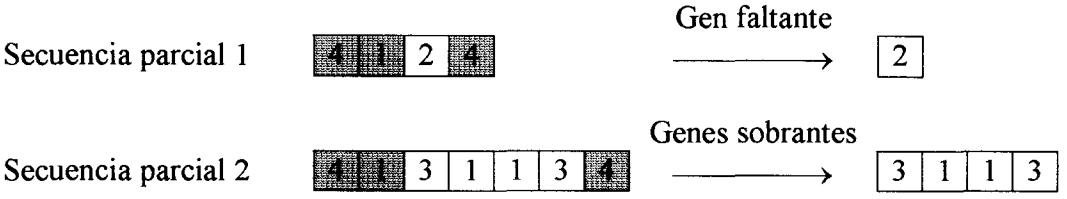


Figura 6-2. Intercambio de Secuencias parciales

El siguiente paso es legalizar la descendencia mediante la eliminación de genes excedentes o la adición de genes faltantes

Falta y exceso de genes para o_1:



Falta y exceso de genes para o_2:

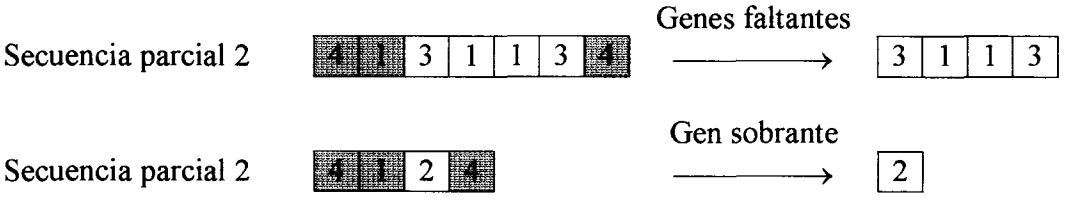


Figura 6-3. Falta y exceso de genes

El proceso para legalizar la descendencia se muestra en las siguientes figuras

(1) Eliminar los genes sobrantes 3113

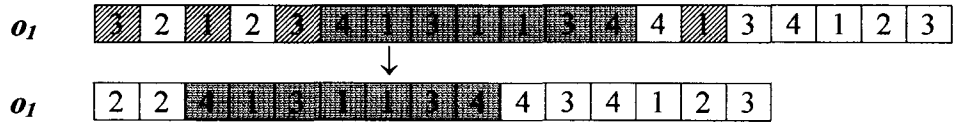


Figura 6-4. Proceso para legalizar la descendencia o_1 al eliminar los genes sobrantes

(2) Insertar el gen faltante 2

Figura 6-5. Proceso para legalizar la descendencia o_1 al insertar genes faltantes

(1) Eliminar el gen excedente 2

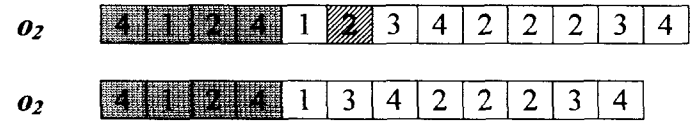


Figura 6-6. Proceso para legalizar la descendencia o_2 al eliminar los genes sobrantes

(2) Insertar los genes faltantes 3 1 1 3

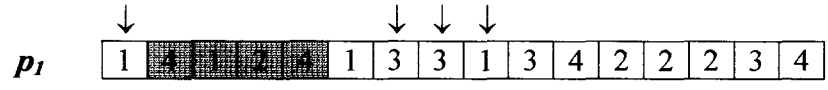


Figura 6-7. Proceso para legalizar la descendencia o_2 al insertar genes faltantes

6.1.3 MUTACIÓN

Aunque el cruzamiento trata de mejorar la aptitud de la descendencia, existe la posibilidad de que la descendencia converja en una solución local. Para reparar esta situación existe un mecanismo de escape, el cual es la mutación.

En este método el mecanismo de mutación se define como sigue: se generan aleatoriamente dos posiciones y se intercambian sus genes.

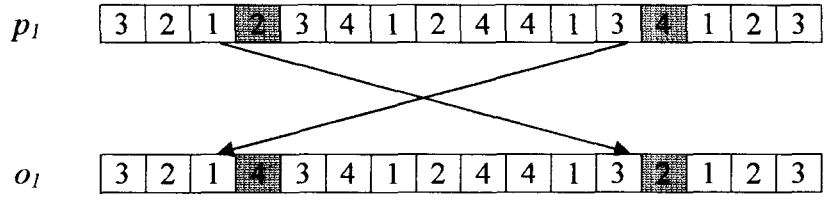


Figura 6-8 Mutación

6.1.4 CONSERVACIÓN DEL MEJOR INDIVIDUO

En cada generación se mejora la aptitud de la población, sin embargo los operadores genéticos pueden destruir al mejor individuo que ha aparecido hasta el momento. Para eliminar este inconveniente, el mejor individuo de cada generación se mantiene, para ser comparado con los individuos de la siguiente generación.

6.1.5 EJEMPLO

Mostrar el método Gen, Tsujimura y Kubota para el problema mostrado enseguida

Tabla 3-1

	Operaciones (máquina, tiempo proc.)		
Trabajos	1	2	3
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

En el caso del cromosoma, [3, 2, 2, 1, 1, 2, 3, 1, 3], el primer gen "3" representa la operación 1 del trabajo 3. El segundo gen "2" representa la operación 1 del trabajo 2. El tercer gen "2" representa la operación 2 del trabajo 2.

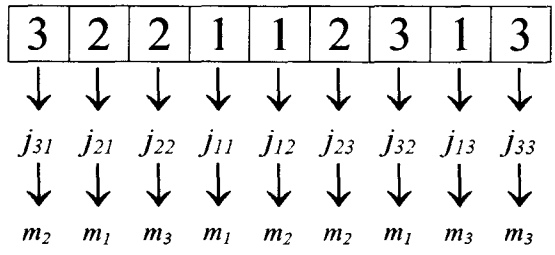


Figura 4-1. Representación basada en operaciones

6.1.5.1 Algoritmo

Paso 0: Se establecen los parámetros

Se establece la población como un total de 4 individuos, es decir *pop-size* = 4

Se establece p_m como el índice de mutación, $p_m = 0.25$

Se establece p_c como el índice de cruzamiento, $p_c = 0.25$

Paso 1: Se genera aleatoriamente la población inicial

Se generan tantos individuos como se haya establecido en el parámetro *pop-size*. Los individuos deben contener M × N genes, y cada trabajo debe aparecer exactamente M veces

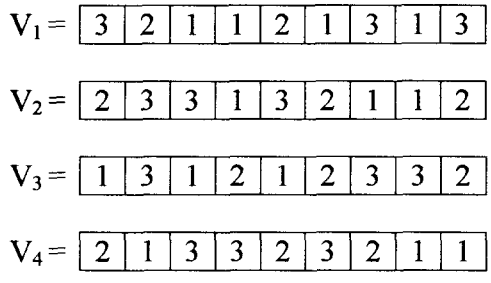


Figura 6-9. Población inicial (ejemplo)

Paso 2: Se calcula la función evaluación y se conserva al mejor individuo

Las siguientes gráficas de Gantt corresponden a cada individuo de la población e indican el tiempo requerido para completar todos los trabajos (makespan) de cada individuo

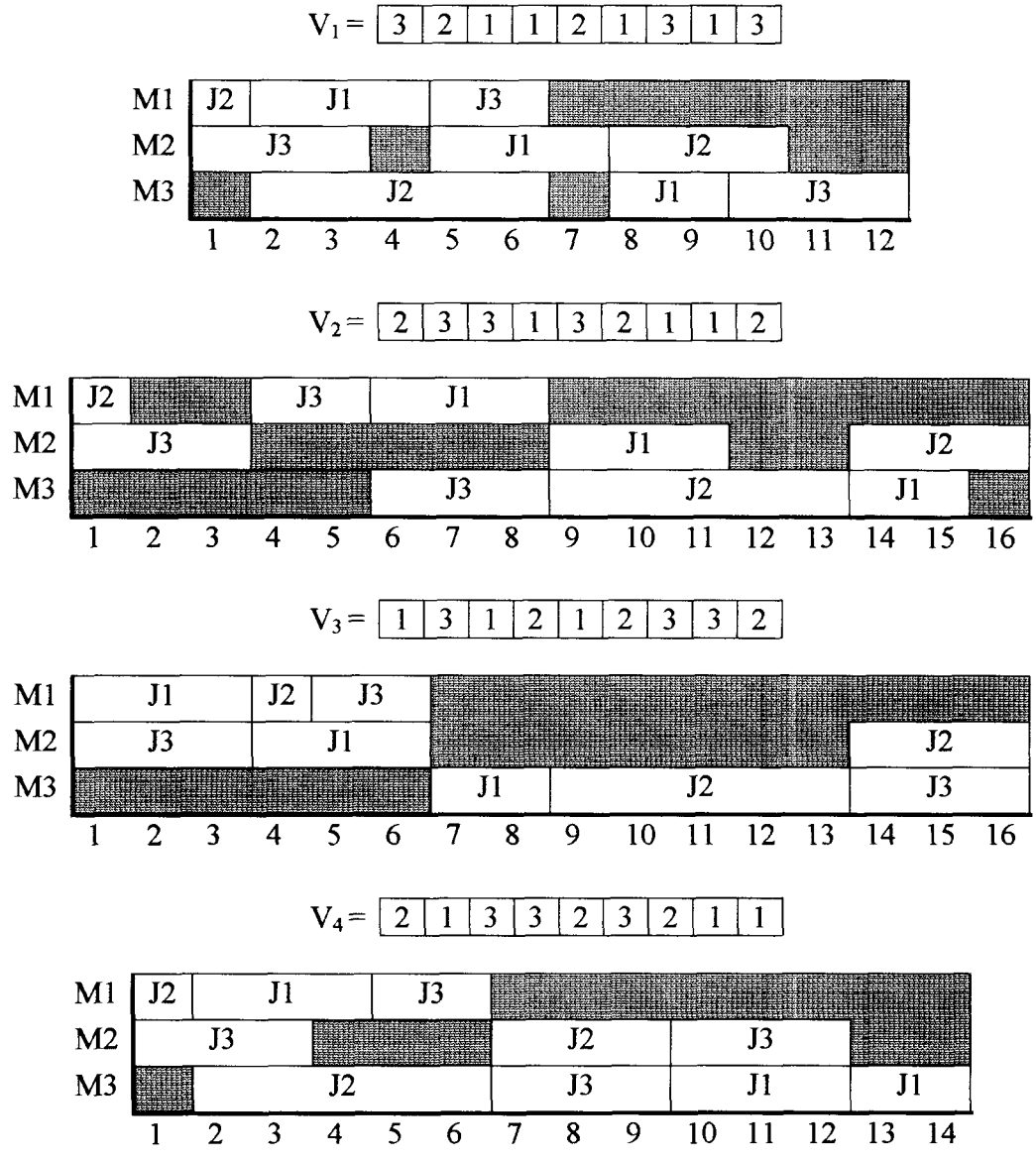


Figura 6-10. Evaluación de la población (ejemplo).

La población se evalúa, en base a la función objetivo, es decir se obtiene el tiempo total en que se realizan todos los trabajos. La evaluación de la población se obtuvo al generar los diagramas de Gantt de cada individuo.

$$f(S_i) = \max_{1 \le j \le N} \{ft_{(j)}\}$$

Los resultados de la evaluación son los siguientes:

$$\text{eval}(V_1) = 12$$

$$\text{eval}(V_2) = 16$$

$$\text{eval}(V_3) = 16$$

$$\text{eval}(V_4) = 14$$

El mejor individuo es el primer cromosoma, este se conserva como el mejor individuo de la población

V_1 =	3	2	1	1	2	1	3	1	3	Mejor individuo S*
V_2 =	2	3	3	1	3	2	1	1	2	
V_3 =	1	3	1	2	1	2	3	3	2	
V_4 =	2	1	3	3	2	3	2	1	1	

Figura 6-11. Mejor individuo (ejemplo).

Paso 3: Cruzamiento

El cruzamiento se realiza tantas veces como nos indique el producto entre el índice de cruzamiento p_c y el tamaño de población *pop-size*

$$pop\_size \times \boldsymbol{p_c} = 4 \times 0.25 = 1$$

A continuación se selecciona un par de cromosomas, se identifican dos secuencias parciales, se intercambian estas y se generan dos nuevos individuos, y se legaliza la descendencia

Retomando el ejemplo, se seleccionan aleatoriamente dos cromosomas, en este caso V_2 y V_3 y en cada uno de estos se seleccionan secuencias parciales.

Las secuencias parciales se seleccionan aleatoriamente, como sigue

1. Se escoge aleatoriamente una posición en el cromosoma padre V_2. Supóngase la cuarta posición en la cual se localiza el trabajo 1.

2. Se busca el siguiente trabajo 1 más cercano. Ahora tenemos el secuencia parcial 1, definida como (1 3 2 1).
3. La secuencia parcial en el cromosoma V_3 no se genera aleatoriamente, esta debe empezar y terminar con el trabajo 1 como en la primer secuencia parcial.

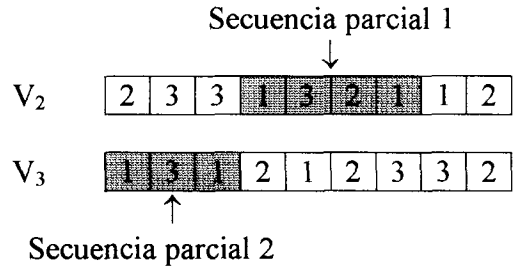


Figura 6-12. Generación de secuencias parciales (ejemplo).

Se intercambian los Secuencias parciales:

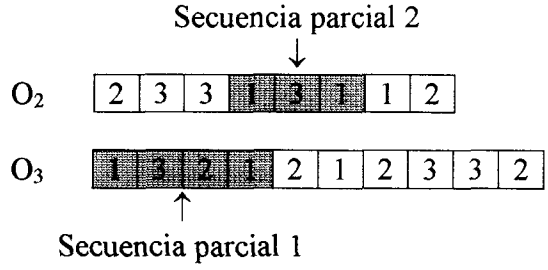
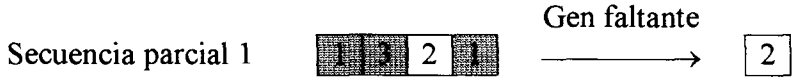


Figura 6-13. Intercambio de secuencias parciales

Proceso para legalizar la descendencia

Genes faltantes O_2:



Genes en exceso para O_2:

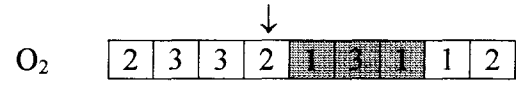
Figura 6-14. Genes faltantes y en exceso para O_2 (ejemplo).

Genes faltantes para O_3:

Genes en exceso para O_3:

Figura 6-15. Genes faltantes y en exceso para O_3 (ejemplo).

Insertar el gen faltante 2



Eliminar el gen en exceso 2

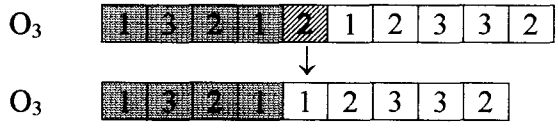


Figura 6-16. Legalización de la población (ejemplo).

Lo anterior quedaría resumido en pseudocódigo, como sigue:

Begin
  $i = 0$;
    while $i < pop\_size \times p_c$ **do**
      Se selecciona un par de cromosomas, se identifican dos secuencias; parciales se intercambian estas y se generan dos nuevos individuos;
      Se legaliza la descendencia;
      $i = i + 2$;
    end
end

Paso 4: Mutación

Se selecciona un individuo, se escogen dos posiciones del cromosoma, y se intercambian sus genes.

