ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Instituto de Ciencias Matemáticas

"Diseño e Implantación de un Sistema de Información para la Asignación Optima de Pedidos de una Empresa Manufacturera basado el algoritmo Recocido Simulado"

TESIS DE GRADO

Previa la obtención del título de:

INGENIERO EN ESTADÍSTICA INFORMÁTICA

Presentada por:

Mónica Paola Rojas Ramírez

GUAYAQUIL – ECUADOR

AÑO

2004

AGRADECIMIENTO

Agradezco primeramente a Dios, porque siempre me ha acompañado en los momentos mas difíciles haciéndome comprender que en esta vida nada se obtiene fácilmente, sino a través de mucho esfuerzo y sacrificio.

A mi familia, por el apoyo que siempre me brindo durante mi etapa de estudiante en la ESPOL.

A mis amigos, que me acompañaron en mis años de estudio tendiéndome su mano cuando necesitaba de ellos.

Gracias.

DEDICATORIA

A mi familia y amigos que me brindaron apoyo en todo momento.

TRIBUNAL DE GRADUACIÓN

Ing. Washington Armas
DIRECTOR DEL ICM

Ing. Juan Alavarado
DIRECTOR DE TESIS

Mat. Fernando Sandoya
VOCAL

Ing. Carola Pino
VOCAL

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de esta Tesis de Grado, me corresponden exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL"

_____

Mónica Paola Rojas Ramírez

RESUMEN

La “ICE”, es la Industria Cartonera Ecuatoriana, que se dedica a la producción de láminas y cajas de cartón corrugado y microcorrugado para satisfacer las necesidades de embalaje de un gran número de productos, los que a su vez pueden pertenecer al mercado nacional como internacional dirigidos a diferentes sectores tales como camaronero, pesquero, bananero, floricultor, doméstico entre otros.

Actualmente la Industria Cartonera Ecuatoriana ha venido experimentando serios problemas en la producción, además lo más crítico es la falta del personal para formar turnos adicionales de producción.

La situación anteriormente expuesta a provocado que se incremente el porcentaje de incumplimiento de los tiempos de entrega de los pedidos llegando a niveles del 60 %, básicamente provocado por la planificación de la producción, ya que para cumplir con la mayoría de pedidos sobre todo con los imprevistos a última hora que suelen ser urgentes, se modifica la planificación de dicha producción, trayendo como consecuencia desperdicio de material y recursos, mayor tiempo y parada de máquinas.

El objetivo del presente trabajo consiste en demostrar que los profesionales en Estadística Informática pueden diseñar, desarrollar e implementar sistemas que optimicen la planificación de la producción, lo que será una herramienta de gran ayuda para el encargado de la planificación, mediante la implementación de algoritmos de búsqueda rápidas y exhaustivas dentro de las teorías de optimización combinatorias.

ÍNDICE GENERAL

ANEXOS

A. Programación Modular

B. Programación Modular de Clases

C. Creación de un Store Procedure

GLOSARIO

BIBLIOGRAFIA

ABREVIATURAS

ICE	Industria Cartonera Ecuatoriana
gr. / m2	Gramos por metros cuadrados
m.	Metros lineales
D.B.	Doble Backer
S.F.	Single Facer
Seg.	Segundos
Libras/ pulg2	Libras por pulgadas cuadradas
m. / min.	Metros por minuto
$pies^3$	Pies cúbicos
ASCII	American Standard Code International Institute
CD-ROM	Compaq Disc - Read Only Memory
DBMS	Database Management System
DSN	Data Source Name
GB	GigaByte
MB	Megabytes
Mhz	Megahertz
MTS	Microsoft Transaction Server
NT	Netware
ODBC	Open DataBase Conectivity
UNIX	Servidor
RAM	Random Access Memory
SVGA	Super Video Graphics Advanced
SQL	Structured Query Language
VB	Visual Basic
XP	Experience

SIMBOLOGÍA

$d(i)$	distancia más corta desde el nodo origen hasta el nodo i
$N(s)$	Conjunto obtenido de la permutación del conjunto S
E_i	Energía i
DE	Incremento en la Energía
$P(DE)$	Probabilidad de aceptar el incremento de la energía
$\hat{A}$	Conjunto de los Reales
E_0	Estado Fundamental
Ω	Conjunto de todos los posibles microestados
$card(\Omega)$	Numero de microestados posibles
U	Número aleatorio Uniforme
$U(0,1)$	Número aleatorio Uniforme entre 0 y 1
T	Temperatura
T_0	Temperatura Inicial
S_0	Solución Inicial
$C(S)$	Costo de la solución
d	Diferencia entre el costo de la solución candidata y la actual
$e^{-d/T}$	Factor de Boltzmann
$N(S_{act})$	Solución en la vecindad de la solución actual
f	Probabilidad de ser aceptada una solución (Factor Boltzmann)
m	Probabilidad de ser una solución peor que la anterior
$L(T)$	Número de iteraciones o Velocidad de Enfriamiento
$a(T)$	Decrecimiento de la Temperatura
e	Error Estimado
q	Probabilidad de obtener una solución cuyo costo menos el de la óptima sea e
T_f	Temperatura Final

INDICE DE FIGURAS

INDICE DE TABLAS

Pág.

INDICE DE ALGORTIMOS

INTRODUCCIÓN

La manera más antigua de realizar la díficil tarea de planificar una producción cumpliendo con restricciones de tiempo, ha sido siempre por medio de la experiencia del recurso humano especializado en dicha area, pero este puede cometer algún error o demorarse en tener una planificación óptima.

Así surge la necesidad de contar con una herramienta que permita la planificación óptima y rápida en la Producción Industrial (Asignación de pedidos a las unidades de trabajo)

Por ello el sistema implementado en esta tesis, es una alternativa a esa necesidad, que brinda funcionalidad a partir de información y herramientas de desarrollo de bajo costo y mayor disponibilidad; proveyendo además, mayor velocidad de procesamiento, logrando así un mayor alcance al momento de obtener los resultados ya que permite al usuario no solo utilizar una solución óptima sino también, seleccionar otra solución factible, entre varias soluciones factibles.

CAPÍTULO I

1. GENERALIDADES Y CARACTERISTICA DE LA INDUSTRIA

1.1 COMERCIALIZACIÓN DE PRODUCTOS DE LA INDUSTRIA

La Industria Cartonera Ecuatoriana (ICE) comercializa sus productos a través del Departamento de Comercialización, cuya oficina está ubicada dentro de la empresa.

El departamento de Comercialización a nivel nacional cubre las siguientes provincias:

- Guayas
- El Oro

- Manabí
- Esmeraldas
- Azuay
- Pichincha
- Imbabura
- Tungurahua
- Cotopaxi

La comercialización se la hace en forma directa con el cliente, por medio del ejecutivo de ventas.

A continuación se mencionan los mercados cubiertos por la ICE:

- Floricultor
- Bananero
- Doméstico
- Camaronero
- Pesquero
- No tradicionales
- Convertidores
- Otros

1.2 ANALISIS DEL PROCESO DE PRODUCCION

1.2.1 Descripción del producto

Los productos que maneja la ICE son básicamente los elaborados de cartón.

La empresa ofrece láminas y cajas en pared sencilla y doble en flauta (tipo de corrugado) C, B y BC; cajas regulares para el sector doméstico e industrial, cajas troqueladas y láminas de microcorrugado para el sector floricultor, cajas de doble pared que soportan grandes pesos para la exportación de productos agrícolas con o sin recubrimiento impermeabilizante, además hacen cajas laminadas, exhibidores y divisiones interiores de cajas (aditamentos).

La empresa comenzó con la elaboración de cajas en pared sencilla, luego con las cajas de doble pared, en función del crecimiento del mercado floricultor la empresa empezó a incursionar en la producción de láminas de microcorrugado y finalmente a la producción de láminas sencillas destinadas a los convertidores.

1.2.2 Características y Propiedades de los componentes del producto

El cartón está compuesto por los siguientes componentes:

- Papel liner
- Papel médium
- Almidón.
- Tinta.
- Goma.

Analizaremos a continuación brevemente la características mas notables de los componentes anteriormente mencionados:

Papel Liner: Puede ser de color blanco o kraft (café). El papel liner tiene dos lados, el lado lis forma la cara interna plana del cartón, el lado áspero la cara exterior. Tiene como característica ser de:

Fibra larga: permite que las fibras puedan amarrarse entre sí, aumentando la resistencia al papel, esto lo hace más resistente que el papel médium.

Se clasifica según su peso en gr. / m2 de la siguiente manera:

- 337 gr. / m2
- 300 gr. / m2
- 270 gr. / m2
- 205 gr. / m2
- 186 gr. / m2
- 125 gr. / m2

Los anchos de las bobinas de papel liner más conocidas y utilizadas son:

- 1.540 m.
- 1.610 m.
- 1.740 m.
- 1.780 m.
- 1.830 m.
- 1.880 m.
- 1.911 m.

En el Proceso de Producción se lo conoce de la siguiente manera:

i. **Liner D.B.:** Papel Liner exterior que se une al papel médium de flauta C o al médium de flauta B en el proceso de producción de la Sección Doble Backer de la corrugadora.

ii. **Liner S.F. "C":** Papel liner interior que se une al papel médium de flauta C en el proceso de producción de la Sección Single Facer de la corrugadora.

iii. **Liner S.F. "B":** Papel liner interior que se une al papel médium de flauta B en el procedo de producción de la Sección Single Facer de la corrugadora.

Papel Médium: Es de color kraft, y forma la flauta del cartón y se encuentra entre los liners. Tiene las siguientes características:

Fibra corta: Da facilidad al corrugarse debido a su flexibilidad Porosidad: Da facilidad de absorber la humedad, necesaria para ablandar y acondicionar el corrugado (formación de la flauta).

Flauta: Corrugado u Ondulación del papel médium, la misma que sirve de separador entre los liners. Hay diversos tipos de flautas dependiendo del número de ondas que se formen por

pulgadas y las más conocidas y utilizadas nacional e internacionalmente se muestran a continuación en la tabla 1.1

Flauta	flautas por pie lineal	cm. de profundidad
A	33	0.167
B	47	0.097
C	39	0.142
E	90	

Tabla 1.1: *Flautas*

Se clasifica según su peso en gr. / m2. Existen varios gramajes pero más utilizado en nuestro medio es el de 146 gr. / m2.

En el Ecuador, solo son comerciales las flautas B, C, E; pero la empresa ICE solo produce las dos primeras.

En el Proceso de Producción se lo conoce de la siguiente manera:

i. **Médium S.F. "C":** Es el papel médium que en el proceso de producción de la Sección Single Facer de la corrugadora S & S o Langston se convierte en corrugado de flauta C.
ii. **Médium S.F. "B":** Es el papel médium que en el proceso de producción de la Sección Single Facer de la corrugadora Langston se convierte en corrugado de flauta B.

Almidón: Sirve como medio de adhesión entre el papel liner y el médium. Esto gracias a una combinación de productos químicos, que le dan propiedades especiales al almidón formulado, las cuales son: impermeabilidad, penetración, absorción, consistencia, mayor fijación y adherencia, viscosidad, con lo que podemos incrementar la velocidad de producción. Estos productos son los siguientes:

- Aditivo o almidón modificado
- Soda Cáustica
- Bórax
- Resina

- Agua

Se tiene dos tipos de almidón necesarios en el proceso de producción: Almidón Primario y Secundario que se diferencian únicamente por la cantidad en gramos o kilogramos de los productos químicos a combinar.

Tinta: Es el elemento básico y necesario para obtener la impresión en la caja. El tipo de tinta utilizado en el proceso es la flexográfica, ya que tiene la propiedad de secado instantáneo, lo que ayuda a evitar manchas en la impresión.