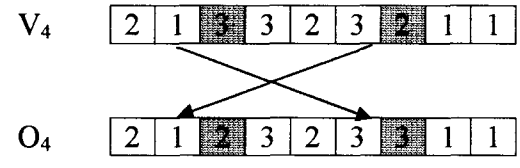


Figura 6-17. Mutación (ejemplo).

El proceso de mutación se describe en pseudocódigo como sigue:

Begin
  $i = 0$;
    while $i < pop\_size \times p_m$ **do**
      escoger dos posiciones del cromosoma, e intercambiar sus genes;
      $i = i + 1$;
    end
end

Después del cruzamiento y de la mutación, la población crece en un rango de $(1+p_c)\times(1+p_m) \times$ *pop_size*.

La nueva población es la siguiente.

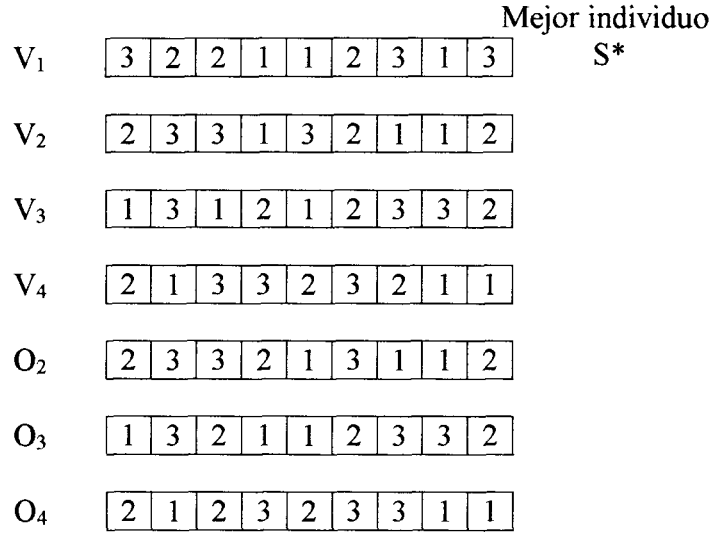


Figura 6-18. Nueva población

Paso 5. Selección

El método de selección es el de *Selección elitista*, es decir se seleccionan los mejores individuos de la nueva población, en base a la función aptitud. El número de individuos que se seleccionan debe ser igual al tamaño de la población establecido desde el Paso 0.

Para realizar el proceso de Selección, previamente se deben evaluar los individuos generados o descendientes. Esto se realiza a continuación

La evaluación de la población actual es la siguiente

$$\text{eval}(V_1) = 12$$

$$\text{eval}(V_2) = 16$$

$$\text{eval}(V_3) = 16$$

$$\text{eval}(V_4) = 14$$

$$\text{eval}(O_2) = 14$$

$$\text{eval}(O_3) = 16$$

$$\text{eval}(O_4) = 14$$

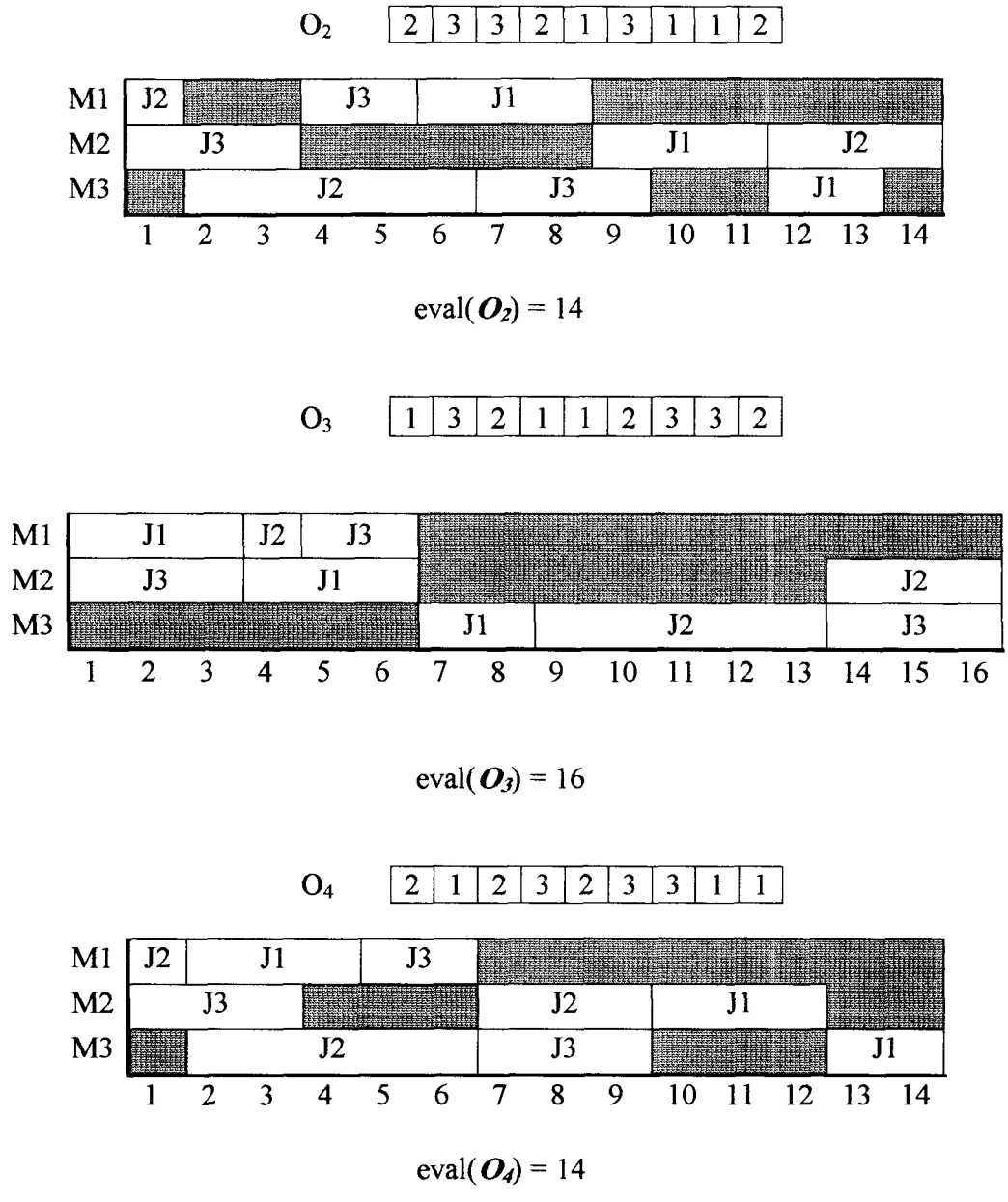


Figura 6-19. Evaluación de la nueva población (ejemplo).

Una vez conocida la evolución de la población actual se procede a realizar la Selección. Entonces se mantiene el mejor cromosoma de la población inicial que es el V1, y se seleccionan los individuos que tengan la mejor evaluación, es decir el menor tiempo requerido para completar todos los trabajos.

Los individuos seleccionados serian los siguientes:

V_1	3	2	2	1	1	2	3	1	3
V_4	2	1	3	3	2	3	2	1	1
O_2	2	3	3	2	1	3	1	1	2
O_4	2	1	2	3	2	3	3	1	1

Figura 6-20. Individuos seleccionados (ejemplo)

$$\text{eval}(V_1) = 12$$
$$\text{eval}(V_4) = 14$$
$$\text{eval}(O_2) = 14$$
$$\text{eval}(O_4) = 14$$

Con esto se concluye una generación, manteniendo a V1 como la mejor secuencia y en este caso, nuestro resultado óptimo.

Para continuar el proceso se inicia otro ciclo en el paso 2, y así sucesivamente se van completando iteraciones hasta cumplir con el máximo número de generaciones, o que el algoritmo converja al resultado óptimo.

6.2 MÉTODO SHI, IIMA, SANNOMIYA

Este método fue diseñado para la búsqueda en un espacio de calendarización semiactivo. Además en este método se introduce un cruzamiento diferente, el cruzamiento por conjuntos o bloques (SPX), acompañado de una búsqueda genética. Se establecen también una estrategia de selección y una estructura de producción de individuos, que evitan la producción de generaciones que no sean factibles [26].

6.2.1 REPRESENTACIÓN

En este método se usa la representación basada en operaciones. Esta representación codifica la calendarización como una secuencia de operaciones y cada gen representa una operación.

Utilizando el ejemplo de tres máquinas, tres trabajos representados en la siguiente tabla

Tabla 3-1

Trabajos	Operaciones (máquina, tiempo proc.)		
j_1	1,3	2,3	3,2
j_2	1,1	3,5	2,3
j_3	2,3	1,2	3,3

La siguiente figura muestra gráficamente la representación basada en operaciones

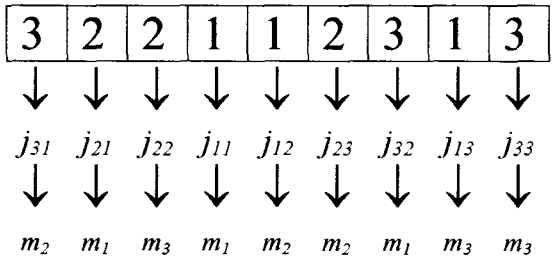


Figura 4-1. Representación basada en operaciones

6.2.2 ESQUEMA DE SELECCIÓN

En este método de selección se utiliza la estrategia "selección 2/4". En esta estrategia se seleccionan aleatoriamente dos padres con diferente makespan, siendo estos de la población anterior. Estos dos padres se pasan por el esquema de "producción genética", para generar dos hijos, entonces se seleccionan los mejores dos individuos de los cuatro candidatos incluyendo los padres y la descendencia generada, entonces se mantienen los dos individuos seleccionados por elitismo, para formar parte de la siguiente generación remplazando a los padres de la generación anterior

Esta estrategia de selección continúa hasta que se completa la población de la siguiente generación. Este esquema de selección es llamado "selección 2/4", el cual significa que los dos mejores individuos son seleccionados de cuatro candidatos. A través de la "selección 2/4" los individuos inferiores son eliminados para el nacimiento de nuevos individuos superiores. Por lo tanto el mínimo makespan y el máximo makespan de la población van en decremento conforme avanzan las generaciones.

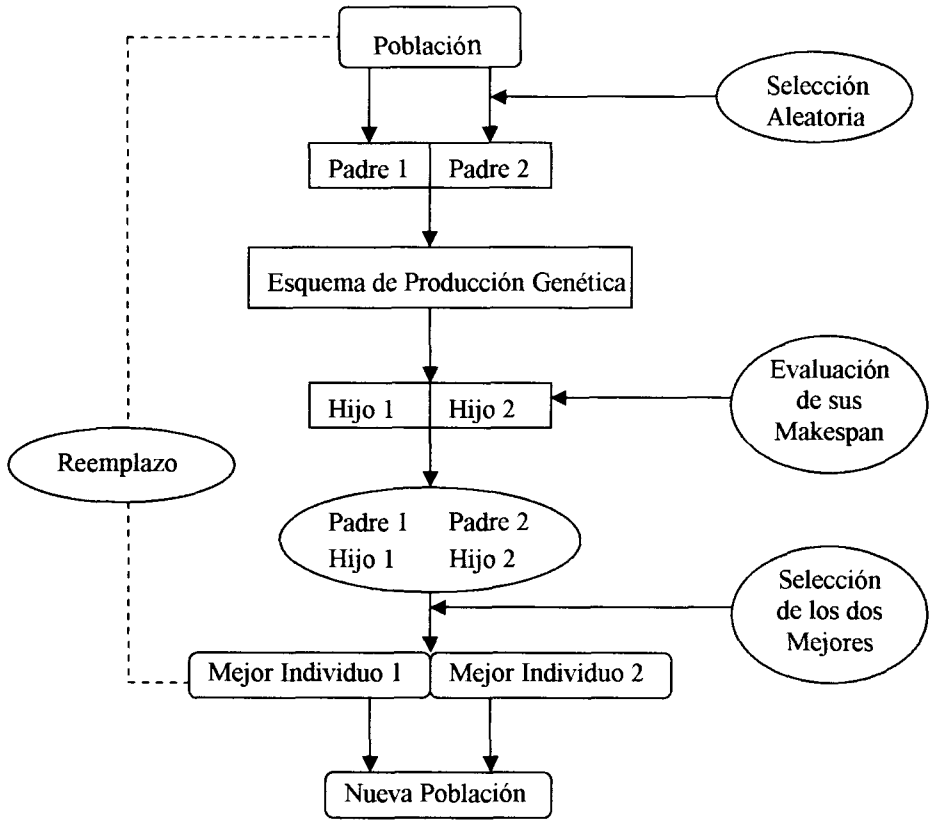


Figura 6-21. Esquema de selección

6.2.3 ESQUEMA DE PRODUCCIÓN GENÉTICA

La "Producción Genética" en el esquema de selección consiste en el cruzamiento y la mutación. El esquema de producción es ilustrado en la figura 6-22

Como se puede observar en la figura la mutación se realiza en dos partes. Una parte se realiza dentro del cruzamiento, con probabilidad Pm; la otra parte es independiente del cruzamiento. Si la tasa de cruzamiento es Pc, la tasa de la mutación independiente es 1-Pc. La mutación trabaja puramente con una búsqueda genética, y específicamente como búsqueda en la vecindad, mientras que la mutación que esta dentro del cruzamiento, trabaja para que el cruzamiento explore un campo más amplio de búsqueda. Las probabilidades Pc y Pm son ajustables interactivamente. Por la construcción de este esquema, todo cromosoma Padre debe pasar por lo menos por una operación genética.

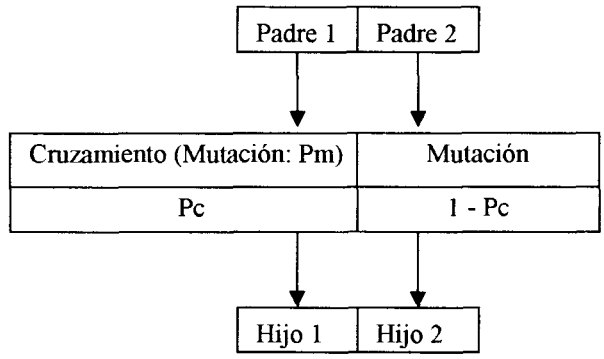


Figura 6-22. Esquema de Producción Genética

6.2.4 MUTACIÓN.

Las dos partes de la mutación usan el mismo procedimiento, en cual dos operaciones están involucradas. Una de estas es el Intercambio o "Swap", el cual aleatoriamente intercambia dos números distintos de la cadena [JxM]. El otro es el intercambio segmentado o "seg-swap", el cual intercambia dos segmentos de la cadena, aleatoriamente seleccionados. Las dos operaciones de intercambio se realizan alternadamente de manera aleatoria con igual probabilidad de realización.

El cruzamiento es una de las operaciones principales en cualquier Algoritmo Genético

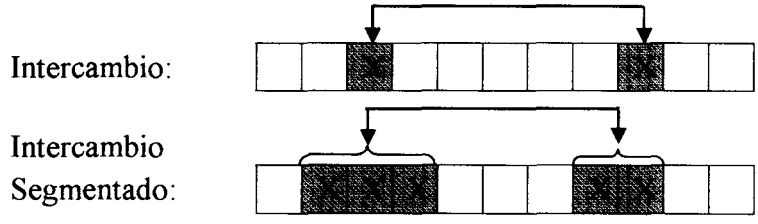


Figura 6-23. Operaciones de Mutación

6.2.5 CRUZAMIENTO

En este método se diseño un nuevo cruzamiento, que se le nombra cruzamiento por conjuntos o bloques, o Set- Partition Crossover (SPX).