Goma: Consigue el cierre de la caja, el mismo, que debe tener la propiedad de secamiento instantáneo (5 seg.). El tipo apropiado es la goma cascoréz y su viscosidad apropiada debe ser 90-120 segundos con viscosímetro Copa-Love.

1.2.3 Características y Propiedades del producto.

Las cajas de cartón corrugado pueden ser clasificadas según sus componentes, sus propiedades, sus formas y utilización:

a) Según el número de componentes (papel) hay dos tipos:

- **i.** **Pared Simple**, el mismo que está conformado por dos papeles liner y un médium (flauta C).
- **ii.** **Pared Doble**, que está conformado por tres papeles liner y dos médium (uno flauta B y otro flauta C).

b) Según las propiedades (test o combinación de papeles que le da la medida de resistencia de una caja, es el estallido en libras / pulg2).

En las tablas 1.2 y 1.3 se muestran los diferentes tipos de test y las combinaciones de los papeles liner y médium, según el tipo de pared.

Test	Liner	Medium	Liner
125	125	146-150	125
150	125	146-150	175-186
175	175-186	146-150	175-186
200	205	146-150	205
250	205	146-150	300-337
275	300-337	146-150	300-337

Tabla 1.2: *Test de Pared Simple*

Test	Liner	Médium	Liner	Médium	Liner
350	205	146-150	205	146-150	205
394	205	146-150	205	146-150	300-337
405	300-337	146-150	205	146-150	300-337

Tabla 1.3: *Test de Pared Doble*

c) Según su forma y utilización :

i. Cajas Banano:

- Caja 22 x U
- Caja US21 A
- Caja Europea

ii. Cajas Domésticas:

- Caja tipo Jaba
- Caja tipo bandeja
- Caja regular
- Caja regular autoarmable
- Caja regular flan cruzado
- Caja regular flauta invertida
- Caja flauta invertida flan cruzado
- Caja Bandeja autoarmable
- Caja regular con doble rayado

- Cajas para pollos
- Caja Telescopio

Para el proceso de producción según su forma y utilización cada tipo de caja se descompone en los siguientes productos y cada uno se producirá mediante 1unaorden de fabricación:

- Caja
- Tapa
- Fondo
- Refuerzo

1.2.4 Descripción del Proceso de Producción

a) Máquinas que intervienen en el Proceso de Producción:

- Corrugadora
- Imprenta
- Rayadora y Cortadora
- Troqueladora, y Guillotina Bobst
- Mezcladora y Batidora

- Caldero

i. **Corrugador:** Tiene una capacidad de diseño expresada en m. / min. de 7.800 cortes x hora en la producción de fondos para cajas de banano. Existen dos corrugadoras en esta empresa:

   - S & S
   - Langston

   S & S: Solo es usada para los corrugados de Pared Simple de flauta C

   Langston: Esta proporciona los corrugados de Pared Doble (flauta B y C) y los corrugados de Pared Simple de flauta C y de flauta B.

ii. **Imprenta:** Existen seis imprentas en esta empresa:

   - Hooper
   - United 1
   - United 2
   - United 3
   - Ward

- S & S

Hooper, United 1, United 2: se utiliza para programas de corridas largas como son los pedidos de banano, de los cuales se producen mas de 100.000 cajas, entonces no importa que se demore un cambio de 45 min. a 1 hora, de manera que si se compensan las corridas largas de 10 a 20 horas o más, según la cantidad de cajas.

United 3, S & S, Ward: Se usan para programas de corridas cortas como son los pedidos de doméstico (mercado local), de los cuales se producen 500, 2000,3000,4000 y hasta 10000 cajas, lo que ocasiona un cambio de programación y esto demora entre 15 a 20 min. máximo. La imprenta Ward es una máquina flexográfica de 4 colores, imprime dos cajas en una lámina y en caso de banano tres cajas en una lámina

Máquina Troqueladora y Guillotina Bobst.- Un proceso adicional, que depende del tipo de producto,

necesita de esta máquina, para realizar un troquelado especial que no lo puede hacer la imprenta, en el caso de cajas de pollo o de pizza, terminado el troquelado se pasa a la sección Guillotina y Aditamentos de esta máquina, par realizar divisiones interiores de las cajas, cortes simples de láminas. Una vez terminado este proceso ahora si se puede realizar el paso 12 del proceso de producción en imprenta.

Máquina mezcladora y batidora.-Tiene dos tanques primario y secundario para la preparación, almacenamiento y circulación del adhesivo y de la goma. El primero es para el engomado de corrugados de Pared Simple y el segundo para el engomado de corrugados de Pared Doble.

Caldero.- Para la dotación de vapor de agua al corrugador, el cual trabaja a 130 pies promedio y a una temperatura de 1700C promedio utilizando para esto el

bunker como combustible. Dos Compresores y secadores de aire de 350 pies^3 cada uno.

b) Proceso de Producción en Corrugadora:

1. Verificar que el papel a utilizarse tenga el gramaje (peso) y ancho requerido que se registra en la Hoja de Fabricación del Cartón Corrugado.

2. Revisar si la viscosidad del almidón es la adecuada para el proceso.

3. Las bobinas de papel son montadas en los elevadores del corrugador.

4. Se realiza el proceso de la sección Single Facer del corrugador:

a) Se preacondiciona el médium SF"C": como medio de ablandamiento para darle facilidad al corrugado (ondulación o formación de la flauta C).

b) Luego se ingresa el almidón y goma al rodillo corrugador, como medio adhesivo

c) Se pasa médium SF"C" por el rodillo corrugador para formar el corrugado y producir el papel engomado.

d) Por medio del rodillo de presión se une y pega el liner SF"C" precalentado al médium SF"C". En caso de que la orden de producción lo indique, antes de esta operación se aplica el recubrimiento Michelman sobre la cara lisa del papel liner.

e) Estos dos papeles unidos, son ubicados en la parte superior de la corrugadora donde al contacto con el medio ambiente se enfría.

f) Inspección de la flauta, consiste en chequear fractura, formación y adhesión de la flauta.

g) Se realiza los pasos: a, b, c, d, e y f para los papeles liner SF"B" y médium SF"B".

h) Se procede a engomar la parte descubierta de los médium SF"C" y médium SF"B".

5. Se transportan las uniones a la sección Doble Backer, del corrugador. En esta sección se engoma los Single Facer para unirlos cada uno con otro papel liner (exterior).

6. Se ingresa a la sección Plancha a Vapor, donde precalentadores ajustables secan la humedad del producto, estas planchas tienen una temperatura promedio de 3400F.
7. Formada la lámina pasa por la sección Triplex, donde un grupo de cuchillas realizan cortes longitudinales y rayados de acuerdo a la velocidad de la corrugadora.
8. Luego son cortadas de forma transversal en la sección Cut-Off
9. Por último se realiza la inspección de verificar las medidas y características del producto.

Este es un proceso de producción, cuando se trabaja una orden de Pared Doble. Cuando la orden requiere una Pared Sencilla, se elimina el material y proceso de un Single Facer.

c) Proceso de Producción en Imprenta:

1. Se prepara y procesa la sección Cuerpo de Impresión, que consiste en el montaje de clisés (sello de caucho) de

la impresora flexográfica con las especificaciones y características según orden de producción.

2. Se prepara y procesa la sección Cuerpo de Troquelador, que consiste en el montaje de los troqueles (perforadores de madera de medida estandarizada), para realizar el cambio de las medidas de una orden a otra.
3. Verificación de las medidas de la lámina según la Orden de Producción.
4. Inspección de: cantidad de tinta necesaria; viscosidad, color y tono de la tinta y viscosidad de la goma.
5. Se regula el pateador de láminas de acuerdo a las dimensiones de las mismas.
6. Se ingresa las láminas (material en proceso) en la sección Alimentadora de la imprenta; e inicia la corrida hasta conseguir que todas las especificaciones y características de la orden se cumplan.
7. Se procesa la sección Dobladora, realiza los dobles al producto lo que da facilidad para armarlo; se ingresa la

goma a la imprenta, lo que permite pegar los filos del producto para el cierre del mismo.

8. Se procesa en la Máquina Rayadora y Cortadora, para realizar los cortes y/o las perforaciones a la caja.
9. Inspección y verificación de: medidas, profundidad de las ranuras, calidad de cortes y perforaciones, buenos rayados y adhesión de la goma en el cierre y la impresión.
10. Se realiza la sección Embaladora, el producto se separa en bultos amarrados de 25 unidades con cinta plástica.
11. Se coloca en pallets los de bultos, la cantidad depende del tipo de producto y de las dimensiones del mismo.
12. Se estiba el producto, que consiste finalmente en el ingreso del producto terminado a la bodega.

d) Descripción del Flujo del Proceso del Producto

El proceso para obtener el cartón corrugado se empieza en la corrugadora a la cual se le introduce la materia prima necesaria y ésta realiza diversas operaciones hasta obtener

las láminas especificadas para una caja, a la que se le realizan múltiples inspecciones , luego generalmente son estibada en pallets y transportadas al área de material en proceso, donde están en espera, hasta el momento que según programa de producción, tenga que entrar a la imprenta.

En otros casos las láminas son llevadas directamente a la corrugadora hasta la imprenta omitiendo ser almacenadas temporalmente, esto es debido a que:

- No hay material en el área de producto en proceso
- El programa de producción así lo estima
- Por ser un producción de entrega urgente.

1.2.5 Pruebas de calidad en el proceso de producción:

a) **Pin Adhesión.-** Resistencia a la adhesión de la lámina, prueba para determinar si al sufrir daños por diferentes factores siguen adheridas las láminas.

b) Edge Crush.- Resistencia al aplastamiento de los bordes del cartón, es una prueba que nos permite determinar los golpes internos que puede recibir la caja.

c) Flat Crush.- Resistencia al aplastamiento horizontal de la lámina, es una prueba que nos permite determinar los golpes externos que puede recibir la caja. Conocido también como prueba de cóncora.

d) Calibre del papel.- Para determinar el espesor de la lámina, se realizan seis pruebas y el promedio obtenido de ellas es el calibre del papel.

e) Calibre del cartón.- Es el calibre o espesor del cartón corrugado, es aplicable a todas las combinaciones del cartón corrugado producidas en la ICE.

1.3 PROBLEMÁTICA DE LA INDUSTRIA

El mayor problema en una industria de estas características siempre ha sido, la difícil tarea de elaborar la más óptima planificación de la producción, y se hace más difícil aún cuando tenemos órdenes urgentes que cumplir, las mismas que no estaban previstas. Es así como se trunca la programación de un día para sacar dichas órdenes

En lo particular la ICE, no solo tiene este problema sino que también lo más crítico en los actuales momentos es la falta de personal, lo que se trata de suplir con personal eventual, el mismo que por no tener antigüedad, no tiene experiencia suficiente para arreglar las máquinas, y así surge un nuevo problema, la falla mecánica de las máquinas.

Por lo tanto no se puede confiar en un programa (planificación del proceso de producción para cada máquina) ya que no hay solución a problemas inmediatos.

De manera que en esta industria para llevar a cabo la producción se elabora una Hoja de Fabricación de Cartón Corrugado mediante órdenes de producción emitidas previamente y por cada una de éstas hojas se elabora una Hoja de Imprenta.

En estas hojas tanto de fabricación de cartón corrugado como de imprenta, se debe tener presente las causas de parada de las máquinas (corrugadora o imprenta).

a) Causas de Paradas en Corrugadora:

1. Corriendo

2. Sin programa

3. Cambio de medidas

4. Mantenimiento programado

5. Falla mecánica

6. Falla eléctrica

7. Falta de material

8. Falta de almidón preparado

9. Rotura y atoramiento de papel

10. Falta de montacargas

11. Limpieza

12. Preparación de máquina

13. Falta de vapor

14. Falta de pallets

15. Falta de empalme

16. Falta de espacio

17. Falta de presión de aire

18. Falta de personal

19. Varios

Sin programa: esto casi nunca sucede, solo se para a propósito porque esta lleno el piso (exceso de material) y se dice causa sin programa, esto ocasiona una parada de aproximadamente 4 hrs.

Cambio de medidas: Esto sucede por la inexistencia de una herramienta necesaria para que la Barra Triples tenga habilitada la tres barras, y así la parada seria solo de 2 minutos, actualmente funciona como barra mono (una barra) entonces hay mismo se cambia las medidas lo que ocasiona una parada de 15 a 20 minutos.

Mantenimiento programado: Se programa la máquina para una reparación preventiva, esto no quiere decir que la máquina este dañada, se trata de un mantenimiento para evitar daños mediante alguna corrida. Este mantenimiento es semanal para cada máquina y dura 1 día, lo mismo que es 8 hrs.

Falta de almidón preparado: En la cocina de almidón donde se prepara la goma, un rodillo coge la goma y va pegando al papel, cuando ésta se daña, no se puede producir.

Rotura y atoramiento de papel: Cuando se mete la bobina de papel en los rodillos corrugadores y éste se atora o se rompe se debe parar la producción y cambiar la bobina.

Falta de montacargas: Los montacargas transportan el material de máquina a máquina, en esta industria se tienen montacargas obsoletos, actualmente se dañan muy seguido de manera que se ven obligados a alquilarlos a compañías de afuera pero se lo hace en el momento que se daña uno. Si esto sucede en el día ocasiona parada de 1 hora, si sucede en la noche de 3 a 4 hrs.

Limpieza: Esto es muy normal, se lo hace cada vez que la máquina está sucia y esto es dos veces por turno (8 horas), ocasiona una parada de 20 minutos.

Preparación de máquina: Ocasiona una parada de 30 hrs. Máximo y esto es cuando arranca la máquina, cuando esta en marcha revisando el almidón que se pone en las bobinas, o cuando ha estado en mantenimiento, que es generalmente los lunes.

Falta de vapor: Esto es cuando se daña un caldero y ocasiona una parada de 2 a 3 horas porque todas las corrugadoras trabajan con vapor.

Falta de pallets: Cuando no hay flujo de retiro, se llenan las bodegas de material, no hay pallets desocupados por el despacho necesario, lo que ocasiona una parada de todas las maquinas durante 2, 3 o 4 horas.

Falta de empalme: La máquina empalmadora, coge las puntas de las bobinas nuevas y las une (empalma) automáticamente, cuando se daña se lo hace manualmente y eso ocasiona una demora de 15 minutos.

Falta de espacio: Esto es muy similar a la de falta de pallets

Falta de presión de aire: Cuando hay problemas en los calderos de vapor y compresores que producen aire, esto ocasiona una parada de ½ hora porque se tiene dos compresores, si se daña uno, está habilitado el otro. Todas las máquinas trabaja con aire.

Falta de personal: Esto es muy frecuente, por lo que se contrata personal eventual, que normalmente carece de experiencia.

Varios: Esto alguna causa no muy frecuente, detalles (especificaciones esporádicas) que casi nunca suceden como cambio de cuchillas, limpieza de dedos.

b) Causa de Parada en Imprenta:

1. Corriendo
2. Sin programa
3. Cambio de medidas
4. Mantenimiento programado
5. Falla mecánica
6. Falla eléctrica
7. Falta de material
8. Atoramiento
9. Falta de montacargas
10. Limpieza
11. Preparación de máquina
12. Falta de pallets
13. Falta de espacio
14. Falta de presión de aire
15. Falta de personal
16. Falta de troquel
17. Falla de clisé
18. Falla de tintas

19. Defecto y saneo de láminas

20. Varios

Falta de troquel: Troquel es una masa de madera que tiene cuchillos para cortar el material, y posee un molde (caucho o plástico donde va la impresión de las caja) cuando se daña, el troquel depende si es defecto de perforación lo que ocasiona una parada de 30 minutos, o si es daño total lo que ocasiona 2 horas.

Falla de clisé: Esto puede ser solo de limpieza, cuando el clisé esta sucio, o por cambio de alguna pieza pequeña lo que ocasiona una parada de 10 a 15 minutos pero si esta dañado completamente y no hay repuesto hay que cambiar de corrida (programación) y cuando eso ocurre deja de ser falla de clisé y pasa a ser cambio de medidas.

Falla de tintas: Cuando falla el tono de la tinta, el tono programado (el que requiere el cliente) y cuando no hay el tono ideal se acondiciona hasta llegar al tono ideal, lo que demora 15 minutos.

Defecto y saneo de láminas: Esto es cuando de corrugadora sale el material con falla (no es muy frecuente), en la imprenta se detecta el material con falla por lo que se sanea el material, es decir eliminar hojas malas, y utilizar la buenas, la demora depende de la cantidad de hojas malas que se encuentren.

CAPÍTULO II

2. OPTIMIZACIÓN COMBINATORIA

2.1 RESUMEN

La optimización combinatoria es un campo vivo de la matemática aplicada, que estudia el modelado combinando técnicas de combinatoria, programación lineal, lineal-entera, la teoría general de problemas extrémales y la teoría de algoritmos, con el propósito de encontrar una solución algorítmica de problemas donde se busca optimizar (maximizar o minimizar) una función de varias variables definidas sobre un conjunto discreto.

Esta disciplina tiene numerosas aplicaciones a problemas que se presentan en la industria, logística, ciencias, ingenierías y en la administración de organizaciones.

La versatilidad del modelo de la optimización combinatoria resulta del hecho en que muchos problemas prácticos, actividades y recursos, tales como máquinas, aeroplanos y personas son indivisibles. También, muchos problemas tiene únicamente un número finito de alternativas y consecuentemente pueden apropiadamente ser formuladas como problemas de optimización combinatoria (la palabra combinatoria se refiere al hecho que únicamente existen un número finito de soluciones factibles).

Como ejemplos podemos mencionar, entre otros, el ruteo y carga de vehículos en redes de distribución, el diseño de redes de telecomunicación, la planificación de la producción, la selección de carteras financieras, la asignación de tareas a procesadores, el análisis de estructuras moleculares, las subastas de frecuencias para radiotransmisión, la asignación de tripulaciones en líneas aéreas, la planificación de la generación de electricidad y la distribución de ambulancias en una región para asegurar un cierto nivel de servicio a su población. La gran variedad de posibles

aplicaciones es una fuente constante de problemas nuevos para la investigación en este área.

El término combinatorio se entiende en el sentido de que el conjunto factible es de gran talla

Un problema de programación lineal consiste en hallar el valor óptimo de una función objetivo lineal cuyas variables están sujetas a restricciones lineales. Si además se exige que las variables tomen valores enteros, entonces se tiene un problema de programación lineal-entera.

Varios problemas de Optimización Combinatoria pueden ser resueltos en tiempo polinomial utilizando los métodos y teoría de la programación lineal.

En el caso de los problemas NP-Completos, los métodos más eficaces en la actualidad para encontrar una solución exacta a muchos de estos problemas utilizan, típicamente, la programación lineal-entera o técnicas de ramificación y acotamiento y generación de columnas.

2.2 COMPLEJIDAD COMPUTACIONAL

Las Ciencias Computacionales tratan en mayor o en menor grado problemas de "complejidad computacional", que informalmente se puede describir como el costo requerido para encontrar la solución a un problema en términos de recursos computacionales (fundamentalmente memoria o tiempo de cómputo).

Los procedimientos desarrollados para la solución de problemas son llamados algoritmos, y la complejidad computacional, trata de encontrar los algoritmos más eficientes para resolver cada problema. La eficiencia hace referencia a todos los recursos computacionales, sin embargo la eficiencia puede ser tomada como sinónimo de rapidez.

La teoría de complejidad plantea los problemas como casos de decisión para los cuales su solución corresponde a una respuesta SI/NO

2.3 PROBLEMAS P, NP Y NP-COMPLETOS

Para los problemas de carácter combinatorio (donde el dominio de soluciones es finito y se elige la mejor solución posible de éste), existen distintas formas de resolverlos, una de ellas es la búsqueda exhaustiva del conjunto de soluciones y así poder encontrar la óptima, es decir generar todas las soluciones factibles (que cumplan con todas las restricciones), calcular su costo respectivo asociado y de éstas elegir la mejor. Pero el tiempo de cálculo crece de manera exponencial de acuerdo al número de items del problema.

Podemos encontrar problemas en que se produce una explosión combinatoria (donde el tiempo de ejecución es no polinomial), de acuerdo al tamaño del problema, de los que solo se conocen algoritmos que encuentran una solución exacta en tiempos excesivamente largos.

Cuando nos enfrentamos a un problema concreto, habrá una serie de algoritmos aplicables. Se suele decir que el orden de complejidad de un problema es el del mejor algoritmo que se conozca para resolverlo. Así se clasifican los problemas, y los

estudios sobre algoritmos que se aplican a la realidad. Estos estudios han llevado a la constatación de que existen problemas muy difíciles, problemas que desafían la utilización de los ordenadores para resolverlos. En lo que sigue esbozaremos las clases de problemas que hoy por hoy se escapan a un tratamiento informático.

Clase P. Los algoritmos de complejidad polinómica se dice que son tratables en el sentido que suelen ser abordables en la práctica. Los problemas para los que se conocen algoritmos con esta complejidad se dice que forman la clase P. Aquellos problemas para los que la mejor solución que se conoce es de complejidad superior a la polinómica, se dice que son problemas intratables. Seria muy interesante encontrar alguna solución polinómica (o mejor) que permitiera abordarlos.

Clase NP. Algunos de estos problemas intratables pueden caracterizarse por el curioso hecho de que puede aplicarse un algoritmo polinómico para comprobar si una posible solución es válida o no. Esta característica lleva a un método de resolución no determinista consistente en aplicar heurísticos para obtener soluciones hipotéticas que se van desestimando (o aceptando) a

ritmo polinómico. Los problemas de esta clase se denominan NP (la N de no-deterministicos y la P de polinómicos). Es evidente que **P** $\subseteq$ **NP**.

Clase NP-completos. En 1971 Cook demostró que hay problemas en N**P** que son especialmente difíciles , son los denominados NP-completos. Se conoce una amplia variedad de problemas de tipo NP, de los cuales destacan algunos de ellos de extrema complejidad. Gráficamente podemos decir que algunos problemas se hallan en la "frontera externa" de la clase NP. Son problemas NP, y son los peores problemas posibles de clase NP. Estos problemas se caracterizan por ser todos "iguales" en el sentido de que si se descubriera una solución P para alguno de ellos, esta solución sería fácilmente aplicable a todos ellos.

Clase NP-duros. Cualquier problema de decisión, pertenezca o no a los problemas NP, el cual pueda ser transformado a un problema NPC (NP-completo) tendrá la propiedad que no podrá ser resuelto en tiempo polinomial a menos que P=NP. Podríamos entonces decir que dicho problema es al menos tan difícil como uno NP Completo.

2.4 ALGORITMO DE CAMINOS MAS CORTOS (DIJKSTRA)

Dijkstra es un algoritmo que encuentra caminos más cortos desde un nodo origen s hacia todos los nodos en una red, con pesos no negativos en los arcos, es decir encuentra una arborescencia de caminos más cortos.