6.2.5.1 Cruzamiento SPX

El cruzamiento se realiza a partir de la cadena [JxM] la cual es una colección de números de trabajos, donde cada numero de trabajo se repite M veces. Primero se divide aleatoriamente el conjunto de números de trabajos, {1, 2,..., J}, en dos subgrupos J1 y J2, entonces se combinan los números del Padre 1, que están en J1 y los números del Padre 2 que están en J2. El orden de la combinación es en forma entrelazada, es decir uno por uno de arriba hacia abajo y de izquierda a derecha, esto se describe de una manera precisa por el siguiente pseudocódigo

```
j = 1;
From (i = 1)
        If      Padre1(i) ∈ J1 Then
                Hijo(j) = Padre1(i);
                J = j + 1;
        If      Padre2(i) ∈ J2 Then
                Hijo(j) = Padre2(i);
                j = j + 1;
        i = i + 1;
Until (i< j × M)
```

Pseudo código para el cruzamiento SPX

Esta parte del procedimiento crea una nueva cadena. Lo siguiente es intercambiar los dos subgrupos J_1 y J_2, y hacer la combinación otra vez para formar otra cadena nueva. Después de este procedimiento dos descendientes habrán nacido. En la siguiente figura se muestra un ejemplo del cruzamiento SPX

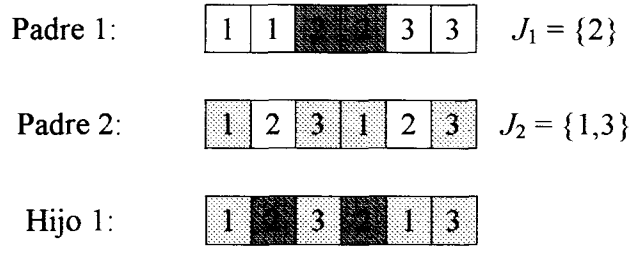


Figura 6-24. Ejemplo del cruzamiento SPT ($J = 3, M = 2$).

7 IMPLEMENTACIÓN DEL ALGORITMO GENÉTICO

El método en el que se basó el trabajo para la implementación del algoritmo genético fue el método Shi, Iima, Sannomiya. Este método fue diseñado para la búsqueda en un espacio de calendarización semiactivo. La representación de la calendarización es por operaciones, la cual codifica al cromosoma como una secuencia de operaciones, donde cada gen representa una operación. Además en esté método se introduce un cruzamiento diferente, el cruzamiento por conjuntos o bloques (SPX), acompañado de una búsqueda genética. Se establecen también una estrategia de selección y una estructura de producción de individuos, que evitan la creación de generaciones no factibles.

7.1 DESARROLLO DEL PROGRAMA DEL ALGORITMO GENÉTICO

El desarrollo del programa del algoritmo genético se planteó que fuera dinámico para poder calendarizar problemas de n trabajos y m máquinas, con la finalidad de generar un programa flexible capaz de poder resolver problemas de dimensiones rectangulares, es decir que el número de trabajos sea diferente al número de máquinas, y así extender el espacio de problemas a resolver.

Para el desarrollo del algoritmo genético se uso el lenguaje C porque es un programa estructurado capaz de manejar memoria en tiempo de ejecución, maneja con facilidad punteros, estructuras y arreglos dinámicos, que son elementos esenciales en el desarrollo del programa.

Se planteó el diseño de una aplicación fácil para el usuario, por lo que el programa del algoritmo genético toma los datos de entrada de un archivo de texto, los cuales son número de máquinas, número de trabajos, tamaño de la población, tasa de cruzamiento, tasa de mutación, y máximo número de generaciones, con esto se inicia el algoritmo genético, una vez que se ejecuto el algoritmo de manera satisfactoria el programa genera un archivo de texto de salida que contiene la solución de la calendarización, esto es la definición de tiempos de inicio y termino de cada una de las operaciones que forman parte del problema.

7.2 DESARROLLO GENERAL DE LA APLICACION

Una vez instalado el programa CAJSAG (Calendarización JobShop con Algoritmos Genéticos), para ingresar al sistema, vaya a menú inicio, aparecerá una opción que dice calendarización JobShop con algorimos genéticos, al dar clic entrará al sistema para iniciar su uso.

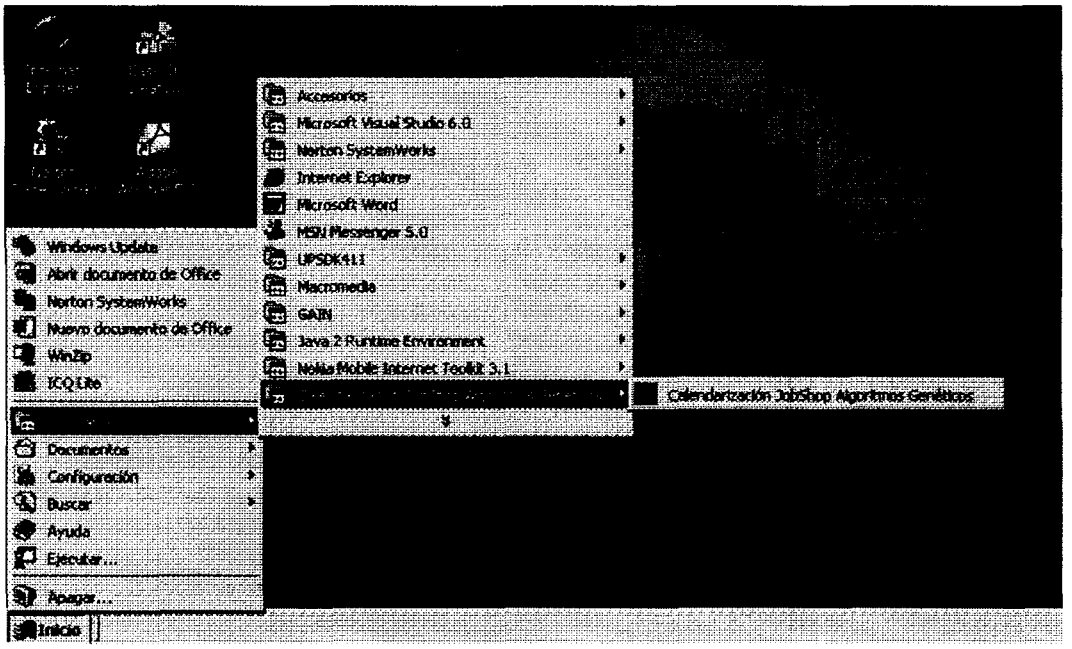


Figura 7-1. Acceso al sistema

El sistema muestra una pantalla de bienvenida donde aparece el titulo del sistema, el desarrollador y la versión

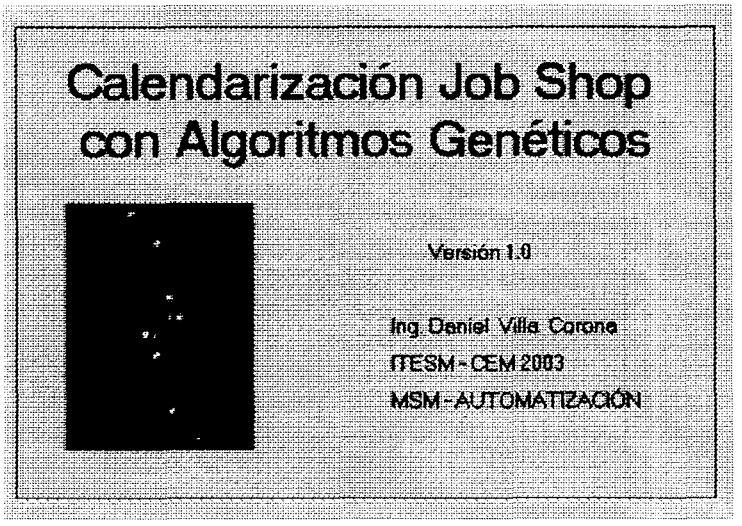


Figura 7-2. Pantalla de bienvenida

Posteriormente aparece la pantalla principal del sistema, esta cuenta con una barra de menú para realizar diversas operaciones.

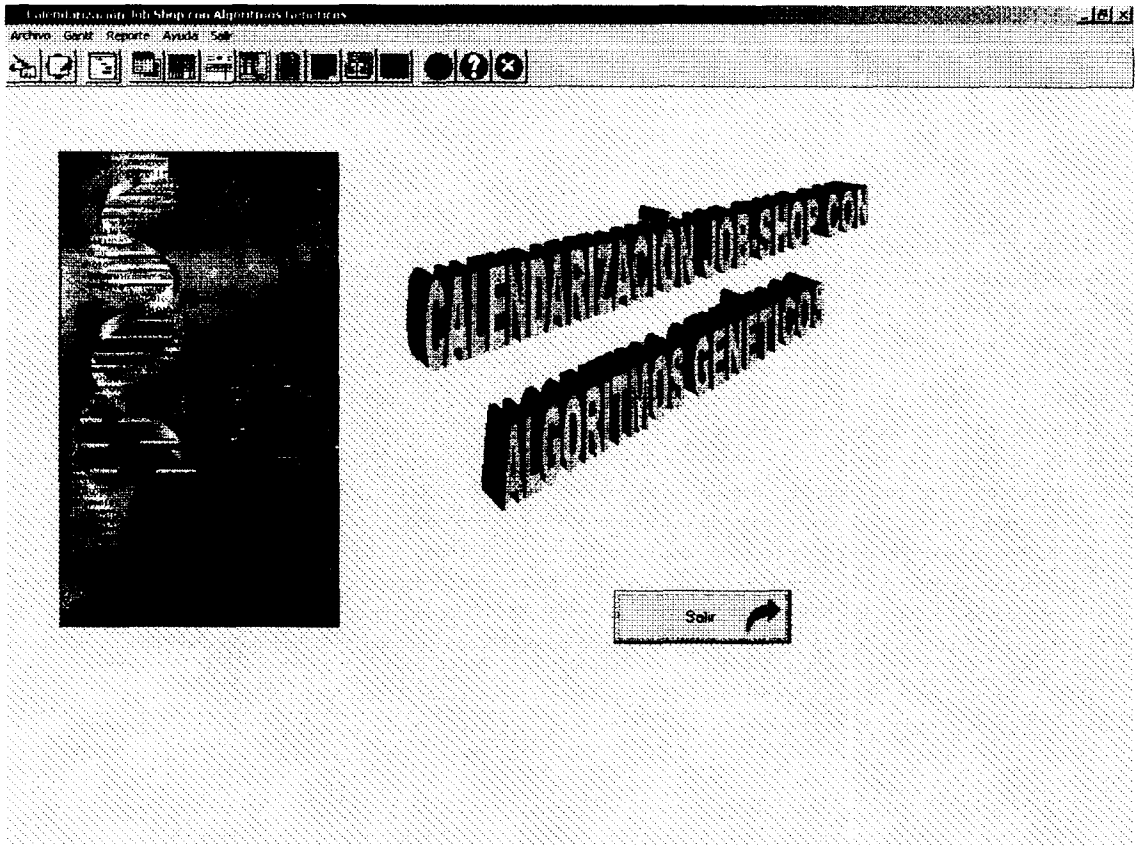


Figura 7-3. Pantalla Principal

La barra de menú de la pantalla principal es la siguiente: cuenta con cinco opciones que presentan un acceso rápido mediante un icono correspondiente.

Figura 7-4. Barra de menú de pantalla principal

Los parámetros de inicio necesarios para empezar a utilizar el sistema, se cargan a partir del menú Archivo y la opción parámetros.

Figura 7-5. Opción parámetros de barra de menú.

También puede entrar a esta opción mediante el acceso rápido, que es el primer icono de la barra de menú.

Figura 7-6. Opción parámetros de barra de iconos.

Al entrar en la opción parámetros, de cualquiera de las dos formas mencionadas (mediante barra de menú o icono), se mostrará una pantalla de ingreso de parámetros.

Figura 7-7. Parámetros de entrada

Al ingresar el número de trabajos, el número de máquinas y al hacer clic en guardar, se despliega la siguiente pantalla con parámetros recomendados para el algoritmo genético

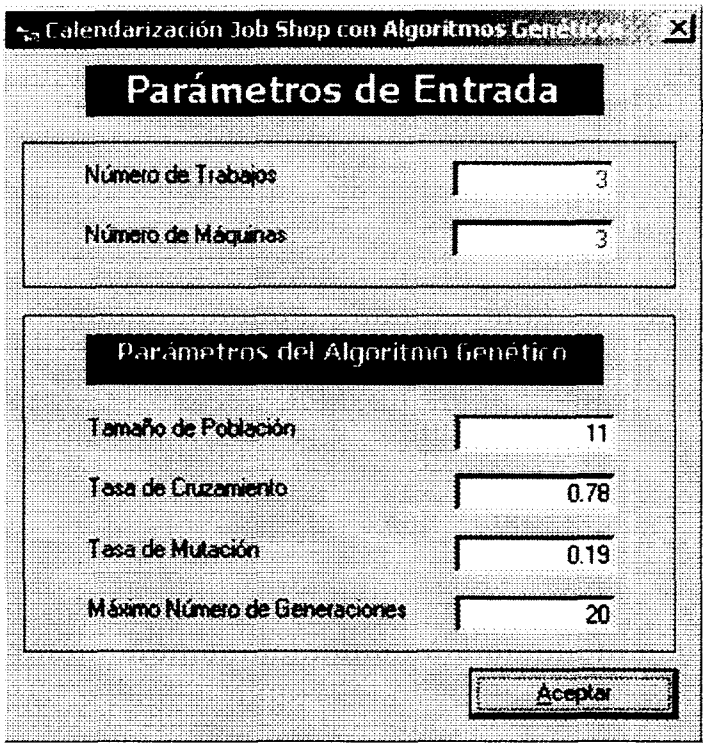


Figura 7-8. Captura de parámetros de entrada

Los parámetros recomendados son una sugerencia, estos se pueden cambiar de acuerdo a la calendarización que desee obtener.

Al llenar los campos con datos válidos y dar clic en aceptar podrá realizar otras operaciones que tendrán como entrada los datos que ha ingresado.

La matriz de operaciones necesaria para empezar a utilizar el sistema, se cargan a partir del menú Archivo y la opción matriz de operaciones, igualmente cuenta con un icono de acceso rápido que realiza la misma función.

Figura 7-9. Opción matriz de operaciones

Al dar clic en esta opción aparecerá la siguiente pantalla. Puede proseguir llenando la matriz con datos validos, en la cuadricula que se presenta, las filas representan los trabajos, las columnas numeradas representan las operaciones que están dadas cada una por dos columnas, donde la

primera representa la máquina que realizará la operación y la segunda el tiempo de procesamiento.

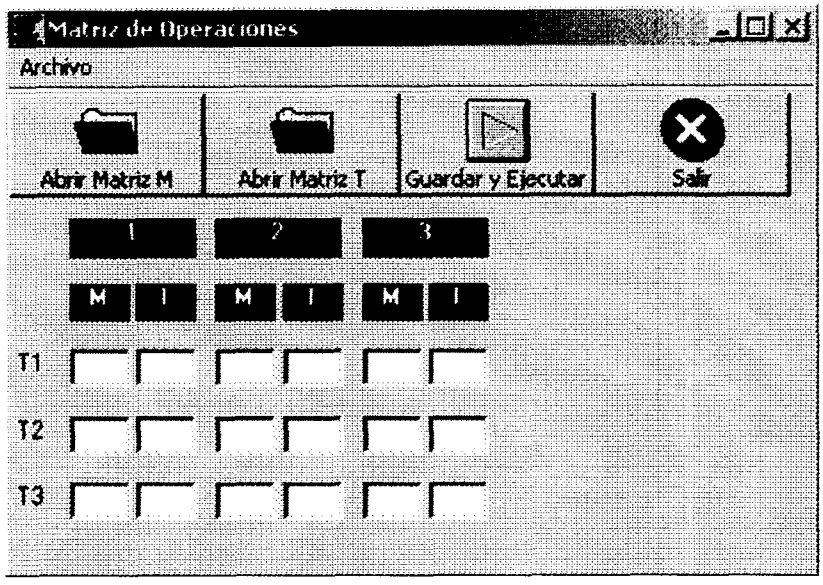


Figura 7-10. Matriz de Operaciones

Puede llenar los datos de manera manual, para casos de un gran número de maquinas y trabajos, el sistema cuenta con una opción de inserción automática, en la cual puede cargar un archivo de entrada (con extensión .txt), que contenga la matrizm de maquinas y otro archivo que contenga la matrizt de tiempos de procesamiento. Esta opción se encuentra en menú archivo, opción abrir y opción ya sea matrizm o matrizt.