El seudocódigo del Algoritmo de Dijkstra se muestra a continuación en el Algoritmo 2.1, donde se comienza etiquetando todos los nodos como infinito y llamando a esta etiqueta $d(i)$ para cada nodo i, la cual representa la distancia más corta desde el nodo origen hasta el nodo i, a excepción del nodo s que se etiqueta con 0. Se tiene además que identificar dos conjuntos S y $\overline{S}$ que representan los nodos etiquetados definitivamente y los nodos etiquetados temporalmente respectivamente, estos conjuntos van cambiando de acuerdo a cada iteración del Algoritmo según el siguiente criterio: Sea $i \in \overline{S}$ tal que $d(i) = \min\{d(j): j \in \overline{S}\}$, acuatizándose así los dos conjuntos S y $\overline{S}$. El criterio de parada es cuando todos los nodos están etiquetados definitivamente.

Algoritmo 2.1: ALGORITMO DIJKSTRA

BEGIN

$S := f; \overline{S} := N;$

$d(i) := \infty$ *para cada nodo* $i \in N;$

$d(s) := 0$ *y* $predecesor(s) := 0;$

WHILE $|S| < n$ *DO*

BEGIN

Sea $i \in \overline{S}$ *tal que* $d(i) = \min\{d(j): j \in \overline{S}\};$

$S := S \cup \{i\};$

$\overline{S} := \overline{S} - \{i\};$

FOR EACH $(i, j) \in A(i)$ *DO*

IF $d(j) > d(i) + c_{ij}$ *THEN* $d(j) := d(i) + c_{ij}$ *y* $predecesor(j) := i;$

END;

END;

END;

END;

END;

2.5 HEURISTICAS

En los últimos años ha habido un crecimiento espectacular en el desarrollo de procedimientos heurísticos para resolver problemas combinatorios. Este hecho puede ser constatado examinando el gran número de artículos en revistas de Investigación Operativa en los que se proponen y estudian métodos heurísticos. Además, han aparecido publicaciones específicas para el estudio y divulgación de dichos procedimientos tales como Journal of Heuristics.

El auge que experimentan los procedimientos heurísticos se debe sin duda a la necesidad de disponer de herramientas que permitan ofrecer soluciones rápidas a problemas reales. Es importante destacar el hecho de que los algoritmos heurísticos (por sí solos) no garantizan la optimalidad de la solución encontrada, aunque su propósito es encontrar una solución cercana al óptimo en un tiempo razonable. Sin embargo, la gran cantidad de publicaciones en donde problemas de gran dificultad son resueltos con gran rapidez (en muchos casos óptimamente), avalan estos métodos.

Son procedimientos simples, a menudo basados en el sentido común, que se supone ofrecerán una buena solución (aunque no

necesariamente la óptima) a problemas difíciles, de un modo fácil y rápido.

Los factores que pueden hacer interesante la utilización de heurísticas para la resolución de un problema pueden ser los siguientes:

1. Cuando no existe un método exacto de resolución o éste requiere mucho tiempo de cálculo o memoria.
2. Cuando no se necesita la solución óptima
3. Cuando los datos son poco fiables
4. Cuando hay limitaciones de tiempo
5. Cuando es paso intermedio en la aplicación de otro algoritmo

Por otro lado la implementación de heurísticas presenta ciertos inconvenientes una de ellas es que por lo general no es posible conocer la calidad de la solución, es decir cuan cerca está del óptimo.

2.5.1 Clasificación de Heurísticas

Según el modo en que buscan y construyen sus soluciones, podemos clasificarlas de la siguiente manera:

a) Métodos constructivos

Consisten en ir poco a poco añadiendo componentes individuales a la solución hasta que se obtiene una solución factible. El más popular de éstos métodos lo constituyen los algoritmos “Glotones”, los cuales construyen paso a paso la solución buscando el máximo beneficio en cada paso.

b) Métodos de descomposición

Este tipo de algoritmo trata dividir el problema en subproblemas más pequeños, siendo la salida del uno la entrada del siguiente, de forma que al resolverlos se obtenga una solución para el problema global.

c) Métodos de reducción

Tratan de identificar alguna característica que pueda poseer la solución óptima y así hacer más simple el problema, por

ejemplo si algunas variables están altamente correlacionadas o si tomarán el valor de 0 siempre, etc.

d) Métodos de búsqueda por entornos

Aquí se encuentran la mayoría de las heurísticas, parten de una solución factible inicial (obtenida quizás mediante otra heurística) y mediante modificaciones de esa solución, van pasando de forma iterativa, y mientras no se cumpla algún criterio de parada determinado, a otras factibles de su entorno o vecindad, almacenando como óptima la mejor de las soluciones visitadas.

Definición 2.4.- Entorno o Vecindad de la solución *S*

Es el conjunto $N(s)$ de soluciones parecidas a s. El significado de "parecidas" debe entenderse como la posibilidad de obtener una solución $s' \in N(s)$ a partir de la s realizando solo una operación elemental, llamada movimiento, sobre s (eliminar o añadir un elemento en el subconjunto solución, intercambiar elementos en la permutación s, etc.).

2.6 HEURISTICAS DE ÉXITO (META-HEURISTICAS)

Dado que estos métodos generales sirven para construir o guiar el diseño de métodos que resuelvan problemas específicos se les ha dado el nombre de Meta heurísticos. Los profesores Osman y Kelly (1995) introducen la siguiente definición:

"Los procedimientos Meta heurísticos son una clase de métodos aproximados que están diseñados para resolver problemas difíciles de optimización combinatoria, en los que los heurísticos clásicos no son ni efectivos ni eficientes. Los Meta heurísticos proporcionan un marco general para crear nuevos algoritmos híbridos combinando diferentes conceptos derivados de: inteligencia artificial, evolución biológica y mecanismos estadísticos"

- Algoritmos Genéticos
- Búsqueda Tabú
- Redes Neuronales
- Templado Simulado

a) Algoritmos Genéticos

Son técnicas de búsqueda basadas en la mecánica de la selección natural y la genética [Goldberg, 1989]. Su estructura

se ha diseñado en un intento de abstracción artificial del algoritmo de selección propio de la naturaleza, con la esperanza de que así se consigan éxitos similares en relación con la capacidad de adaptación a un amplio número de ambientes diferentes.

La información hereditaria en los organismos biológicos es pasada a través de los cromosomas que contienen la información de todos esos factores, es decir, los genes, los cuales a su vez están compuestos por un determinado número de “valores”. Varios organismos se agrupan formando una población, y aquellos que mejor se adaptan son aquellos que tienen mayor probabilidades de sobrevivir y reproducirse. Algunos de los supervivientes son seleccionados para crear nuevos organismos. Además los genes de un cromosoma pueden sufrir cambios produciendo mutaciones en los organismos.

En los algoritmos genéticos los “valores” por lo general son binarios (0 o 1), representando valores de las variables de decisión, que se correspondan a los genes. Los cromosomas

representan a las soluciones, en otras palabras los cromosomas no son más que una cadena de bits.

Un algoritmo genético aplica operaciones de reproducción, sobrecruzamiento y mutación a una población y generar nuevos organismos de forma iterativa. La reproducción se guía a través de una función que mide el grado de adaptación del cromosoma y el proceso de selección es dependiente de los valores que esa función le asigne: a mayor valor, mayor probabilidad de selección y supervivencia y los miembros de la nueva selección de cromosomas que son emparejados de nuevo aleatoriamente. Para sobre cruzar se intercambian genes entre las parejas, dividiendo cada una de las cadenas de bits en dos subcadenas y luego intercambiar valores de las últimas subcadenas, obteniendo así dos nuevos cromosomas de una nueva generación. Las mutaciones consisten en alterar un bit de un cromosoma (de 0 a 1 o viceversa), con una probabilidad mas o menos pequeña. Todo esto se repite hasta que algún criterio de parada se cumpla.

b) Búsqueda Tabú

Es un tipo de búsqueda por entornos. Guía un procedimiento de búsqueda local para explorar el espacio de soluciones más allá del óptimo local [Glover, 1994]. La búsqueda Tabú selecciona de una manera agresiva el mejor de los movimientos posibles a cada paso. Al contrario que en la búsqueda local que siempre se mueve al mejor de su entorno y finaliza con la llegada a un óptimo local, la búsqueda Tabú permite moverse a una solución de su entorno o vecindad que no sea tan buena como la actual, de tal manera que pueda tener oportunidad de salir de óptimos locales y continuar estratégicamente la búsqueda de soluciones aún mejores.

Para evitar ciclos (volver a un óptimo local) clasifica un determinado número de los más recientes movimientos como "movimientos Tabú". Por lo tanto el escape de los óptimos locales se produce de manera sistemática y no aleatoria. Es decir la búsqueda Tabú mantiene una memoria de eventos. Actualmente, el método consiste en tratar de explotar la memoria adaptativa para así controlar el proceso de búsqueda.

La filosofía de la búsqueda Tabú es la creencia de que la elección de una mala estratégica sistemática de búsqueda es mejor que una buena elegida al azar.

c) Redes Neuronales

Surgieron dentro del proceso de tratar de entender el comportamiento del cerebro humano, viéndolo como un sistema de procesamiento de la información de tipo altamente complejo, no lineal y en paralelo. Los modelos de redes neuronales intentan conseguir unos buenos resultados basándose en una interconexión de unos nodos computacionales llamados neuronas.

"Es un procesador distribuido paralelo que posee una propensión natural para el almacenamiento de conocimiento experimental haciéndolo posible para su uso. Recuerda al cerebro humano en dos aspectos: el conocimiento se adquiere mediante un proceso de aprendizaje, y la conexión interneuronal se utiliza para el almacenamiento del conocimiento" [Aleksander y Morton, 1990].

Otra de las heurísticas muy conocidas y que tiene ganado un lugar importante en la resolución de problemas difíciles es el Recocido Simulado, el que se implementará para el desarrollo de esta tesis. Es por esta razón que hemos profundizado la teoría de éste algoritmo en el capítulo 3, desde sus orígenes en la Mecánica Estadística hasta su forma Actual.

CAPÍTULO III

3. RECOCIDO SIMULADO (SIMULATED ANNEALING)

3.1 INTRODUCCION

En los primeros años de la década de los 80 quedó definido el Recocido Simulado, una heurística (Algoritmo de Aproximación) para la resolución de problemas de alta complejidad en el campo de la Combinatoria. Lo que puede sorprender es el hecho de que es producto de una analogía entre este tipo de problemas y resultados teóricos obtenidos en un campo tan distinto como lo es la Termodinámica.

Aunque recientemente ha aparecido en la Literatura científica en Español el término “Temple Simulado” para ésta heurística, debe aclararse que el proceso físico de “Annealing” es “Recocido”, la traducción de “Temple” al Inglés es “Hardening” o “Quenching” que

dentro del proceso de formación molecular tiene unos objetivos totalmente distintos a los del recocido.

3.2 RESUMEN

El algoritmo heurístico denominado Recocido Simulado, (figura 3.1) es un método global de optimización combinatoria establecido a través de una analogía al proceso físico de recocido de sólidos. Cuando a un material fundido se le baja la temperatura muy lentamente sus partículas se agrupan en un arreglo donde la energía interna del sistema es mínima. Metrópolis (1953) desarrolló un algoritmo para simular la evolución de un sólido en un baño de calor, a una temperatura específica, a través de una caminata estocástica entre configuraciones de átomos. Esto es, el sistema pasa sucesivamente de una configuración a otra mediante el siguiente criterio: Estando en una configuración, *i*, con energía E_i, el sistema pasa a una nueva configuración, *j, si* $DE \pounds 0$ *donde* DE = *Ej – Ei; o si P(DE) > r, donde P(DE) = exp {(Ei – Ej) / kB t}* es la probabilidad de aceptar el incremento DE en la energía del sistema *(kB = 1.38 x 10-16 y t = temperatura) y r* $\hat{I}\ \hat{A}$ *con 0* $\pounds$ *r* $\pounds$ *1*. La caminata se efectúa hasta el equilibrio térmico.

Usando la función de costo Z(xi) como la energía del sistema y definiendo la configuración de átomos mediante los valores del conjunto de variables {xi} se establece la analogía al sistema termodinámico. Mediante el algoritmo de Metrópolis se puede generar una sucesión de configuraciones en alguna temperatura efectiva, c = kB t, la cual no es otra cosa que un parámetro de control. Si c es descendido entre ejecuciones del algoritmo de Metrópolis se simula un proceso de recocido y las configuraciones generadas a lo largo del procedimiento tienden a la solución de costo mínimo.

3.3 ALGORITMO DE METRÓPOLIS

El algoritmo de Metrópolis (1953) el cual es muy conocido en la Física Química es la base del Recocido Simulado. Existen dos métodos de solución numérica en Mecánica-Estadística para estudiar el comportamiento microscópico de los cuerpos el método de “Dinámica Molecular” y el método de “Montecarlo”, pero es este último con el que se estudian usualmente las propiedades de equilibrio, y fue éste el utilizado por Metrópolis para su algoritmo.

El origen Químico-Físico de este procedimiento es el siguiente. Dada una sustancia cualquiera, no todas sus moléculas tienen la misma energía, sino que éstas se encuentran distribuidas en distintos niveles y el menor de los cuales se denomina “Estado Fundamental” e_0. Si la sustancia está a $0^o K$ todas las moléculas están en su estado fundamental, y de acuerdo al incremento de temperatura, las moléculas van ocupando niveles superiores. Cada una de las maneras en que las moléculas pueden estar distribuidas en los distintos niveles se denomina “Microestado”. Al conjunto de todos los posibles microestados se llama Ω y al número de partículas en el nivel i se denomina “número e ocupación n_i.

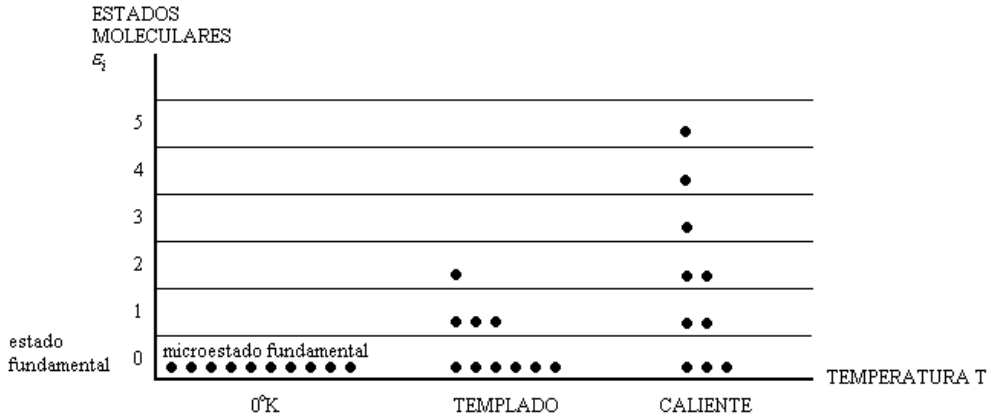

Figura 3.1 *Ejemplo de diferentes Microestados según la temperatura, para* $N = 10$. *En el Microestado correspondiente a TEMPLADO, es* $n_0 = 6, n_1 = 3, n_2 = 1, E = \sum n_i i = 5$. *El número de moléculas decrece exponencialmente para una* T *fija.*

El físico austriaco L. Boltzmann observó que cuando un sistema aislado evoluciona espontáneamente, además de aumentar la entropía, se produce un mayor desorden desde el punto de vista mecánico.

El problema era calcular dado un número de moléculas N y la energía E de la sustancia, de cuántas maneras pueden estar ubicadas las partículas en los distintos niveles, es decir cuantos estados diferentes $card(\Omega)$ pueden existir. Según la Estadística de Maxwell-Boltzmann (la cual considera las partículas totalmente discernibles), $card(\Omega)$ es igual al número de microestados posibles considerando fijos unos números de ocupación $\langle n_0', n_1', ..., n_i' \rangle$ (los cuales representan la complexión más probable en el equilibrio, dados N y E), siendo $n_j' = P \exp(-be_j)$ con un P factor de proporcionalidad, $b = 1/kT$, y $k = 1{,}38054\, x10^{-23}\, J/^0 K$ una constante.

Estrictamente hablando, todo lo anterior deja de ser aplicable en el momento en que se introducen las iteraciones entre partículas ya que las energías e_j no tienen significado físico alguno. W. Gibbs resolvió este inconveniente introduciendo el concepto de

“Colectivo”, una entidad formada por un número suficientemente grande de “Configuraciones”, cada una de ellas constituida por N partículas.

Metrópolis quería definir un algoritmo que permitiera calcular valores medios de la energía interna y de la presión dentro del formalismo de los colectivos de Gibbs, sin necesidad de recurrir a la solución analítica de las correspondientes integrales. Entonces fue cuando utilizó el conocido método de Montecarlo a la hora de elegir las n configuraciones que serían elegidas para calcular los valores medios buscados. En la elección de esas n configuraciones tienen en cuenta la probabilidad exponencial de que se presente una configuración determinada con energía potencial E_i.

Algoritmo 3.1 : ALGORITMO DE METROPOLIS

---

1. Se colocan las N partículas de la configuración en un cuadrado de arista L, cada una con coordenadas $\langle x_i, y_i \rangle$. Se define una longitud máxima de desplazamiento en un movimiento, b

2. Se calculan dos números aleatorios U_1, U_2 uniformes en el intervalo $[-1,1]$

3. Se calcula la nueva posición de la partícula: $x_{i+1} = x_i + bU_1$, $y_{i+1} = y_i + bU_2$ (en caso de que se excedan los límites del cuadrado, se entraría por el otro lado)

4. Se calcula la diferencia de energía d de $\langle x_{i+1}, y_{i+1} \rangle$ respecto a $\langle x_i, y_i \rangle$

5. Se elige la nueva configuración según lo siguiente:

5.1. Si $d < 0$, se acepta directamente $\langle x_{i+1}, y_{i+1} \rangle$ como complexión por considerar

5.2. Si no, se acepta $\langle x_{i+1}, y_{i+1} \rangle$ con una probabilidad $p = e^{-d/(kT)}$ (para lo cual se genera un $U' \in [0,1]$ y se acepta $\langle x_{i+1}, y_{i+1} \rangle$ si $U' < p$, caso contrario nos quedamos con $\langle x_i, y_i \rangle$

6. Se vuelve al paso 2 para calcular la siguiente configuración a partir de la actual

---

3.4 ANALOGIA ENTRE TERMODINAMICA Y OPTIMIZACION

A principios de los 80 Kirkpatrick (1983) en el diseño de circuitos electrónicos y de modo independiente Cerny (1985) investigando el "Traveling Salesman Problem", aplicaron el algoritmo de metrópolis en algunos problemas de minimización en optimización combinatoria. Es evidente que si lo hicieron tuvieron que crear analogías entre el proceso de simulación termodinámica y los procesos de Optimización, como se muestra en la tabla 3.1 .

Termodinámica		**Optimización**
Configuración	**Analogía**	Solución Factible
Configuración Fundamental	**Analogía**	Solución Optima
Energía de la Configuración	**Analogía**	Costo de la Solución
Temperatura	**Analogía**	?

Tabla 3.1: *Analogía entre Termodinámica y Optimización*

Al concepto físico de temperatura no le corresponde un significado real en el campo de optimización, es entonces un parámetro que tiene que ser ajustado. Se pueden imaginar entonces de manera similar los procesos que ocurren cuando las moléculas de una sustancia van colocándose en los diferentes niveles energéticos buscando el equilibrio a una determinada temperatura, y los que ocurren en optimización. Si se fija la temperatura, las partículas en los distintos niveles siguen una distribución de Boltzmann, entonces cuando una molécula se mueve, ese movimiento será aceptado en la simulación si la energía disminuye, o con una probabilidad proporcional al factor de Boltzmann en caso contrario.

Análogamente en Optimización, si fijamos el parámetro T, se produce una perturbación, aceptando directamente la nueva solución cuando su costo disminuye, o bien con una probabilidad proporcional al factor de Boltzmann en caso contrario.

Como la elección de una solución vecina se selecciona aleatoriamente, este tipo de estrategias presenta el inconveniente de que si durante el proceso de búsqueda se cae en un óptimo local, la técnica no sería capaz de salir de él, para poder evitar esto el recocido simulado nos permite el paso a soluciones peores pero

cada vez con menor probabilidad de acuerdo nos vayamos acercando a la solución óptima.

Si analizamos el factor de Boltzmann en función de la temperatura, vemos según la figura 3.2, que de acuerdo disminuye ésta, también disminuye la probabilidad de un movimiento a una nueva solución peor que la anterior. Recordemos que d es la diferencia entre la energía de $\langle x_{i+1}, y_{i+1} \rangle$ respecto a $\langle x_i, y_i \rangle$.

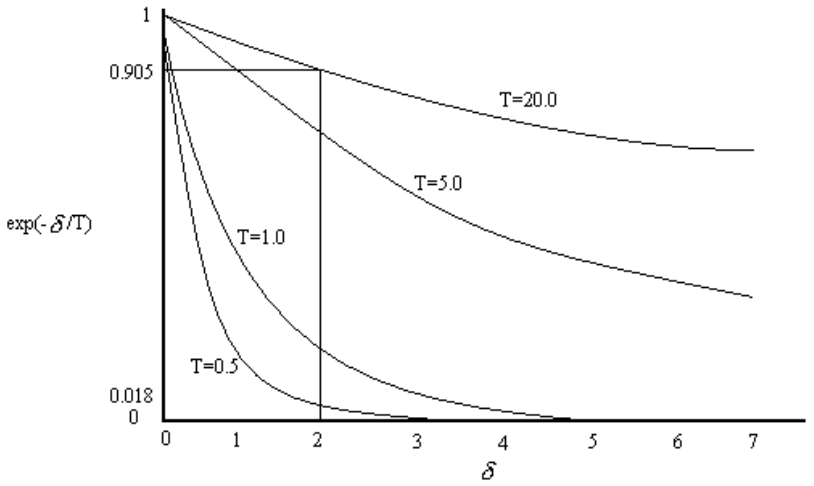

Figura 3.2 *Valor del Factor de Boltzmann en función de la temperatura* T *y de* d *. Para* $d = 2$*, es 50 veces menos probable un movimiento si* $T = 0{,}5$*, que si* $T = 20$

3.5 ESTRATEGIAS PARA EL RECOCIDO SIMULADO

Lo que haremos será lo siguiente:

Primero partir de una “temperatura” alta, con lo cual se permitirá pasos a soluciones peores con mayor probabilidad en los primeros pasos cuando estemos aún lejos de la solución óptima. Luego se irá reduciendo la temperatura, disminuyendo así la probabilidad de pasar soluciones peores cuando estemos cerca del óptimo.