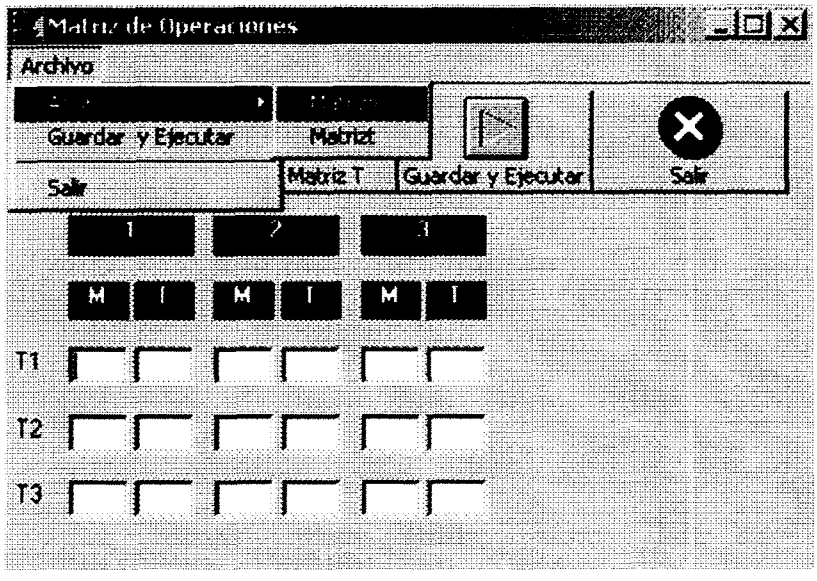


Figura 7-11. Inserción automática

Al dar clic aparecerá un cuadro de diálogo para buscar el archivo que contiene la matriz, esta pantalla es similar para ambos casos tanto matrizm como matrizt. De igual forma para abrir los archivos de matrices de entrada, existe una opción rápida en forma de iconos, debajo de la barra de menú.

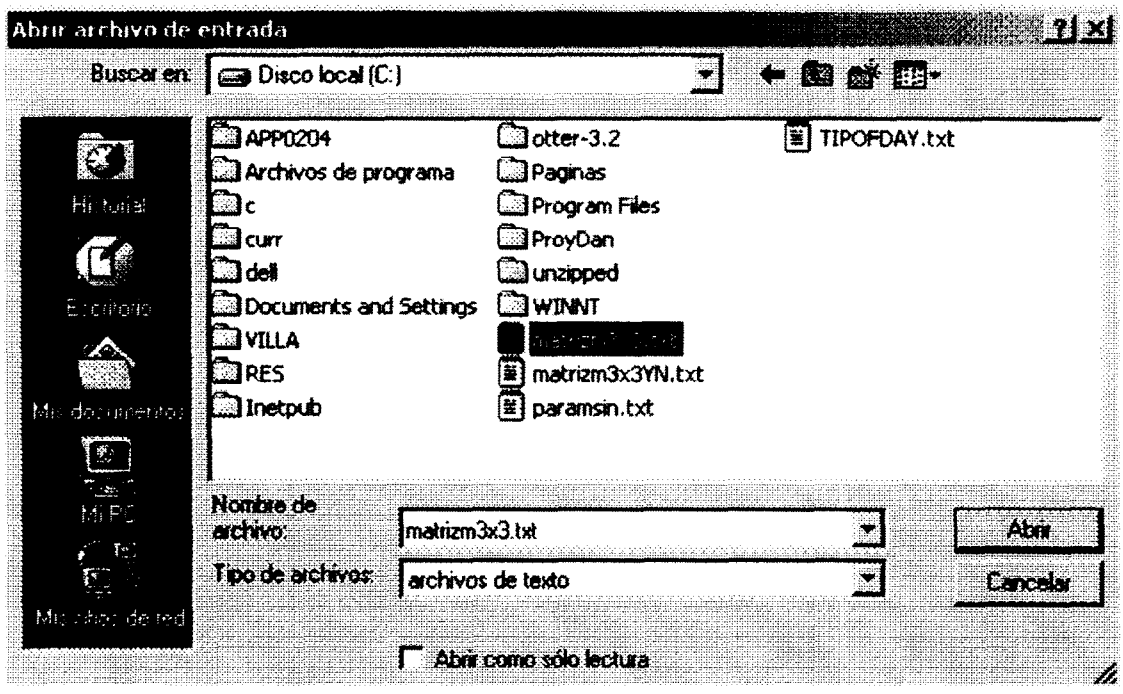


Figura 7-12. Cuadro de dialogo para elegir archivo.

Una vez que se lleno la matriz de operaciones con datos válidos se tienen que guardar estos datos dando clic en el menú Archivo -> Guardar.

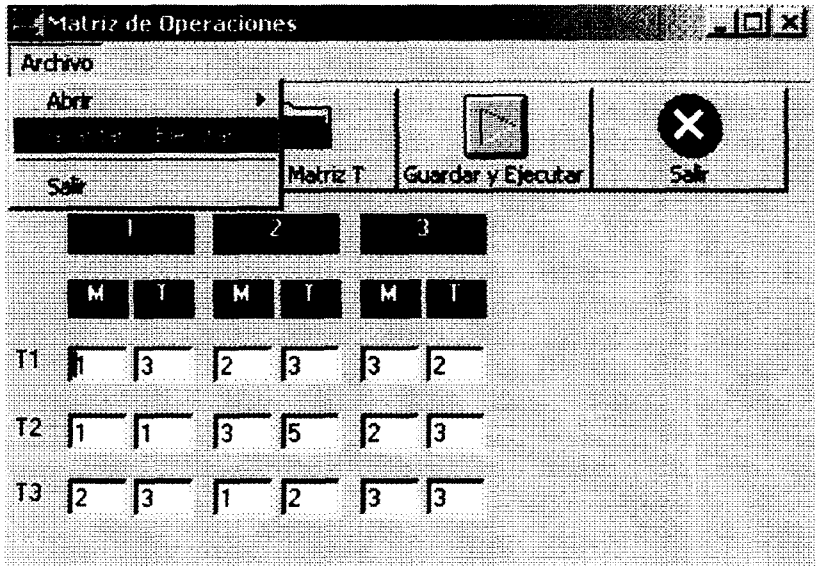


Figura 7-13. Matriz de operaciones con datos

Aparecerá un mensaje que le indica que va a realizarse el algoritmo genético e informa que la operación puede tardar en ejecutarse, esto depende del tamaño del problema a resolver y del número de generaciones que se halla indicado al algoritmo genético.

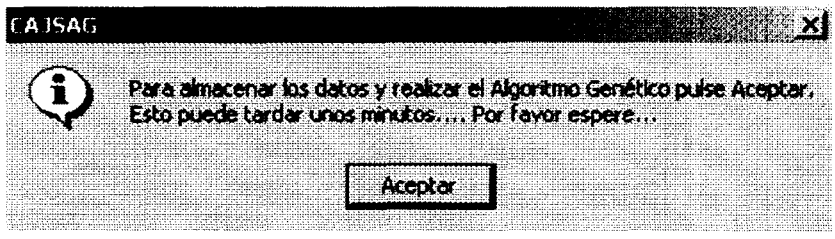


Figura 7-14. Mensaje de información.

Cuando se termina de realizar el algoritmo genético el sistema mostrará el siguiente mensaje.

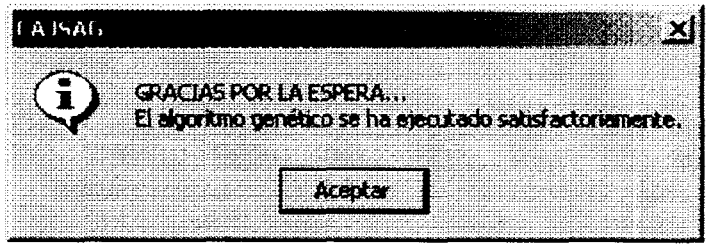


Figura 7-15. Mensaje de finalización del algoritmo

La gráfica de gantt aparece automáticamente cuando hace clic en aceptar en cuadro de diálogo anterior

Una vez que se realizó el algoritmo genético, puede ver la gráfica de gantt mediante el menú gantt -> Graficar o mediante el icono de gráfica de gantt para acceso rápido.

Figura 7-16. Menú Graficar

Al dar clic en la opción anterior se mostrará la gráfica de Gantt donde se especifican los tiempos de inicio y termino de cada operación, además cada operación posee una etiqueta que muestra su identificación por medio del índice *jo* (trabajo, operación), también el conjunto de operaciones que conforman un trabajo se pintan con el mismo color, esto para hacer mas comprensible el diagrama de Gantt.

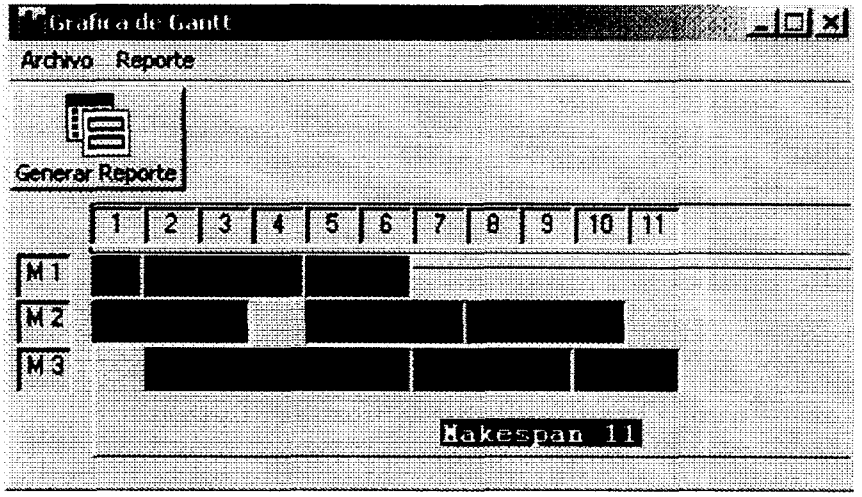


Figura 7-17. Gráfica de Gantt

El sistema brinda la posibilidad de ver un reporte completo, así como reportes individuales, para ver el reporte completo debe elegir el menú Reporte -> Reporte Completo, también puede verlo dando clic en el cuarto icono de izquierda a derecha para acceso rápido.

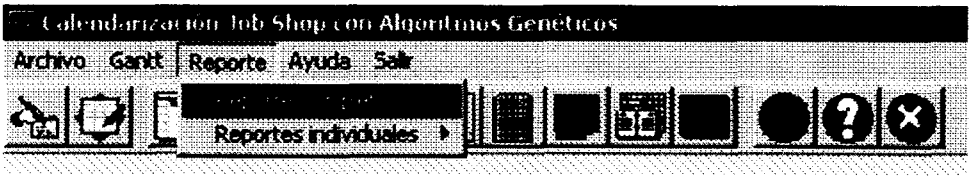


Figura 7-18. Menú Reporte Completo

El reporte completo muestra todos los resultados obtenidos mediante el algoritmo genético, que se identifican por pestañas en la siguiente forma. La primera pestaña muestra el Tiempo requerido para completar todos los trabajos o CMax.

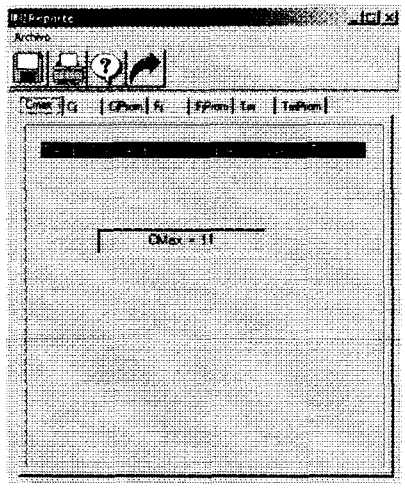


Figura 7-19. Reporte Completo Cmax

La segunda pestaña es Tiempo de terminación de cada trabajo, también se puede acceder a esta opción mediante menú Reporte -> Reportes individuales -> Cj.

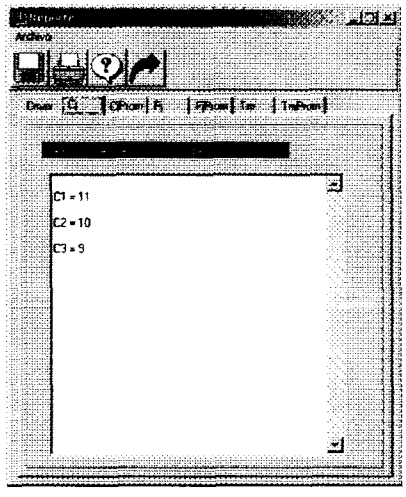


Figura 7-20. Tiempo de terminación de cada Trabajo

La tercera pestaña es Tiempo promedio de terminación de cada trabajo, también se puede acceder a esta opción mediante menú Reporte -> Reportes individuales -> CjProm.

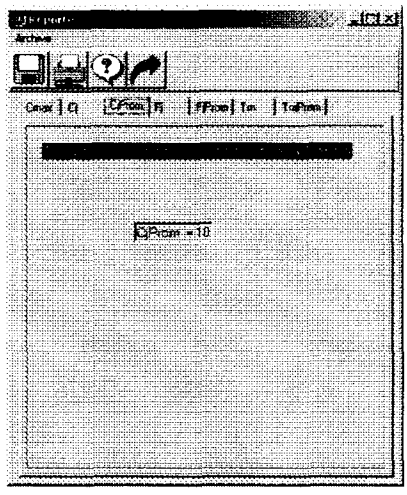


Figura 7-21. Tiempo Promedio de Terminación de cada Trabajo

La cuarta pestaña es Tiempo de Flujo de cada Trabajo, también se puede acceder a esta opción mediante menú Reporte -> Reportes individuales -> Fj.

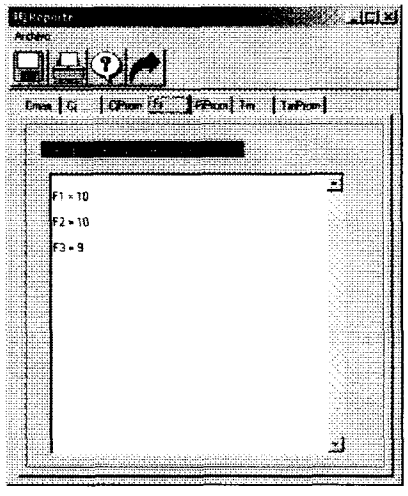


Figura 7-22. Tiempo de Flujo de cada Trabajo

La quinta pestaña es Tiempo Promedio de Flujo de Trabajos, también se puede acceder a esta opción mediante menú Reporte -> Reportes individuales -> FjProm.

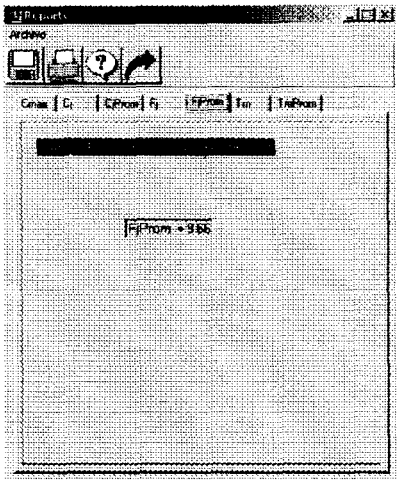


Figura 7-23. Tiempo Promedio de Flujo de Trabajos

La sexta pestaña es Tiempo de Terminación de cada máquina, también se puede acceder a esta opción mediante menú Reporte -> Reportes individuales -> Tm.

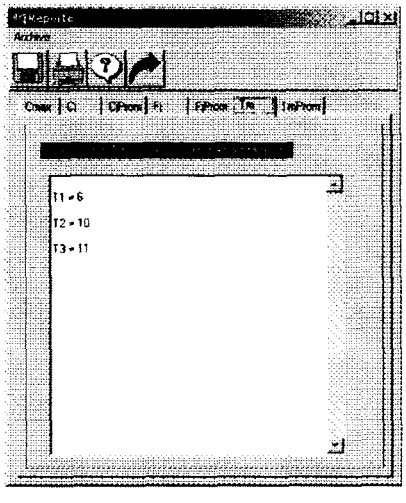


Figura 7-24. Tiempo de Terminación de cada Máquina

La septima pestaña es Tiempo Promedio de Terminación de cada máquina, también se puede acceder a esta opción mediante menú Reporte -> Reportes individuales -> TmProm.

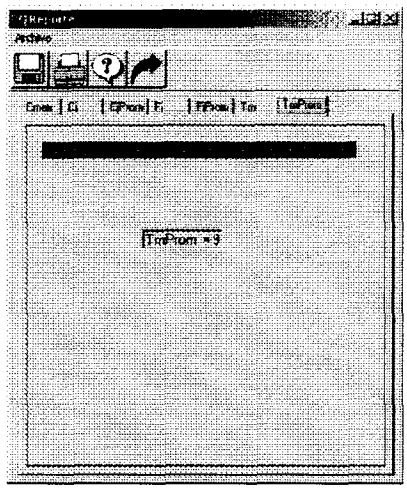


Figura 7-25. Tiempo Promedio de Terminación de cada Máquina.

También se puede acceder a cada uno de estos reportes de manera individual mediante menú Reportes -> Reportes Individuales -> opción que se desee consultar.

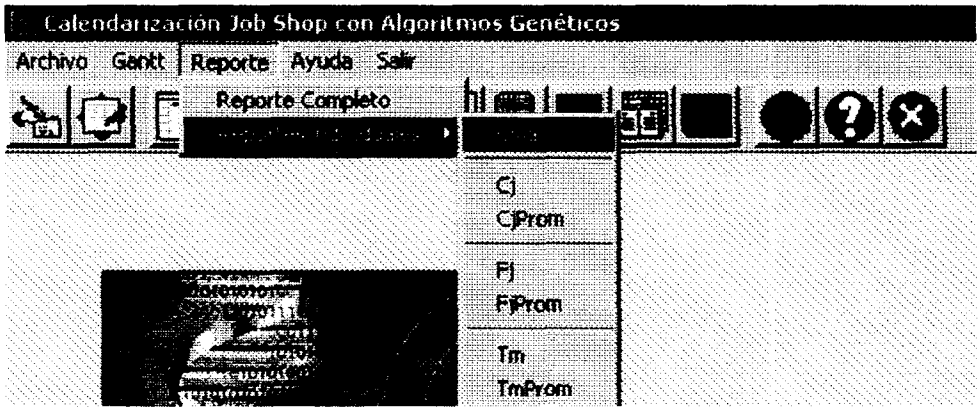


Figura 7-26. Reportes Individuales.

El reporte completo se puede guardar en la ruta y con el nombre que se desee, esto en un archivo TXT

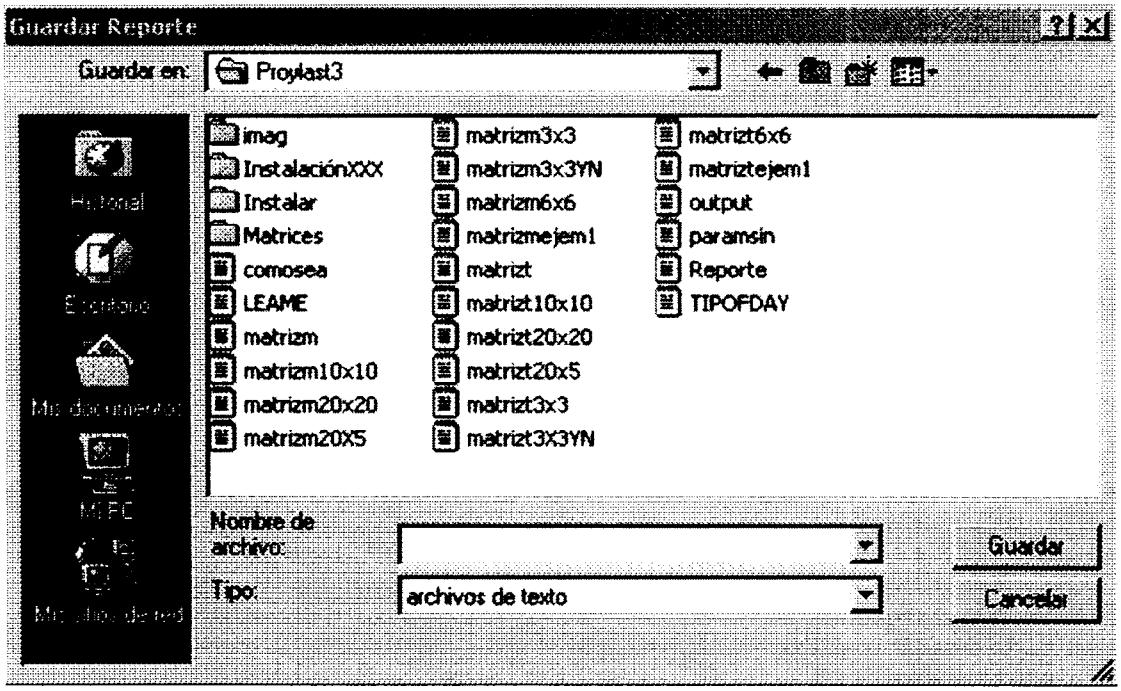


Figura 7-27. Guardar Reporte completo

Cuando se termina de guardar el reporte el sistema mostrará el siguiente mensaje.

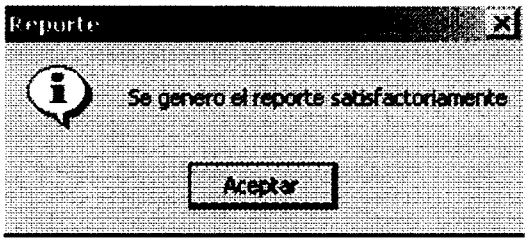


Figura 7-28. Mensaje de confirmación de la generación del reporte.

El formato del reporte generado como archivo TXT es el siguiente

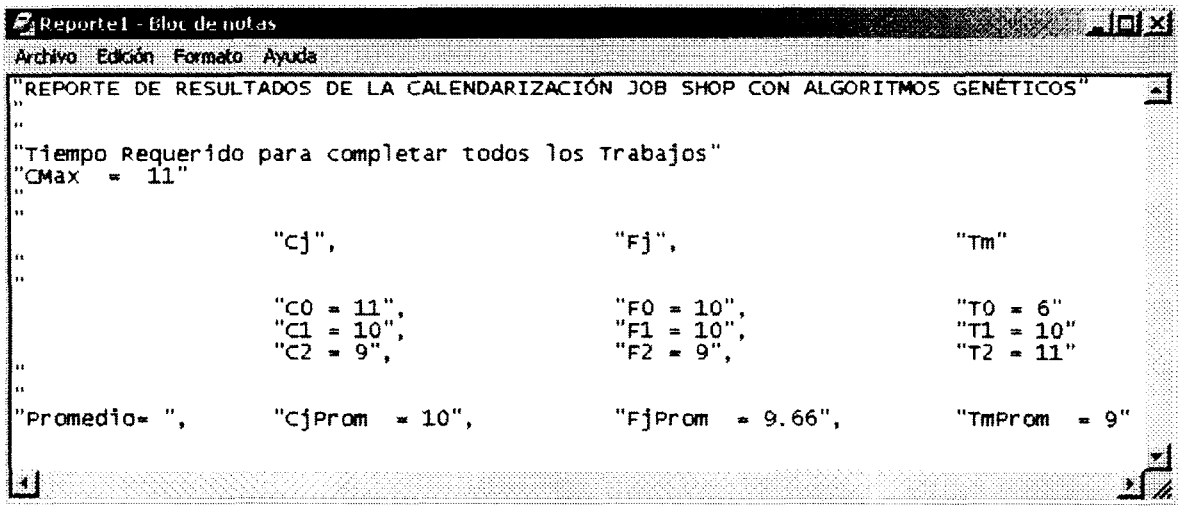


Figura 7-29. Archivo TXT del reporte generado

El sistema también presenta opciones adicionales que se encuentran en menú Ayuda, como es Sugerencia del día

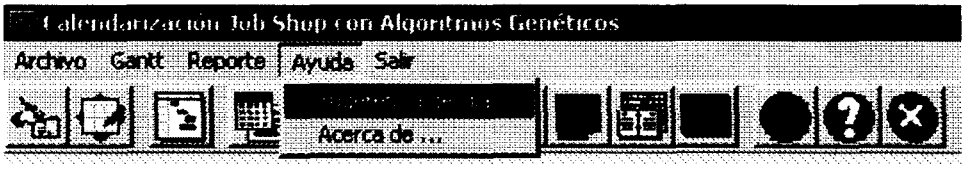


Figura 7-30. Menú Ayuda -> Sugerencia del día

Al seleccionar esta opción se mostrará una sugerencia para facilitar el uso del sistema. También se puede acceder a esta opción mediante el icono de signo de interrogación.

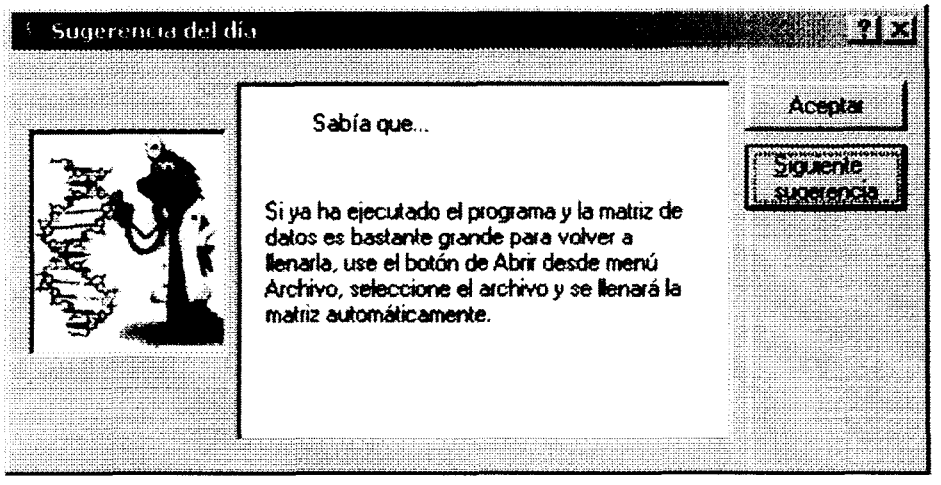


Figura 7-31. Sugerencia del día

Para ver otras sugerencias dar clic en el botón siguiente sugerencia y se mostrará otra sugerencia.

Para observar la información referente a los créditos del programa seguir la siguiente ruta, en el menú desplazarse hasta la opción de ayuda, de la siguiente manera Ayuda -> Acerca de...

Figura 7-32. Menú Ayuda -> Acerca de...

Al dar clic aparecerá la información acerca del sistema de Calendarización Job Shop con Algoritmos Genéticos.

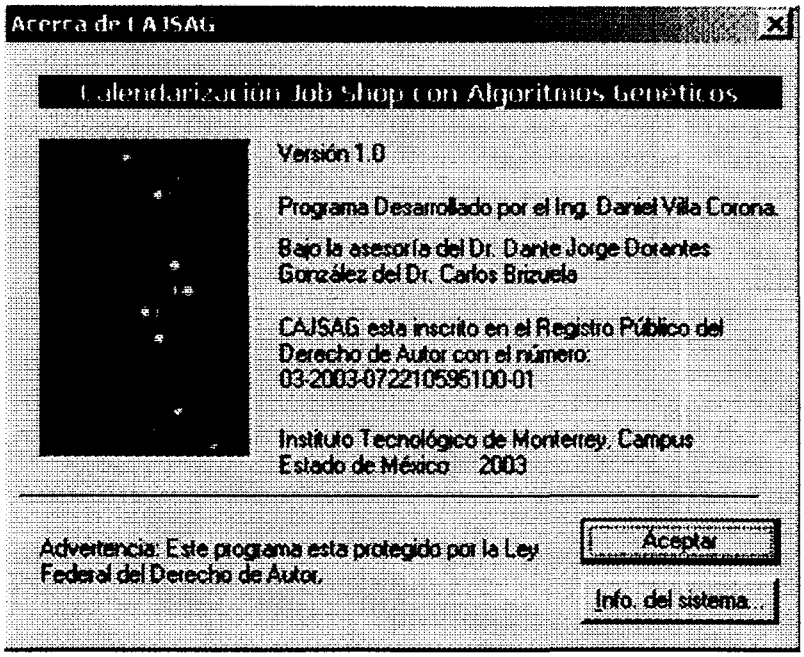


Figura 7-33. Acerca de CAJSAG.

Para salir del sistema se presenta en el menú salir la opción de cerrar Aplicación, que funciona igual que el icono con una X.

Figura 7-34. Menú Salir

Al dar clic en cerrar sesión aparecerá un cuadro de dialogo para preguntarle si realmente desea terminar la sesión, si da clic en si, se cerrará la sesión y saldrá del sistema, si selecciona no, puede continuar trabajando con el sistema de manera normal.

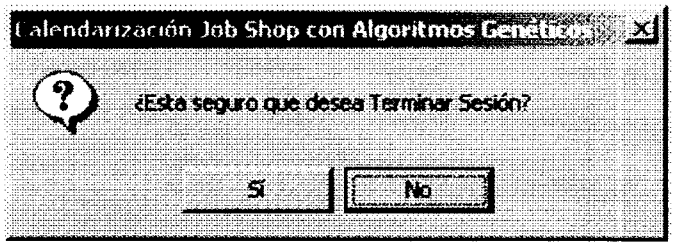


Figura 7-35. Mensaje finalizar sesión

7.3 CÓDIGO DEL PROGRAMA EN LENGUAJE C

Las variables globales principales utilizadas en el programa son las siguientes:

M;	matriz donde se almacena la matrizm, es decir la secuencia de operaciones de cada trabajo
T;	matriz donde se almacena la matrizt, es decir los tiempos de procesamiento.
StrL;	Variable que indica la longitud del cromosoma.
Sigma;	Estructura auxiliar para realizar operaciones con los cromosomas
Individual;	Estructura que contiene cada uno de los individuos junto con su makespan. De este tipo son newpop, oldpop y pop.
Jptr;	Lista enlazada para apuntar cierta dimensión de tamaño del número de máquinas
Mptr;	Lista enlazada para apuntar cierta dimensión de tamaño del número de trabajos

Para conocer más detalles sobre el código del programa ver el ANEXO A que es el diagrama del Algoritmo Genético.

7.4 RESULTADOS NUMÉRICOS

La implementación del algoritmo genético se hizo dinámica y estática. Las pruebas de la implementación del algoritmo genético se hicieron con los problemas de 10 máquinas, 10 trabajos y 20 trabajos, 5 máquinas, conocidos como de MUTH-THOMPSON (ver Anexo B). El programa utilizado en la aplicación CAJSAG, es el programa dinámico.