Algoritmo 3.2: ALGORITMO RECOCIDO SIMULADO

$INPUT(T_0, a, L, T_f)$

$T \leftarrow T_0$

$S_{act} \leftarrow Genera\_solución\_inicial$

$WHILE\ T \geq T_f\ DO$

$BEGIN$

$FOR\ \ cont \leftarrow 1\ TO\ L(T)\ DO$

$BEGIN$

$S_{cand} \leftarrow Selecciona\_solución\_N(Sact)$

$d \leftarrow \cos to(S_{cand}) - \cos to(S_{act})$

$IF\ U(0,1) < e^{-d/T}\ \ OR$

$d < 0\ THEN\ S_{act} \leftarrow Scand$

END

$T \leftarrow a(T)$

END

$\{Escribe\_como\_soluión,\_la\_mejor\_de\_las\_S_{act}\_visitadas\}$

De acuerdo al algoritmo descrito arriba, se selecciona una temperatura inicial T_0, la velocidad de enfriamiento, es decir, cuanto va a disminuir T_{i+1} con respecto a T_i respecto al parámetro a, luego de estar $L(T)$ iteraciones en esa temperatura T, y una temperatura final T_f. Se genera una solución inicial que pertenece al espacio de soluciones Ω y mientras no se llegue al fin del proceso, para cada T se calcula un numero $L(T)$ de veces, antes de disminuir la temperatura, una solución que esté en la vecindad $N(S_{act})$de la actual, la cual sustituirá a ésta si tiene por su puesto menor costo, o con una probabilidad $e^{-d/T}$ (para está probabilidad se debe generar un número aleatorio uniforme $U(0,1)$). Y finalmente la mejor de todas las S_{act} visitadas será nuestra solución.

En el artículo original de Kirkpatrick (1983) la condición de parada (estado frío) consistía en que en tres temperaturas sucesivas no se produjese un número mínimo aceptable de mejoras de la solución.

3.6 PROGRAMA DE ENFRIAMIENTO

3.6.1 Introducción

Un programa de enfriamiento no tiene reglas generales, la elección de los parámetros correspondientes depende de las ideas generales asociadas a éstos.

Muchos han trabajado en el programa óptimo de enfriamiento como por ejemplo: S. Geman y D. Geman, B. Hajek [1988], B. Gidas y Hawng, mientras que al mismo tiempo D. Mitra y A. Sangiovanni-Vicentelli encontraron cotas (para tiempos finitos) a la distancia entre una distribución estacionaria a esa misma temperatura.

Uno de los primeros intentos de automatizar el programa de enfriamiento fue el de S.R. White [1984], quien eligió como la

máxima temperatura que se debía utilizar aquella que es proporcional a la desviación típica del costo de la función objetivo, muestreada en configuraciones tomada al azar.

3.6.2 Temperatura Inicial, T_0

Una de las características que debe cumplir una heurística es que no dependa de la solución de inicio o de partida. Como hemos visto esto lo consigue el Recocido Simulado por medio de elegir una temperatura inicial alta, con lo cual esta claro que al principio recorrerá soluciones lejanas a la óptima.

No es conveniente considerar para T_0 valores fijos independientes del problema, esto parece claro en el siguiente sentido, por ejemplo si la solución inicial S_0 tiene costo $C(S_0)$, y la seleccionada aleatoriamente en su vecindad o entorno un costo de $C(S_{cand}) = C(S_0) + 100$ ¿es entonces aceptable aceptar el candidato?, está claro que la respuesta depende del orden de magnitud de los costos: si por ejemplo $C(S_0)$ vale 0.01, no parece conveniente el movimiento, pero si es del orden de 10^6, podríamos entonces elegir al candidato ya que el valor de 100 no hace una gran diferencia en este caso, incluso para valores pequeños de T .

Podemos, entonces determinar T_0 en función del porcentaje de aceptación deseable. Consideremos que pudiera ser aceptable con un f de probabilidad una solución que sea un m por uno peor que la inicial S_0. Entonces, si el $C(S_{cand}) = (1+m)C(S_{act})$, por lo tanto $C(S_{cand}) - C(S_{act}) = mC(S_{act})$, y tenemos luego:

$$f = e^{-d/T_0} = e^{(-m/T_0)C(S_{act})} \qquad \Rightarrow \qquad T_0 = \frac{m}{-\ln(f)} C(S_{act})$$

Para nuestro problema hemos considerado razonable dentro de la lógica aceptar un $f = 13\%$ de las veces (en la primera iteración) una solución que sea un $m = 1\%$ peor que la actual, tendríamos entonces un:

$$T_0 = \frac{0.01}{-\ln(0.13)} C(S_{act}) \approx 0.005 C(S_{act})$$

3.6.3 Velocidad de Enfriamiento

$L(T)$ debería ser lo suficientemente grande para que el sistema llegue a alcanzar su estado estacionario para esa temperatura T, lo cual significaría un número de iteraciones al menos igual a $card(\Omega)^2$ [Eglese, 1990] . Como esto no es factible, al definir el algoritmo hay que lograr un equilibrio entre los parámetros a y L, ya que si se realiza una búsqueda durante pocas iteraciones para cada temperatura, son recomendables rápidos decrecimientos de temperatura y viceversa.

Lundy [1986], realiza una sola iteración para cada temperatura, es decir, $L(T)=1\ \forall T$. Define un límite superior $U=\max\{C(S_j)-C(S)\}\quad \forall S_j, S\in\Omega$, y para cada iteración considerar un cambio de temperatura $a(T)=\frac{T}{1+bT}$, para un $b<U^{-1}$, de este modo se evitan grandes desviaciones respecto al equilibrio en cada paso. Se realiza la elección de T_0 de tal forma que en las primeras iteraciones se aceptan la mayoría de los movimientos, tomando un $T_0>U$, con lo cual se consigue que el factor de Boltzmann sea próximo a uno.

Otro investigador Connolly [1990] para un problema de asignación considera $aT \approx T$, es evidente que la pregunta es cuál es la T más adecuada para éste problema concreto.

La práctica más común y la recomendada como primera aproximación a un nuevo problema [Dowsland, 1993] consiste en considerar una velocidad geométrica de decrecimiento, es decir, $a(T) = kT, 0 < k < 1.$ Se ha visto que los valores más convenientes para el parámetro k son los correspondientes al intervalo $[0.8,0.99]$, el cuál es un conjunto de valores que nos aseguran un enfriamiento lento, [Kuik, Salomon, 1990]. Con el propósito de que se haga una exploración más exhaustiva cuando la temperatura es baja, puede considerarse un $L(T)$ creciente según decrece T.

Es claro que si esto no da buenos resultados debe seleccionarse un programa de enfriamiento específico para cada problema particular.

Otros autores sugieren el uso de un recalentamiento cuando la búsqueda según el enfriamiento sigue sin dar frutos [Dowsland, 1993]

3.6.4 Temperatura Final, T_F

En teoría la temperatura que corresponde al sistema frío debe de ser 0. Sin embargo, bastante antes de llegar a este valor es prácticamente improbable que se acepte un movimiento hacia una solución peor, entonces se puede finalizar con valores $T_f > 0$, sin pérdida en la calidad de la solución.

Según el criterio de Lundy-Mees, el cual consiste en finalizar en cuanto la temperatura sobrepase un umbral mínimo. Estos investigadores han demostrado que si se considera que es q de que finalmente se obtenga una solución para la cual su costo menos el de la óptima global sea cuando mucho e, se cumple que:

$$\frac{q}{1-q} > (n-1)e^{-\frac{e}{T}}$$

siendo $n = card(\Omega)$ el número de elementos del espacio de soluciones. Despejando la temperatura, obtenemos que el valor límite T_f es:

$$T_f < -\frac{e}{\ln(n-1) - \ln(q) + \ln(1-q)} \approx \frac{e}{\ln(n)}$$

CAPÍTULO IV

4. FORMULACION DEL MODELO MATEMATICO

Este capítulo incluye La formulación y planteamiento del modelo matemático para obtener la función de costo y solución inicial, para lo cual debemos tener muy clara la teoría de Scheduling.

4.1. SCHEDULING (PLANEACION - ASIGNACION)

4.1.1 Introducción

Scheduling es un problema de complejidad NP-completo o NP-Duro y es tradicionalmente resuelto mediante la combinación de técnicas heurísticas locales y globales, que intentan reducir el espacio de búsqueda.

4.1.2 Resumen

Scheduling es un problema de optimización que se presenta en sistemas de planificación en la producción industrial, y planificación de transporte, etc. Es un proceso de toma de decisiones que tiene como meta la optimización de uno o más objetivos. Consiste en asignar recursos limitados a tareas donde hay restricciones de tiempo. El resultado será la obtención de una solución que minimice el tiempo necesario para completar la ejecución de todas las tareas del problema.

4.2 NUESTRO PROBLEMA

En la Industria Cartonera el problema que buscamos resolver es la Planificación de la Producción que consiste en la distribución y asignación de unidades de recursos o maquinaria (máquinas corrugadoras) a una combinación de pedidos (trimaje). Si se analiza los componentes de éste problema, se tiene que en primer lugar las unidades para producir son limitadas, la forma en la que se podría efectuar la planificación de la producción de los pedidos en el menor tiempo posible, seria asignar una unidad a cada uno de los pedidos. Pero, lo más realista sería pensar que no se tienen

tantas unidades como pedidos, y el problema consiste en determinar la ruta (planificación) que debe seguirse para producir en el menor tiempo (con el menor costo) todos los pedidos.

Pero aquí hay dos problemas en los que se debe pensar: Primero, el tiempo mínimo (si es que se puede determinar, puede ser demasiado largo). Segundo, las unidades tienen una cierta capacidad de almacenamiento o producción.

Este problema contiene dentro de sí muchos más: Determinar cuál es el tamaño óptimo de la maquinaria, determinar cuáles son los pedidos que deben ser asignados a cada unidad para ser producidos y determinar cuál es la ruta (planificación) que debe seguir cada una para terminar la producción en el menor tiempo posible.

4.3 MODELO MATEMATICO

4.3.1 Función de Costo

Para conseguir una función de tiempo, debemos determinar el tiempo que se demora cada unidad en producir la combinación de ordenes asignadas,. Para lo cual debemos obtener:

a) La cantidad dada en metros que se va ha necesitar de la plancha (caja abierta) para producir cada orden, calculando:

Q_{ij}: Cantidad en mts. de p_{ij} ; ***p_{ij}:*** pedido *i* en la corrugadora *j*

Q_{ij}: $q_{ij} * L_{ij}$; ***i = (****1,... m pedidos****), j= (****1,2 corrugadoras****)***

q_{ij}: cantidad en unidades de ítem o productos de p_{ij}

L_{ij}: largo de plancha en mts. de p_{ij} , y se obtiene con el siguiente calculo: $2* l_{ij} + 2* a_{ij} + tol_{ij}$

l_{ij}: largo del item o producto de p_{ij}

a_{ij}: ancho del item o producto de p_{ij}

$to.l_{ij}$: tolerancia del largo de la plancha, que depende del test de p_{ij}.

b) La relación de la cantidad de producir la orden con respecto a la capacidad de producción de la unidad , determina el tiempo de producir dicha orden.

Tp $_{ij}$: Tiempo de Producción de p_{ij}

Tp $_{ij}$: Q_{ij} / W_j ; ***i = (****1,... m pedidos****), j= (****1,2 corrugadoras****)***

W_j: capacidad de producción de la corrugadora *j* dada en metros por hora.

c) El tiempo que demora el cambio en la programación de una orden a la siguiente, es decir cambio de medidas (m^2) y características (test y flauta) en la máquina corrugadora.

Tcp $_{ij}$: Tiempo de Cambio en Programación de p_{ij}

$$Tcp_{ij}: \begin{cases} 15 \text{ min. ; si trimaje}_{ij} \neq \text{ trimaje}_{i+1j} \\ 0 \text{ min. ; si trimaje}_{ij} = \text{trimaje}_{i+1j} \end{cases}$$

Trimaje $_{ij}$: *flauta, test, m^2 de p_{ij}*

d) La suma del tiempo de producción con el tiempo de cambio en programación, determina el tiempo total que demora la producción de dicha orden.