Los programas después de 10 corridas reportaron los siguientes resultados.

Tabla 7-1. Resultados después de 10 corridas

Corrida	ESTÁTICO Makespan	
	10 X 10	20 X 5
1	941	1250
2	957	1195
3	937	1165
4	956	1181
5	945	1184
6	956	1215
7	945	1190
8	930	1260
9	955	1190
10	953	1230

Corrida	DINÁMICO Makespan	
	10 X 10	20 X 5
1	956	1260
2	980	1242
3	953	1244
4	967	1275
5	977	1256
6	978	1270
7	956	1265
8	961	1260
9	977	1288
10	992	1276

Los mejores resultados obtenidos son:

Tabla 7-2. Los mejores resultados y sus parámetros para el algoritmo genético

	10 X 10		20 X 5	
	Estático	Dinámico	Estático	Dinámico
Makespan	930	953	1165	1242
Tamaño de la población	401	101	801	101
Tasa de cruzamiento	0.75	0.75	0.8	0.75
Tasa de mutación	0.2	0.2	0.2	0.2
Número de generaciones	1000	300	1000	300

Los resultados son satisfactorios comparándolos con los obtenidos por trabajos hechos anteriormente como los mostrados en la Tabla 7-3. Se observa que se logro alcanzar el resultado obtenido por los investigadores Shi/Ima/Sanomiya que crearon el algoritmo en el que se baso el programa del Algoritmo Genético.

Tabla 7-3. Comparación del desempeño de diversos Algoritmos Genéticos con los problemas de Muth- Thompson

Creadores	Algoritmo Genético	10 X 10	20X5
Nakano/Yamada	Convencional AG [22].	965	1215
Yamada/Nakano	Giffler-Thompson GT-GA [29].	930	1184
Dorndorf/Pesch	Basado en Reglas de Prioridad [23].	960	1249
Dorndorf/Pesch	Cuello de botella SB-GA [23].	938	1178
Kobayashi/Ono /Yamamura	Cruzamiento de intercambio de Subsecuencias SXX-GA [21].	930	1178
Bierwirth	Permutación Generalizada GP-GA.	936	1181
Yamada/Nakano	Fusion del Cruzamiento Multipasos MSXF-GA.	930	1165
Shi/Ima/Sanomiya	Cruzamiento por conjuntos o bloques, SPX-GA [26].	930	1165

7.4.1 ESTUDIO COMPARATIVO CON OTROS PROGRAMAS DE COMPUTO PARA CALENDARIZACIÓN

Se realizo el estudio con tres programas, LEKIN, WINQSB y REGLAS DE DESPACHO, este último realizado por el compañero José Carlos Pérez Rodríguez. Se hicieron las pruebas con los problemas propuestos de Muth- Thompson

LEKIN

Tabla 7-4. Resultados del programa LEKIN

Método	Makespan	
	10X10 MT	20X5 MT
SB	1094	1291
SB/sum(wt)	1177	1512
SB/Tmáx	930	1178
Local search	950	1170
ATCS (1,1)	1338	1558
EDD	1246	1672
MS	1168	1516
FCFS	1184	1645
LPT	1168	1516
SPT	1338	1558
WSPT	1338	1558
CR	1411	1513

WINQSB

Tabla 7-5. Resultados del programa WinQSB

	10X10 MT	20X5 MT
Makespan	1074	1267

Reglas de despacho

Tabla 7-6. Resultados del programa Reglas de despacho

	10X10 MT	20X5 MT
Makespan	1103	1312

Los mejores resultados se concentran en la siguiente tabla

Para el problema de 10x10 Muth-Thompson tenemos:

Tabla 7-7 Comparativo de resultados para el problema 10x10MT

Makespan	930	953	930	1074	1103

Para el problema de 20x5 Muth-Thompson tenemos:

Tabla 7-8. Comparativo de resultados para el problema 20x5MT

Makespan	1165	1242	1170	1267	1312

Como se observa en las tablas el programa CAJSAG presenta el mejor desempeño, con esto se comprueba también que los resultados obtenidos por el programa son confiables y útiles.

8 APLICACIONES DEL PROGRAMA CAJSAG

8.1 APLICACIÓN EN LA CALENDARIZACIÓN DE UN PROCESO DE LA INDUSTRIA MUEBLERA

Con el objetivo de probar el desempeño del programa computacional de calendarización por algoritmos genéticos CAJSAG se buscó una empresa en la cual pudiéramos analizar su proceso de producción y recopilar datos de las operaciones involucradas.

Grupo Maple, fabricante de muebles con calidad de exportación nos brindó la facilidad de aplicar el sistema de calendarización por algoritmos genéticos CAJSAG a uno de sus procesos de producción, el cual es la fabricación de los muebles de una recamara modelo *Napoli.* Los muebles de esta recamara son: cabecera, espejo y buró.Cada uno de estos muebles esta compuesto por varias piezas que tienen que ser manufacturadas simultáneamente para después terminar el proceso de fabricación con el armado final.

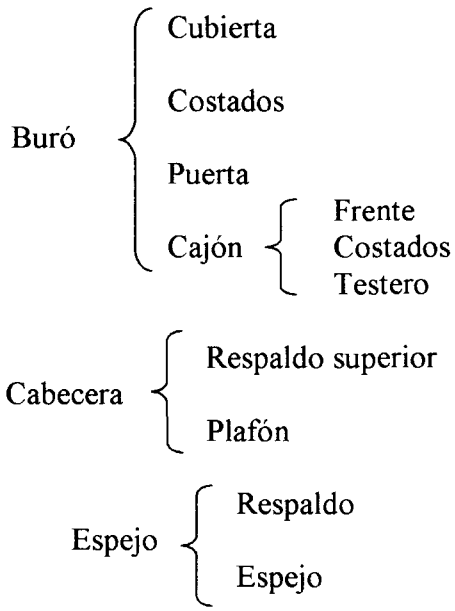


Figura 8-1. Partes que componen a cada mueble

Las operaciones para la fabricación de las piezas de cada mueble se enlistan a continuación

Para el buró son las siguientes

CUBIERTA	COSTADOS	PUERTA
Habilitación	Habilitación	Habilitación
Tablero	Tablero	Tablero
↓	↓	↓
Chapar	Cortar	Ranurar
↓	↓	↓
Cortar	Chapar	Cortar
↓	↓	↓
Ranurar	Barrenar	Chaflán
↓	↓	↓
Barrenar	Pulir	Barrenar
↓		↓
Hacer		Pulir
Chaflán		
↓		
Pulir		

Figura 8-2. Secuencia de operaciones para el buró

El Cajón se subdivide en las siguientes partes, por lo que a continuación se enlistan sus operaciones

CAJÓN		
FRENTE	COSTADOS	TESTERO
Habilitación	Habilitación	Habilitación
Madera	Sobrante	Tablero
↓	↓	↓
Cortar	Cortar	Cortar
↓	↓	↓
Ranurar	Pulir	Ranurar
↓		↓
Chapar		Chapar
↓		↓
Pulir		Pulir

Figura 8-3. Secuencia de operaciones para el cajón

Para fabricar el espejo se realizan las siguientes operaciones:

ESPEJO

Habilitación
Tablero
↓
Cortar
↓
Chapar
↓
Pulir
↓
Aplicar tinta
↓
Aplicar cera
↓
Pegar espejo

Figura 8-4. Secuencia de operaciones para el espejo

Para fabricar la cabecera se realizan las siguientes operaciones:

CABECERA

RESPALDO	PLAFÓN
Habilitación Tablero	Habilitación Tablero
↓	↓
Cortar	Cortar
↓	↓
Prensado a altafrecuencia	Prensado a altafrecuencia
↓	↓
Perfilar en CNC	Perfilar en CNC
↓	↓
Chapar	Chapar
↓	↓
Pulir	Pulir
↓	↓
Aplicar tinta	Aplicar tinta
↓	↓
Aplicar cera	Aplicar cera

Figura 8-5. Secuencia de operaciones para la cabecera

Para proceder a la calendarización de las distintas operaciones para la fabricación de la recamara, procedemos a la elaboración de un listado de todas las operaciones presentes en el proceso de fabricación, resultando 14 operaciones y el listado siguiente:

Tabla 8-1. Operaciones para la fabricación de la recamara

Numero de operación	Descripción
1	Habilitar Tablero
2	Chapar
3	Cortar
4	Ranurar
5	Barrenar
6	Hacer Chaflán
7	Pulir
8	Habilitar Madera
9	Habilitar Sobrante
10	Aplicar Tinta
11	Pegar espejo
12	Prensado a Altafrecuencia
13	Maquinado CNC
14	Aplicar Cera

A los trabajos o piezas a producir se les asignó una letra para identificarlas en el proceso de calendarización, al momento de introducirlas en el sistema CAJSAG se les asignará el número correspondiente al orden de la letra correspondiente.

Tabla 8-2. Tareas a realizarse para la producción de recamara

Identificador	Pieza
A	Cubierta del buró
B	Costados del buró
C	Puerta del buró
D	Frente del cajón del buró
E	Costados del cajón del buró
F	Testera del cajón del buró
G	Espejo
H	Respaldo de la cabecera
I	Plafón de la cabecera

Al introducir los datos al sistema CAJSAG se tuvieron que hacer ajustes, ya que para producir las piezas se tienen que realizar operaciones diferentes en cada una de estas piezas, entonces las operaciones que no se realizan en la pieza en cuestión se le asigna un tiempo de procesamiento de cero, quedando la tabla de secuencia de operaciones y tiempo de procesamiento como sigue:

Tabla 8-3. Tabla de secuencia de operaciones y tiempo de procesamiento

	OPERACIONES																											
	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T
A	1	10	2	10	3	5	4	3	5	5	6	3	7	8	8	0	9	0	10	0	11	0	12	0	13	0	14	0
B	1	10	3	5	2	10	5	5	7	8	4	0	6	0	8	0	9	0	10	0	11	0	12	0	13	0	14	0
C	1	10	4	3	3	5	2	10	6	4	5	5	7	8	8	0	9	0	10	0	11	0	12	0	13	0	14	0
D	8	10	3	4	4	3	2	8	7	10	1	0	5	0	6	0	9	0	10	0	11	0	12	0	13	0	14	0
E	9	10	3	4	7	5	1	0	2	0	4	0	5	0	6	0	8	0	10	0	11	0	12	0	13	0	14	0
F	1	10	3	5	4	3	2	10	7	5	5	0	6	0	8	0	9	0	10	0	11	0	12	0	13	0	14	0
G	1	15	3	5	2	10	7	20	10	5	14	15	11	15	4	0	5	0	6	0	8	0	9	0	12	0	13	0
H	1	15	3	10	12	30	13	10	2	15	7	10	10	8	14	15	4	0	5	0	6	0	8	0	9	0	11	0
I	1	15	3	10	12	30	13	15	2	10	7	8	10	5	14	15	4	0	5	0	6	0	8	0	9	0	11	0

8.1.1 RESULTADO

El resultado obtenido es de un makespan de 135, y la asignación y tiempos de finalizacion de cada operación se observa en el diagrama de Gantt

Los resultados obtenidos nos sirvieron para observar el desempeño del sistema CAJSAG con un problema práctico y además generar una solución en el problema de decisión de asignación de tareas en las máquinas de la empresa que de la cual proceden estos datos.

Figura 8-6. Diagrama de Gantt

En la gráfica de Gantt se observa que las operaciones que marcan la pauta en la calendarización son:

Habilitación de Tablero
Habilitación de Madera y
Habilitación de sobrante

La operación de habilitación de Tablero se realiza a siete de las nueve piezas y se debe hacer en la siguiente secuencia

1. Respaldo de la cabecera
2. Espejo
3. Plafón de cabecera
4. Cubierta de buró
5. Puerta de buró
6. Costados del buró
7. Testero del cajón

El chapado es otra de las operaciones críticas, que se debe hacer con la secuencia siguiente.

1. Frente de cajón
2. Espejo
3. Cubierta de buró
4. Respaldo de la cabecera
5. Puerta de buró
6. Costados del buró
7. Plafón cabecera
8. Testero del cajón

Con lo referente a operaciones de corte se tiene la siguiente secuencia

1. Frente de cajón
2. Respaldo de la cabecera
3. Costados del buró
4. Espejo
5. Plafón de cabecera
6. Cubierta de buró
7. Puerta de buró
8. Costados de buró
9. Testero del cajón

Las secuencias obtenidas por medio del diagrama de Gantt indican cual es la decisión que se debe tomar cuando entran en conflicto varias operaciones. Estas secuencias fueron las recomendaciones hechas a la gente de producción de Grupo Maple S.A. de C.V. con el fin de optimizar este proceso. Sin embargo la calendarización completa no tuvo gran utilidad debido a que los tiempos de holgura que se manejan en esa industria son muy amplios con respecto a la optimización del Cmáx que se puede lograr con el programa computacional CAJSAG.

8.2 APLICACIÓN DE LA CALENDARIZACIÓN EN UN PROCESO DE LA INDUSTRIA METALMECANICA (EJEMPLO)

Considerando un sistema de manufactura, el cual consiste de un torno, una fresadora, un taladro, un robot (brazo), y una banda transportadora. Se pretende calendarizar el maquinado de tres piezas que ocupan todas las máquinas del sistema de manufactura.

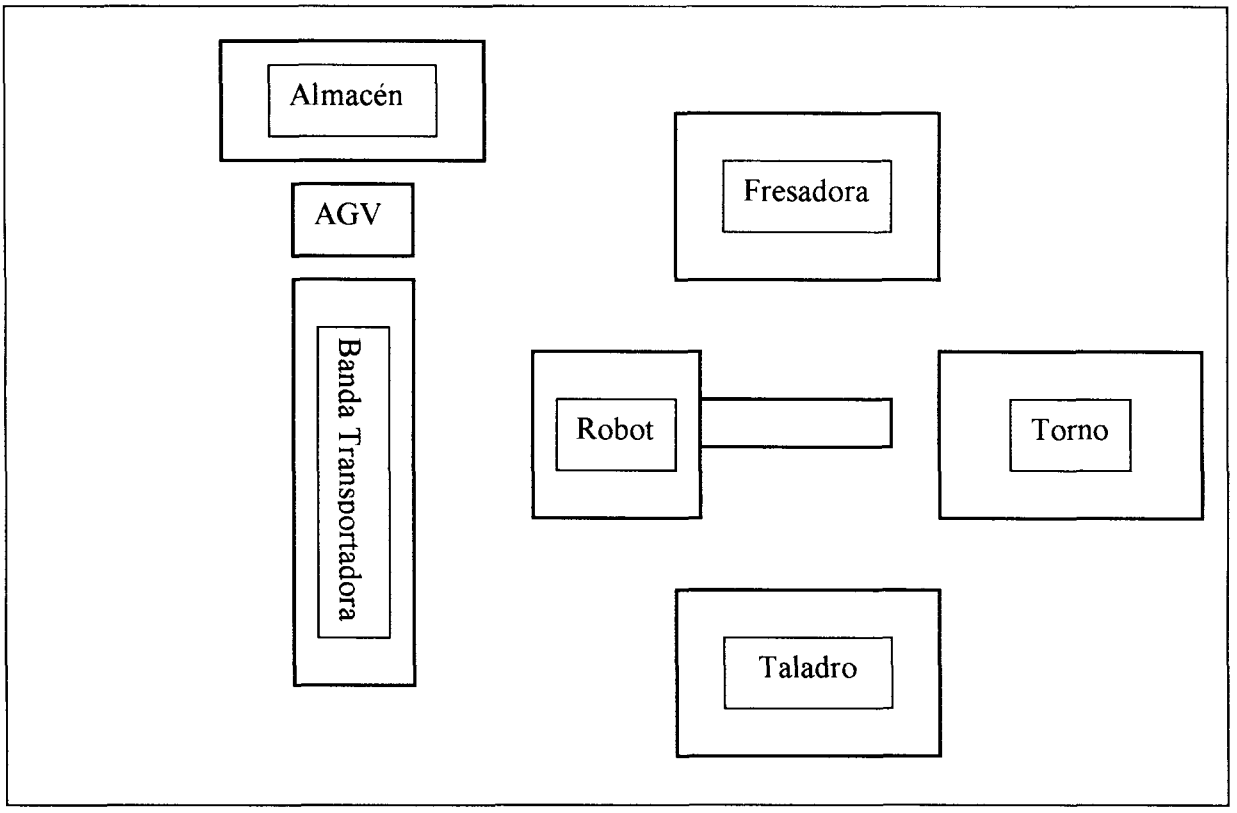


Figura 8-7. Distribución de las máquinas en el sistema de manufactura

Tabla 8-4. Secuencia de operaciones y tiempo de procesamiento

	SECUENCIA DE OPERACIONES								
Pieza 1	AGV Banda (1min)	Robot (1min)	Taladro (3min)	Robot (1min)	Fresa (4min)	Robot (1min)	Torno (3min)	Robot (1min)	AGV Banda (1min)
Pieza 2	AGV Banda (1min)	Robot (1min)	Torno (5min)	Robot (1min)	Taladro (3min)	Robot (1min)	Fresa (4min)	Robot (1min)	AGV Banda (1min)
Pieza 3	AGV Banda (1min)	Robot (1min)	Taladro (3min)	Robot (1min)	Torno (4min)	Robot (1min)	Fresa (5min)	Robot (1min)	AGV Banda (1min)

Piezas a maquinarse

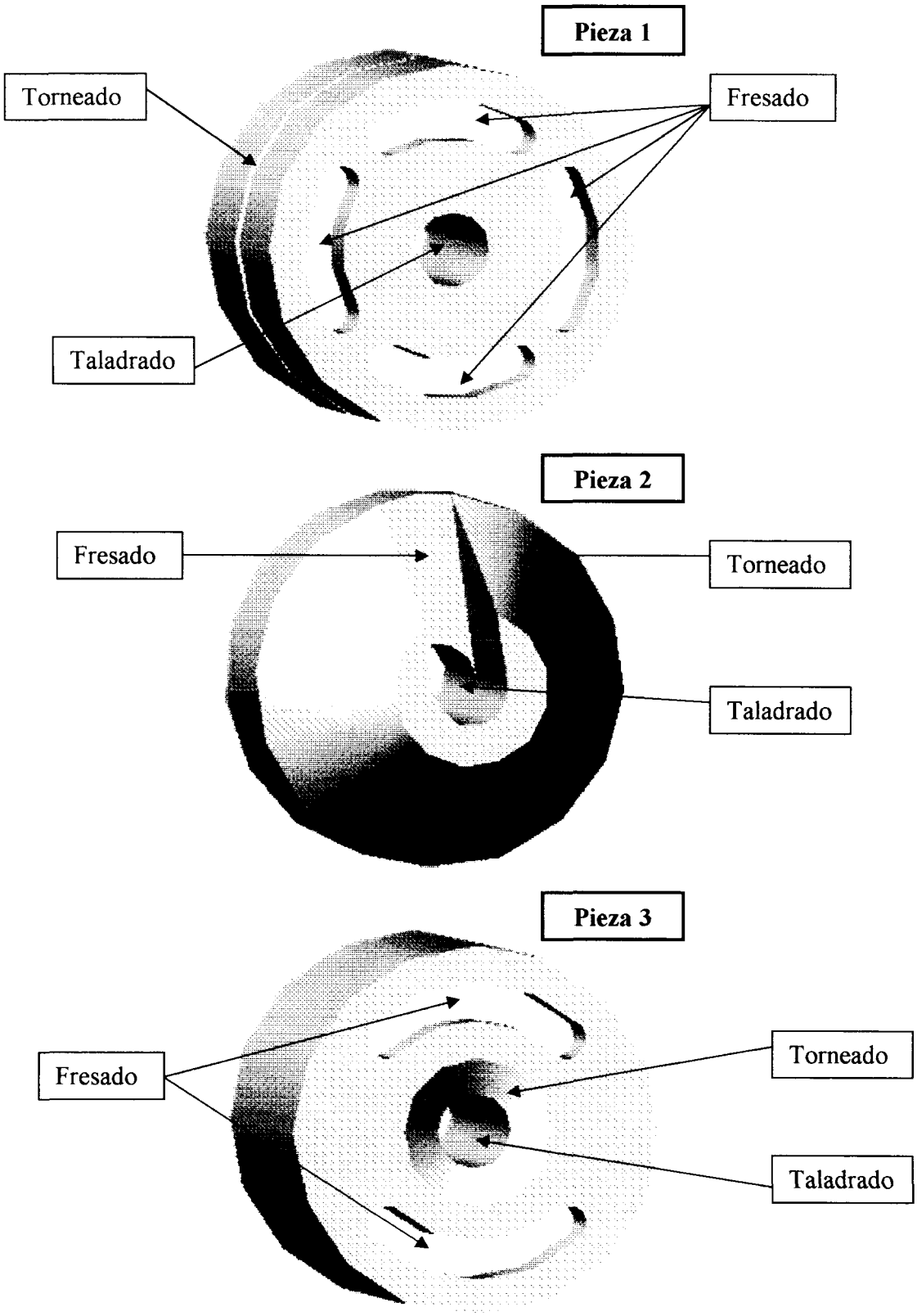


Figura 8-8. Piezas a Maquinarse

La secuencia de operaciones y tiempos de procesamiento al introducirse al sistema CAJSAG queda como sigue:

Tabla 8-5. Introducción de valores en CAJSAG

	1		2		3		4		5		6		7		8		9	
	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T
T1	4	1	5	1	1	3	6	1	3	4	7	1	2	3	8	1	9	1
T2	4	1	5	1	2	5	6	1	1	3	7	1	3	4	8	1	9	1
T3	4	1	5	1	1	3	6	1	2	4	7	1	3	5	8	1	9	1

En la siguiente figura se observa el resultado de la calendarización en un diagrama de Gantt.

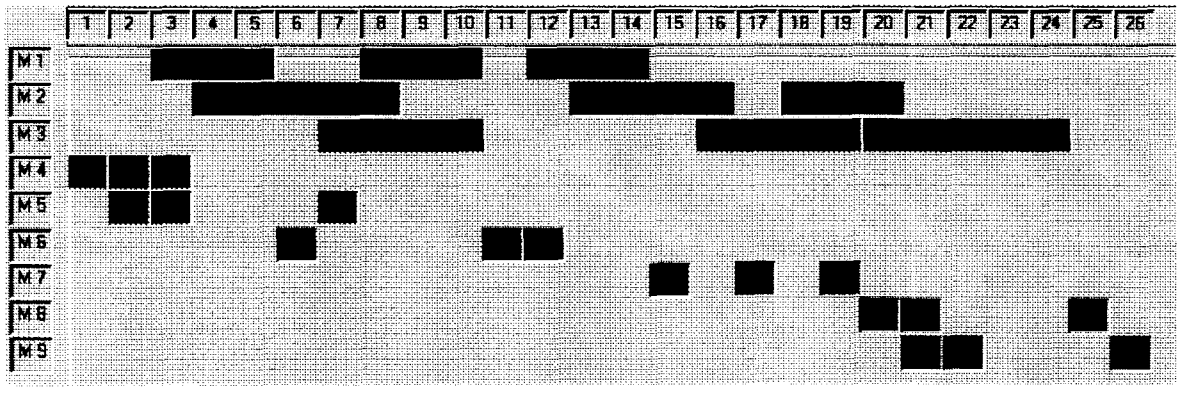


Figura 8-9. Resultado de la calendarización. Diagrama de Gantt

La máquina M4 representa al AGV y a la banda transportadora, los cuales colocan la pieza en cuestión para que pueda ser tomada por el robot. El AGV es un dispositivo que automáticamente toma el pallet en el que se encuentra la pieza y lo posiciona en la banda transportadora.

El robot esta representado por las máquinas M5, M6, M7 y M8. La máquina M5 representa el desplazamiento del robot, desde que toma cada una de las piezas del pallet y hasta que las posicionan en la máquina que realizará la primera operación en la pieza. La máquina M6 representa el desplazamiento del robot desde que toma la pieza de la máquina donde se realizó la primera operación y hasta que coloca la pieza en la máquina donde se realizará la segunda operación. La máquina M7 representa el desplazamiento del robot desde que toma la pieza de la máquina donde se realizó la segunda operación y hasta que coloca la pieza en la máquina donde se realizará la tercera operación. Y por ultimo la máquina M8 representa el desplazamiento del robot desde que toma la pieza de la máquina donde se realizó la tercera operación y hasta que coloca la pieza en la banda transportadora, que llevara las piezas al almacén. La máquina M9 representa también al AGV y a la banda transportadora, que colocan a la pieza terminada en el almacén.

9 CONCLUSIONES

Los capítulos que conforman esta tesis son el resultado del estudio de los algoritmos genéticos enfocado hacia la calendarización de sistemas de producción tipo Job Shop. Se han elaborado los nueve capítulos con el propósito de constituir un texto de referencia que presente de modo coherente y detallado los temas más importantes sobre algoritmos genéticos aplicados a la calendarización Job Shop

En definitiva cabe concluir que la principal cualidad de los algoritmos genéticos con respecto a otras técnicas de búsqueda es la introducción de mecanismos de selección de los candidatos a solución en función de sus respectivas aptitudes y de mecanismos de construcción de nuevos candidatos por cruzamiento y mutación de los individuos existentes. Unos y otros mecanismos proporcionan robustez a la búsqueda, esto es, le añaden eficiencia en favor de la búsqueda y obtención de un óptimo que satisfaga la solución del problema de calendarización

El desarrollo del programa del algoritmo genético se planteó que fuera dinámico, esto es que fuera flexible a poder calendarizar problemas de n trabajos y m máquinas, con la finalidad de resolver problemas de dimensión rectangular, es decir, que el número de trabajos sea mayor al número de máquinas y viceversa, y así extender el espacio de problemas a resolver.

Se logro la implementación de un algoritmo genético para resolver el problema de calendarización Job Shop tipo *N/M/G/Cmáx*. El algoritmo es un algoritmo genético con representación de calendarización por operaciones, cruzamiento por partición de conjuntos SPX y con mutación de intercambio y de intercambio segmentado.

Para el desarrollo del algoritmo genético se usó el lenguaje C porque es un programa estructurado capaz de manejar memoria en tiempo de ejecución y maneja con facilidad punteros, estructuras y arreglos dinámicos, que son elementos esenciales en el desarrollo del programa.

Se planteó el diseño de una aplicación de fácil manejo para el usuario, por lo que el programa del algoritmo genético toma los datos de entrada de un archivo de texto, los cuales son número de máquinas, número de trabajos, tamaño de la población, tasa de cruzamiento, tasa de mutación, y máximo número de generaciones, con esto se inicia el algoritmo genético, una vez que se ejecuto el algoritmo de manera satisfactoria, el programa genera un archivo de texto de salida que contiene la solución de la calendarización, esto es la definición de tiempos de inicio y finalización de cada una de las operaciones que forman parte del problema.

La aplicación es muy gráfica y sencilla en función de su manejo, ya que esta creada para facilitar la introducción de datos del problema por medio de ventanas que indican los parámetros necesarios para iniciar la solución del problema de calendarización, así mismo favorece la visualización de la solución de la calendarización, por medio de una gráfica de Gantt donde se especifican los tiempos de inicio y termino de cada operación, donde cada bloque posee una etiqueta que muestra su identificación por medio del índice *operación, máquina*, además el conjunto de bloques que conforman un trabajo se pintan del mismo color, esto para hacer mas comprensible el diagrama de Gantt. Para mostrar los parámetros de desempeño del algoritmo genético, la aplicación cuenta con un reporte detallado que muestra el *makespan* obtenido, el tiempo requerido para completar cada trabajo, el tiempo de flujo de cada trabajo, el tiempo de terminación de cada máquina, y el promedio de cada uno de estos índices.

La aplicación se ha probado con una serie de problemas, siendo los más representativos los propuestos por Muth- Thompson con dimensiones de 10 trabajos 10 máquinas y 20 trabajos 5 máquinas, obteniéndose resultados satisfactorios, es decir cercanos a los obtenidos por los Algoritmos Genéticos de investigadores en el tema, mostrados en la Tabla 6.3. Además la aplicación se probó contra tres programa de cómputo que resuelven problemas de calendarización Job Shop, estos programas son LEKIN, WinQSB y REGLAS DE DESPACHO. Los resultados de este comparativo se muestran en las tablas 7-7 y 7-8. En estas tablas se observa que el programa CAJSAG obtiene el menor makespan entre los programas con los que se realizó el comparativo, esto muestra el buen desempeño del programa CAJSAG con respecto a la función objetivo del algoritmo genético que es *minimizar el makespan o Cmáx.*

Por los resultados de las pruebas y las características mencionadas en los párrafos anteriores, con el programa CAJSAG se lograron los méritos suficientes para obtener sus derechos de autor, por lo cual estos derechos se tramitaron y se obtuvieron, quedando registrado con el número de registro: 03-2003-072210595100-01, siendo autores y titulares de este programa el Dr. Dante Jorge Dorantes González y el Ing. Daniel Villa Corona. El certificado de los derechos de autor del programa CAJSAG se muestra en el anexo C.

Concretamente se creó una herramienta robusta con muy buen desempeño para la calendarización de un sistema de producción tipo Job Shop por medio de algoritmos genéticos, que presenta una interfaz gráfica y amigable para el usuario.

10 REFERENCIAS Y BIBLIOGRAFÍA

[1] *Cuadros estadísticos por tema.* http://www.inegi.gob.mx

[2] DORANTES D. J., LOPEZ J. A., *Sistemas flexibles de manufactura: una opción impostergable para el desarrollo económico de México.* ITESM CEM.

[3] GÁLVEZ HERNÁNDEZ J.C., *Desarrollo de un algoritmo heurístico para la calendarización multicriterial sin demoras para un sistema de producción tipo job shop,* ITESM CEM. Tesis de Maestría en Sistemas de Manufactura, 2001.

[4] PINEDO, MICHAEL, *Scheduling: Theory, Algorithms, and Systems*, Englewood Cliffs, Prentice Hall, 2002.

[5] CHASE R.B., AQUILANO N. J. *Production and Operation Management*, 8th edition, Irwin-Mc Graw Hill, 1998.

[6] MORTON THOMAS E. AND PENTICO DAVID W., *Heuristic Scheduling Systems with Applications to Production Systems and Project Management*, Wiley-Interscience, 1993.

[7] RALPH V. ROGERS AND K. PRESTON WHITE JR., *Algebraic. Mathematical Programming, and Network Models of the Deterministic Job-Shop Scheduling Problem*, Transaction on Systems, Man and Cybernetics, Vol 21, No. 3, p. 693-697, May/Jun 1991.

[8] EBRAU DEMIRKOL REHA UZSOY AND IRFANM. OVACIK, *Decomposition Algorithms for Scheduling Semiconductor Testing Facilities*, Electronic Manufacturing Technology Symposium, p 199-204, 1995.

[9] JIHUA WANG AND PETER B. LUH. *Optimization-Based Scheduling of a Machining Center*, International Conference on Robotics and Automation, p. 502-507,1995.

[10] EUI-SEOK BYEON, S. DAVID WU, Y ROBERT H. STORER, Decomposition Heuristic for Robust Job-Shop Scheduling, Transaction on Robotics and Automation, Vol 14 No. 2,p. 303-313, 1998.