T_{ij} **:** Función de Tiempo de p_{ij}

$T_{ij} : Tp_{ij} + Tcp_{ij}$; ***i = (****1,... m pedidos****)***, ***j= (****1,2 corrugadoras****)***

e) La suma de la formulación anterior, determina el tiempo total de producir todas las ordenes asignadas

C: Función de Tiempo para los pedidos asignados

$$C: \sum_{J=0}^{2} \sum_{i=0}^{m} T_{ij;} \quad i = (1,... \text{ m pedidos}), j= (1,2 \text{ corrugadoras})$$

f) Para obtener un tiempo total óptimo de producir todas las ordenes debemos minimizar la función de tiempo.

Min. *C*

4.3.2 Solución Inicial

Para obtener una solución inicial, se debe considerar:

a) Numero de Máquinas disponibles para la producción

b) Capacidad de cada máquina

c) Características de cada máquina

d) Determinar el conjunto de pedidos que pueden asignarse a cada máquina.

Para determinar cuales son los pedidos que deben ser asignados a cada unidad, debemos considerar ciertas características de dichas unidades, como se mencionó en el capítulo 1, se tienen diferentes tipos de corrugados, los de Pared Simple (*flauta B y C*) y los de Pared Doble (*flauta BC*), y se tienen dos máquinas Corrugadoras, la ***S & S*** y la ***Langston***, la primera es usada solo para corrugados de flauta ***C***, y la segunda para los corrugados de flauta ***B, C*** y ***BC***.

De manera que lo primero que se debe hacer es particionar el conjunto

$\boldsymbol{P = \{}\ p_1, p_2, p_3, \ldots, p_n\ \boldsymbol{\}}$ de pedidos en 2 subconjuntos,

$\boldsymbol{P = \{}\ LSS, L\ \boldsymbol{\}}$;

$\boldsymbol{LSS = \{}\ p_1, p_2, p_3, \ldots, p_m\ \boldsymbol{\}}$ y $\boldsymbol{L = \{}\ p_1, p_2, p_3, \ldots, p_r\ \boldsymbol{\}}$; $n = m+r$

$Si\ p_i \in LSS \Rightarrow (flauta.p_i = B \vee flauta.p_i = C \vee flauta\ p_i = BC)$

$Si\ p_i \in L \Rightarrow flauta.p_i = C$

Pero no es suficiente seleccionar que pedidos deben ser asignadas a cada unidad, lo mas importante es proponer en que orden pueden ser producidos, y esto se puede solucionar sabiendo cuales son los que tienen prioridad, es decir los mas urgentes de cumplir, lo que se indica en la fecha de despacho del pedido. Por lo que ordenaremos cada partición o subconjunto ***LSS*** y ***L***, es decir se particionará en grupos por fecha de despacho y estos grupos deben ordenarse de menor a mayor.

$\boldsymbol{LSS = \{}\ LSS_1, LSS_2, \ldots, LSS_p\ \boldsymbol{\}}$ y $\boldsymbol{L = \{}\ L_1, L_2, \ldots, L_q\ \boldsymbol{\}}$;

$\boldsymbol{LSS_k = \{}\ p_{k1}, p_{k2, \ldots,} p_{kn}\ \boldsymbol{\}}$

$Si\ p_i\ , p_j \in LSS_k \Rightarrow FechaDespacho.\ p_i = FechaDespacho.\ p_j$;

$k = (1,2, \ldots,p)$

$\boldsymbol{L_k = \{}\ p_{k1}, p_{k2, \ldots,} p_{km}\ \boldsymbol{\}}$

Si p_i , $p_j \in L_k \Rightarrow$ *FechaDespacho.* p_i = *FechaDespacho.* p_j *;* k = *(1,2, ...,q)*

Y para minimizar el tiempo, se debe minimizar el número de paradas entre ordenes, por cambio de medidas (mts.^{2} = largo de la plancha * ancho de la plancha) y características del pedidos (*test y flauta*), que como se menciono en el capitulo 1, es una de las principales causas por las que se para la máquina, lo que ocasiona una demora de 15 a 20 min. mas. Por lo que después de particionar ordenadamente por la fecha de despacho serán agrupados o combinados por test, luego por flauta, y por último por medidas, (a esta clase de combinación de pedidos se le denomina trimaje en el ámbito de la Producción Cartonera). Es decir que cada partición o subconjunto de ***LSS*** y *de* ***L*** será a su vez particionado en los grupos de combinaciones de pedidos:

$\boldsymbol{LSS_k = \{} O_{k1}, O_{k2}, \ldots, O_{kn} \boldsymbol{\}}$;

$\boldsymbol{O_{k1} = \{} p_1, p_{2, \ldots,} p_a \boldsymbol{\}}\ \boldsymbol{O_{k2} = \{} p_1, p_{2, \ldots,} p_b \boldsymbol{\}}$,..., $\boldsymbol{O_{kn} = \{} p_1, p_{2, \ldots,} p_c \boldsymbol{\};}$

$p = a+b+\ldots+c.$

$Si\ p_i\ ,\ p_j \in O_k \Rightarrow (test.p_i = test.p_j \wedge flauta.p_i = flauta.p_j \wedge m^2.p_i = m^2.p_j\ )$;

$k = (1,2, ...,p)$

$\boldsymbol{L_i = \{\ O_{i1}, O_{i2}, ..., O_{im}\ \}}$;

$\boldsymbol{O_{k1} = \{\ p_1, p_{2, ..., p_d}\ \}\ O_{k2} = \{\ p_1, p_{2, ..., p_e}\ \}}$,..., $\boldsymbol{O_{kn} = \{\ p_1, p_{2, ..., p_f}\ \}}$;

$q = d+e+...+f.$

$Si\ p_i\ ,\ p_j \in O_k \Rightarrow (Test.p_i = Test.p_j \wedge Flauta.p_i = Flauta.p_j \wedge m^2.p_i = m^2.p_j\ )$;

$k = (1,2, ...,q)$

Bien, una vez obtenido las particiones o grupos de ordenes, sabemos bajo que criterios podemos asignar los pedidos a cada unidad y en que orden, ya que hemos considerado ciertas restricciones: orden por fecha de despacho, agrupación por características de flauta y test, y medidas de mts^2 por lo que a este resultado lo denominamos ***Conjunto Solución Inicial***

4.3.3 Costo de la Solución

Para determinar el Costo de la Solución, procedemos a obtener los siguientes parámetros:

feci$_{ij}$: fecha inicio, fecha que empieza la producción el p_{ij}

fecf$_{ij}$: fecha fin, fecha que termina la producción el p_{ij} , está dada por: $fi_{ij} + Tp_{ij}$

t$_{ij}$: test de p_{ij}

f$_{ij}$: flauta de p_{ij}

k$_j$: numero de pedidos asignados a la corrugadora j

m$^2_{ij}$: metros cuadrados de la plancha de p_{ij} , está dado por:

$L_{ij} * A_{ij}$

A$_{ij}$: ancho de plancha del pedido i en la corrugadora j , y se obtiene con el siguiente calculo: $a_{ij} + h_{ij} + to.a_{ij}$

a$_{ij}$: ancho del item o producto de p_{ij}

h$_{ij}$: alto del item o producto de p_{ij}

to.a$_{ij}$: tolerancia del ancho de la plancha, que depende del test de $p_{ij.}$

Algoritmo 4.1: Algoritmo de la Función para obtener el Costo de la Solución

---

$\rightarrow$ fecha

j $\leftarrow$ 1 to 2

begin

$\rightarrow k_j$

$fecf_{0j} \leftarrow$ fecha

$C \leftarrow 0$

$f_{0j} \leftarrow f_{1j}$

$t_{0j} \leftarrow t_{1j}$

$m^2_{0j} \leftarrow m^2_{1j}$

j $\leftarrow$ 1 to k_j

begin

$\rightarrow q_{ij}$

$\rightarrow L_{ij}$

$\rightarrow W_j$

Si $f_{i-1j} <> f_{ij}$ or $t_{i-1j} <> t_{ij}$ or $m^2_{i-1j} <> m^2_{ij}$ entonces

begin

Tcp $\leftarrow$ 15

else

$Tcp_{ij} \leftarrow 0$

end

$Tp_{ij} \leftarrow (q_{ij} * L_{ij}) / W_j$

$C \leftarrow C + Tp_{ij} + Tcp_{ij}$

$feci_{ij} \leftarrow fecf_{i-1j} + Tcp_{ij}$

$fecf_{ij} \leftarrow feci_{ij} + Tp_{ij}$

end

end

Mediante la implementación del algoritmo anterior a la partición o grupo de ordenes correspondiente a cada unidad hemos asignado:

a) Fecha en que inicia la producción

b) Tiempo que demora la producción

c) Fecha en que termina la producción

Además hemos obtenido el costo de producir los resultados del algoritmo considerados como una solución

Entonces decimos

$$C : \sum_{J=0}^{2} \sum_{i=0}^{m} T_{ij}$$: costo o tiempo de producir las ordenes asignadas

i = (*1,... m pedidos****), j= (****1,2 corrugadoras****)***

Nuestro trabajo consiste en saber cuando rechazar o aprobar una solución lo que podemos determinar mediante el costo de la solución

4.3.4 Aprobar o Rechazar una Solución

Para rechazar o aprobar un conjunto solución, debemos obtener su costo el que debe ser evaluado por medio de métodos de búsqueda por entornos, que están dentro de las técnicas Meta-Heurísticas, que como hemos dicho anteriormente en el capítulo 2 son Heurísticas de éxito usadas para problemas intratables o NP Completos como es nuestro caso. Se ha escogido entre dichas técnicas la del Recocido Simulado.

Esta evaluación permite rechazar o aprobar el conjunto solución

Algoritmo 4.2: Algoritmo de la Función del Recocido Simulado

$\rightarrow k_j$

$T \leftarrow 0.005 * C_j(Sol_{inicial})$

$N \leftarrow 5*(k_j)$ número de iteraciones estimado

$e \leftarrow 10 \wedge -5$ epsilon o error estimado

$T_f = e /(k_j / 2)$

While $T >= T_f$

begin

$i \leftarrow 1$ to N

begin

$Sol_{cand} \leftarrow$ Permutar (Sol_{act})

$delta \leftarrow C_j(Sol_{cand}) - C_j (Sol_{act})$

$boltzmann \leftarrow Exp(-(delta / T_0))$

Si (Rand < boltzman) Or (delta < 0) entonces

begin

$Sol_{act} \leftarrow Sol_{cand}$

end

$T \leftarrow Rnd * (0.99 - 0.88) + 0.88 * T$

end

end

Algoritmo 4.3: Algoritmo de la Función Permutación de n elementos

---

```
→ n
i ← 1  to  n
  begin
        → V (i)
  end
Randomize
x ← Rand * (n - 1) + 1
y ← Rand * (n - 1) + 1
Si  x < y entonces
   Begin
        i ← 1  to  x-1
        begin
              U (i) ← V(i)
        end
        k←0
        i ← x  to  y
        begin
              U (i) ← V(y-k)
              k←k+1
        end
```

```
        i ← y+1  to  n
        begin
                U (i) ← V(i)
        End
Else
        k←0
        i ← 1  to  x
        begin
                U (i) ← V(x-k)
                k←k+1
        end
        k←0
        i ← x+1  to  y-1
        begin
                U (i) ← V(i)
        End

        i ← y  to  n
        begin
                U (i) ← V(n-k)
                k←k+1
        end
```

```
 end
i ← 1  to  n
   begin
        U (i) → ‘ mostrar la Permutacion
   End
```

Algoritmo 4.4: Algoritmo de la Función Factorial de n

```
→ n
Si n > 1 entonces
 begin
    fact ← 1
    i ← 0 to n – 1
      begin
         fact ← fact * (n - i)
      end
      Factorial ← fact
 else
     Si n = 0 Or n = 1 entonces
       Factorial ← 1
 end
```

CAPÍTULO V

5. DISEÑO E IMPLEMENTACIÓN

Este capítulo incluye los detalles del diseño utilizado para llevar a cabo este proyecto y las particularidades al momento de la implementación.