[11] Y.LI, K.F. MAN, Scheduling and Planning Problem in Manufacturing Systems with Multiobjective Genetic Algorithm, p. 274-279, 1998.

[12] COELLO COELLO, CARLOS A. "Introducción a los Algoritmos Genéticos", *Soluciones Avanzadas. Tecnologías de Información y Estrategias de Negocios*, Año 3, No. 17, Enero de 1995, pp. 5-11.

[13] HOLLAND, JOHN H. "Adaptation in Natural and Artificial Systems", University of Michigan Press, 1975, 211 p.

[14] GEN MITSUO, CHENG RUNWEI, "Genetic algorithms and engineering design", Wiley series in engineering design and automation, 1997.

[15] GOLDBERG, DAVID E. "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison-Wesley Publishing Company, 1989, 412 p.

[16] GEN MITSUO, RUNWEI CHENG, ERICKA KUBOTA, "Solving Job-Shop Scheduling Problems by Genetic Algorithm", Proc. of 1994 IEEE International Conference on SMC, pp, 1577-1582, 1994.

[17] TSUJIMURA YASUHIRO, Y. MAFUNE, G. MITSUO, "Effects of Symbiotic Evolution in Genetic Algorithms for Job-Shop Scheduling", Proc. of the 34th Hawaii International conference on Systems Science, IEEE, 2001.

[18] DAVIS L., "Job Shop Scheduling With Genetic Algorithms", Proceedings of the First International Conference on Genetic Algorithms, pp. 136-140,1985.

[19] FALKENAUER E., S. BOUFFOIX, "A genetic algorithm for Job-Shop", Proceedings of the IEEE International Conference on Robotics and Automation. Pags. 824-829, 1991.

[20] HOLSAPPLE, C.W.; JACOB, V.S.; PAKATH, R.; ZAVERI, J.S. "A genetics-based hybrid scheduler for generating static schedules in flexible manufacturing contexts" Systems, Man and Cybernetics, IEEE Transactions on , Volume: 23 Issue: 4 , July-Aug. Pags. 953 -972, 1993

[21] KOBAYASHI S., I. ONO Y M. YAMAMURA, "An Efficient genetic algorithm for job shop scheduling programs". Proceedings of the Sixth International Conference on Genetic Algorithms, pp. 506-511, 1995.

[22] NAKANO R., T. YAMADA, "Conventional genetic algorithms for job-shop problems", Proceedings of the Fourth International Conference on Genetic Algorithms, pp. 477-479, 1991.

[23] DORNDORF, U., E. PESH, "Evolution based learning in a job shop scheduling environment", Computers and operations research. Vol. 22, pp. 25-40. 1995.

[24] ADAMS, J., E. BALAS Y D. ZAWACK. "The shifting bottleneck procedure for job shop scheduling". International Journal of flexible Manufacturing Systems. Vol. 34, No. 3, pp.391-401, 1987.

[25] BEAN, J., "Genetic algorithms and random keys for sequencing and optimization", ORSA Journal on Computing, vol. 6, no. 2, pp. 154-160, 1994.

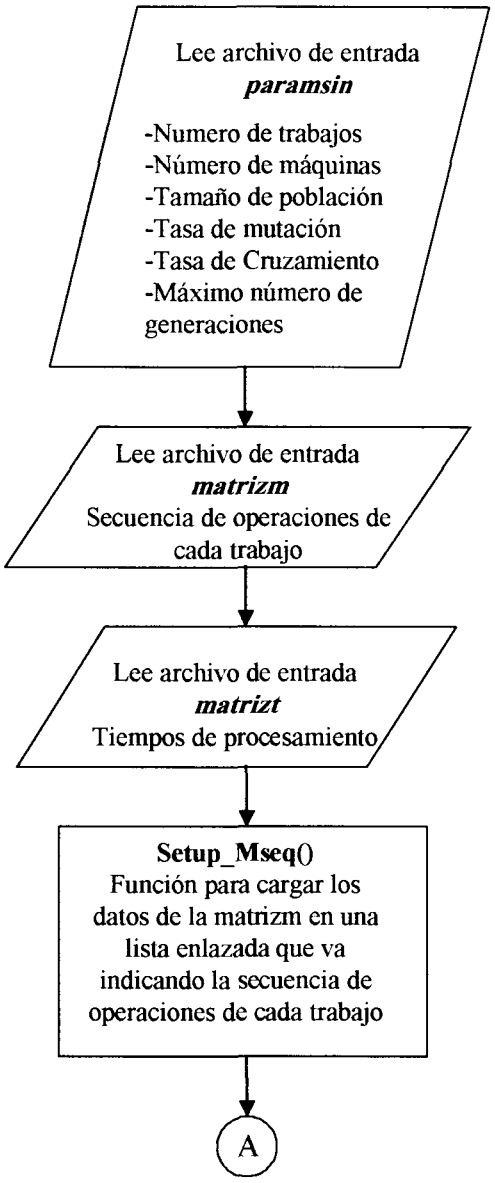
[26] G. SHI, H. IIMA AND N. SANNOMIYA. "A New Encoding Scheme for Solving Job Shop Problems by Genetic Algorithm". Proceedings of 35th IEEE Conference on Decision and Control, Vol. 4, pp: 4395-4400 (1996).

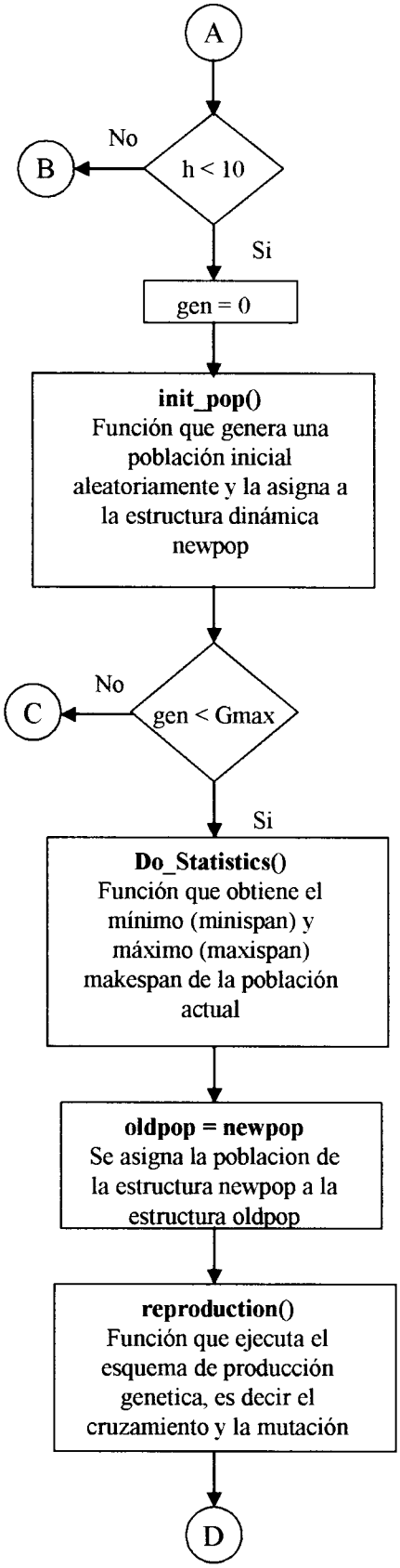
[27] YAMADA T., R. NAKANO.A genetic algorithm applicable to large-scale job-shop problems 1992. Parallel Problem Nolving from Nature: PPSN II, 1992.

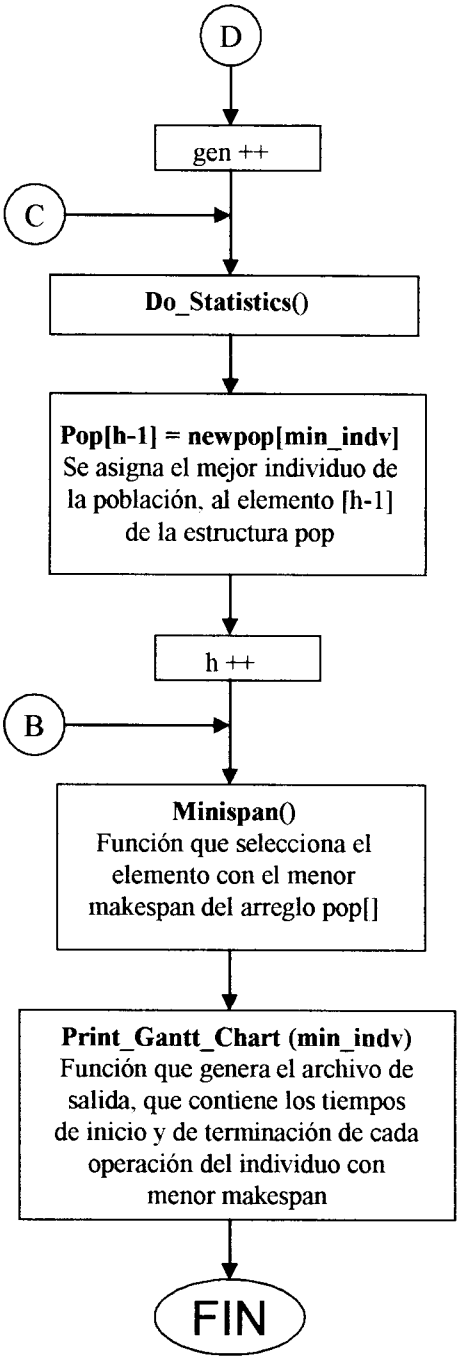
[28] GEN MITSUO, CHENG RUNWEI, "Genetic algorithms and engineering OPTIMIZATION", Wiley-Interscience publication, 2000.

[29] YAMADA T., R. NAKANO, "Genetic algorithms in engineering systems", Cap. 7: Job-Shop scheduling (pp. 134-160). 1997

ANEXO A. DIAGRAMA DE FLUJO DEL PROGRAMA DEL ALGORITMO GENÉTICO

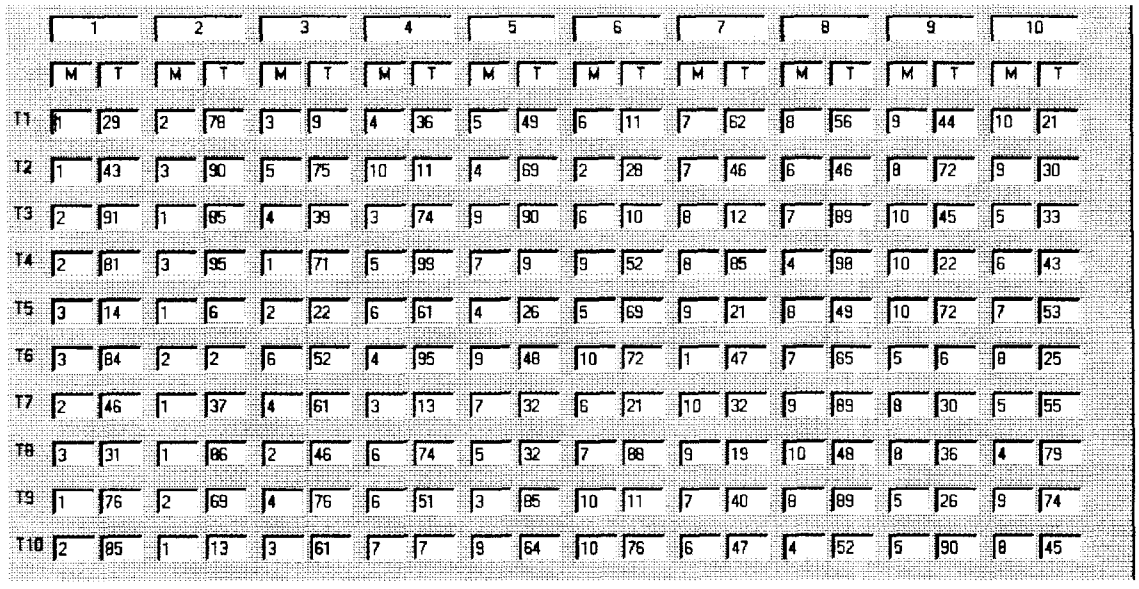



A
No
B
h < 10
Si
gen = 0
init_pop()
Función que genera una población inicial aleatoriamente y la asigna a la estructura dinámica newpop
No
C
gen < Gmax
Si
Do_Statistics()
Función que obtiene el mínimo (minispan) y máximo (maxispan) makespan de la población actual
oldpop = newpop
Se asigna la poblacion de la estructura newpop a la estructura oldpop
reproduction()
Función que ejecuta el esquema de producción genetica, es decir el cruzamiento y la mutación
D


D
gen ++
C
Do_Statistics()
Pop[h-1] = newpop[min_indv]
Se asigna el mejor individuo de la población, al elemento [h-1] de la estructura pop
h ++
B
Minispan()
Función que selecciona el elemento con el menor makespan del arreglo pop[]
Print_Gantt_Chart (min_indv)
Función que genera el archivo de salida, que contiene los tiempos de inicio y de terminación de cada operación del individuo con menor makespan
FIN

ANEXO B. PROBLEMAS DE MUTH-THOMPSON

Problema Muth-Thompson 10 trabajos, 10 máquinas



	1		2		3		4		5		6		7		8		9		10	
	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T	M	T
T1	1	29	2	78	3	9	4	36	5	49	6	11	7	62	8	56	9	44	10	21
T2	1	43	3	90	5	75	10	11	4	69	2	28	7	46	6	46	8	72	9	30
T3	2	91	1	85	4	39	3	74	9	90	6	10	8	12	7	89	10	45	5	33
T4	2	81	3	95	1	71	5	99	7	9	9	52	8	85	4	98	10	22	6	43
T5	3	14	1	6	2	22	6	61	4	26	5	69	9	21	8	49	10	72	7	53
T6	3	84	2	2	6	52	4	95	9	48	10	72	1	47	7	65	5	6	8	25
T7	2	46	1	37	4	61	3	13	7	32	6	21	10	32	9	89	8	30	5	55
T8	3	31	1	86	2	46	6	74	5	32	7	88	9	19	10	48	8	36	4	79
T9	1	76	2	69	4	76	6	51	3	85	10	11	7	40	8	89	5	26	9	74
T10	2	85	1	13	3	61	7	7	9	64	10	76	6	47	4	52	5	90	8	45

Problema Muth-Thompson 20 trabajos, 5 máquinas.

	1		2		3		4		5	
	M	T	M	T	M	T	M	T	M	T
T1	1	29	2	9	3	49	4	62	5	44
T2	1	43	2	75	4	69	3	46	5	72
T3	2	91	1	39	3	90	5	12	4	45
T4	2	81	1	71	5	9	3	85	4	22
T5	3	14	2	22	1	26	4	21	5	72
T6	3	84	2	52	5	48	1	47	4	6
T7	2	46	1	61	3	32	4	32	5	30
T8	3	31	2	46	1	32	4	19	5	36
T9	1	76	4	76	3	85	2	40	5	26
T10	2	85	3	61	1	64	4	47	5	90
T11	2	78	4	36	1	11	5	56	3	21
T12	3	90	1	11	2	28	4	46	5	30
T13	1	85	3	74	2	10	4	89	5	33
T14	3	95	1	99	2	52	4	98	5	43
T15	1	6	2	61	5	69	3	49	4	53
T16	2	2	1	95	4	72	5	65	3	25
T17	1	37	3	13	2	21	4	89	5	55
T18	1	86	2	74	5	88	3	48	4	79
T19	2	69	3	51	1	11	4	89	5	74
T20	1	13	2	7	3	76	4	52	5	45

ANEXO C. CERTIFICADO DE DERECHOS DE AUTOR

El certificado de derechos de autor del programa de computo **CAJSAG** (**C**alendarización **J**ob **S**hop con **A**lgoritmos **G**enéticos) se muestra en la siguiente hoja