5.1. ARQUITECTURA DEL SISTEMA

Este sistema maneja una arquitectura de tres capas con tres niveles lógicos:

1. La capa de ***presentación*** ,
2. La capa de ***negocio***
3. La capa de ***datos*****.**

Capa de Presentación simplemente contiene los componentes de interfaz de usuario, es decir los formularios del sistema

Capa de Negocio es el "puente" entre un usuario y los servicios de datos, ya que responden a peticiones del usuario (u otros servicios de negocios) para ejecutar una tarea de este tipo. Cumplen con esto aplicando procedimientos formales y reglas de negocio a los datos relevantes. Cuando los datos necesarios residen en un servidor de base de datos, garantizan los servicios de datos indispensables para cumplir con la tarea de negocios o aplicar su regla. Esto aísla al usuario de la interacción directa con la base de datos.

El nivel de servicios de negocios es responsable de:

- Recibir la entrada del nivel de presentación (Formularios).
- Interactuar con los servicios de datos para ejecutar las operaciones de negocios para los que la aplicación fue diseñada a automatizar en este caso, el procesamiento de consultas de estudiantes, materias, etc).
- Enviar el resultado procesado al nivel de presentación.

.

Capa de Datos se suele incluir tan sólo la base de datos, aunque a veces también un conjunto de componentes que quedan entre los del

negocio y los de la base de datos. Haya o no componentes en esta capa, la base de datos deberá tener implementada alguna lógica del negocio en procedimientos almacenados (Store Procedures) y disparadores (Triggers).

5.1.1 Diseño de la base de datos

Tan importante como obtener los datos y disponer de un medio para guardarlos, es que dicho medio nos permita extraer, filtrar, organizar y seleccionar conforme a nuestras necesidades. La solución a este problema la proporcionan las aplicaciones denominadas manejador de base de datos

Un manejador de base de datos se puede describir, como un contenedor de información, que organiza la misma en base a una serie de reglas. Dicha información puede ser manipulada mediante un conjunto de instrucciones que permitirán al usuario consultar y modificar los datos contenidos.

La administración de un manejador de datos, por otra parte, se puede definir como el conjunto de labores cuyo objetivo es conseguir un rendimiento óptimo del sistema de base de datos, de forma que la información esté en todo momento disponible y

con el menor tiempo de espera posible para el usuario que la solicita.

Bases de datos en SQL Server 2000.

Para el presente trabajo se utilizara el DBMS “Microsoft SQL Server 2000”, el cual es un manejador de base de datos relacionales compuesto por un conjunto de elementos, que se integran con los sistemas operativos NT de Windows (Windows NT 4.0 Workstation y Server, Windows 2000 Profesional, Server y Advanced Server y Windows XP Professional), y el resto de la familia de productos empresariales de Microsoft, BackOffice, para proporcionar un entorno avanzado de proceso de datos, dentro de una arquitectura cliente-servidor, (en próximos apartados trataremos el concepto cliente-servidor).

El rendimiento conseguido por SQL Server 2000 al ejecutarse en sistemas Windows NT, ediciones Server, es excelente, debido a la mencionada orientación cliente-servidor, y a los componentes específicamente desarrollados en estos sistemas operativos para la ejecución de SQL Server.

Algunas de las ventajas del trabajo conjunto entre SQL Server y los sistemas NT de Windows se enumeran a continuación:

- SQL Server aprovecha las características multiproceso de Windows NT, utilizando todos los procesadores instalados para optimizar el manejo de datos.

- El sistema de seguridad de SQL Server 2000 está conformado por un usuario creado en la base de datos. De manera que el usuario inicia su sesión en el sistema y automáticamente se conecta con SQL Server, ye establece una relación de confianza en la que SQL Server verifica si el usuario y la clave de acceso es correcto.

- Para las labores de supervisión del funcionamiento, SQL Server aprovecha el Visor de sucesos del sistema operativo para insertar sus propios mensajes, unificando en un sólo lugar el sistema de avisos. De igual modo, utiliza el Monitor del sistema de Windows NT para aspectos relacionados con el rendimiento de las base de datos.

- Sobre la disponibilidad inmediata de los datos en casos de fallo del servidor, SQL Server aprovecha las capacidades de clustering de que dispone Windows NT Server, de forma que si en un sistema se han instalado dos servidores en clúster, SQL Server realizará el cambio al servidor de respaldo en el caso de que se produzca una caída del principal.

SQL Server puede ejecutarse en una amplia variedad de sistemas operativos. Dependiendo del sistema, podrá actuar como cliente o servidor. Los sistemas a los que se proporciona capacidad de servidor son:

- Windows NT en cualquiera de sus ediciones (Server,y Workstation) y Windows 9x. En cuanto a los sistemas para los que SQL Server dispone de elementos de cliente están los antes mencionados en el aspecto de servidor más Windows 3.x, MS-DOS, Macintosh y UNIX.

- Finalmente, fuera del ámbito de sistemas, también puede ejecutarse como cliente dentro de los navegadores de Internet.
- En cuanto a la escalabilidad del motor de datos, puede manejar desde pequeñas base de datos en modo local a grandes base de datos con conexiones de miles de usuarios y más de un terabyte de capacidad de almacenamiento.

Una base de datos está formada por una serie de elementos, también denominados objetos de la base de datos, que permiten organizar la información, relacionarla con otros objetos de la base de datos, mantener su integridad, etc.

Entre los objetos más importantes podemos destacar los siguientes:

- Tabla.
- Índice.
- Vista.
- Procedimiento almacenado.
- Función o rol.
- Desencadenador.

El sistema desarrollado para esta tesis necesita de una base de datos con una estructura en la que se han implementado Procedimientos almacenados para todos lo procesos que impliquen llamadas a base de datos, es decir que las sentencias de consultas, inserciones, modificaciones y eliminación de registros no se encuentran en el código de VB, sino en la base de datos aumentando notablemente el rendimiento y la optimización de procesos ya que todo debe estar centralizado en el DBMS.

La estructura de la base de datos mostrada en la figura 5.1, incluye lo siguiente:

Tablas primarias:

Ordenes.- Tabla donde se almacenan las ordenes que el usuario haya ingresado

Tablas de relación:

***Tests*.-** Tabla donde se almacenan los todos los tests que la Industria utiliza

***Flautas*.-** Tabla donde se almacenan las clase de flautas que la Industria utiliza

***TetsFlautas*.-** Tabla donde se encuentra todas las combinaciones posibles de los Tests y Flautas que en la producción Industrial se utiliza

***Corrugadoras*.-** Tabla donde se almacenan los nombres de las Corrugadotas que la Industria posee

***CorrugadorasFlautas*.-** Tabla donde se encuadra la asignación de las Flautas existentes a las Corrugadotas de la Industria

***Papeles*.-** Tabla donde se almacenan los tipos de papeles que en la producción industrial se utilizan

***Clientes*.-** Tabla donde se almacenan los clientes que la Industria tiene registrados

***Encargados*.-** Tabla donde se almacenan los encargados que la Industria posee

***Mercados*.-** Tabla donde se almacenan todos los tipos de mercados a los que la Industria se dirige

***Productos*.-** Tabla donde se almacenan todos los tipos de productos que la Industria maneja en su producción

Usuarios.- Tabla donde se almacenan los usuarios registrados para el uso del sistema, ya sea de tipo Administrador o No Administrador.

***Programaciones*.-** Tabla donde se guardan los encabezados de todas la programaciones, resultado del sistema y seleccionadas por el usuario.

***DetallesProgramaciones*.-** Tabla donde se guardan los detalles de todas la programaciones, resultado del sistema y seleccionadas por el usuario.

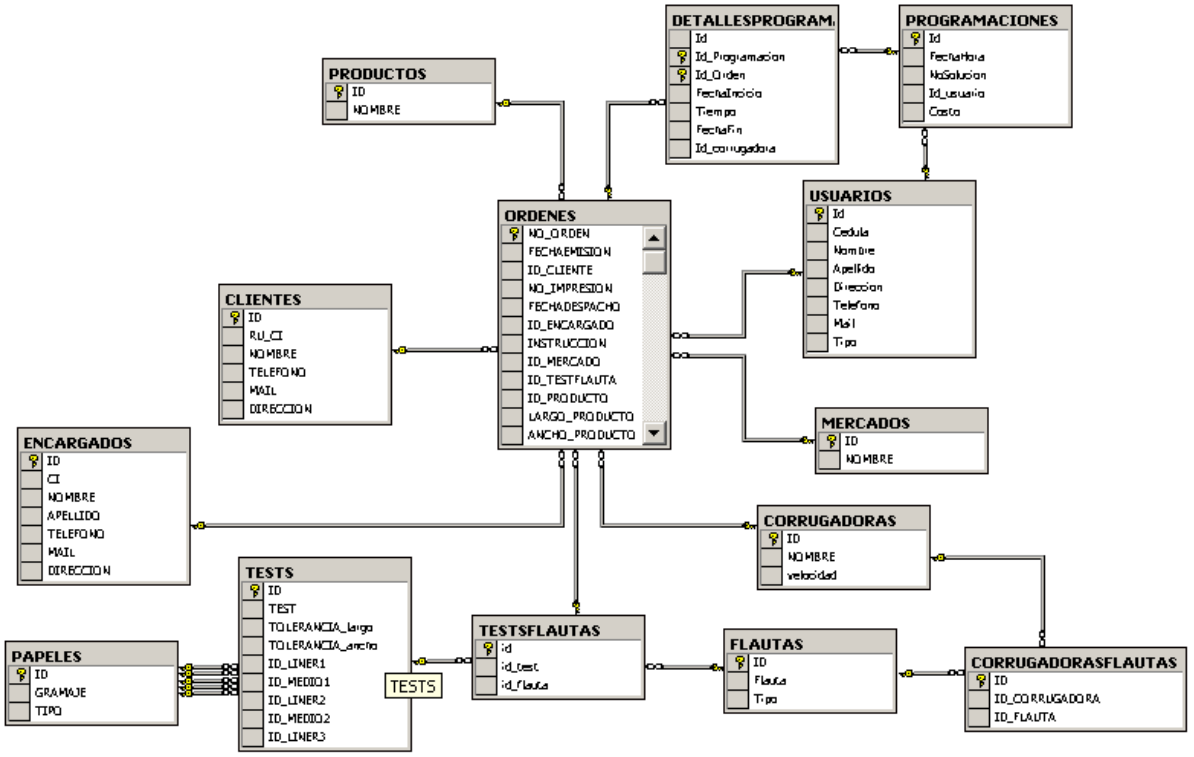

Figura 5.1. *Estructura de la base de datos.*

5.1.2 Diseño de la interfaz del sistema

A continuación presentamos el diseño de la interfaz del sistema propuesto para interactuar con el usuario. Se define la forma de realizar las consultas y la información mostrada, resultados del sistema, así como también la distinción de opciones de acuerdo al tipo de usuario.

Pantallas del Sistema:

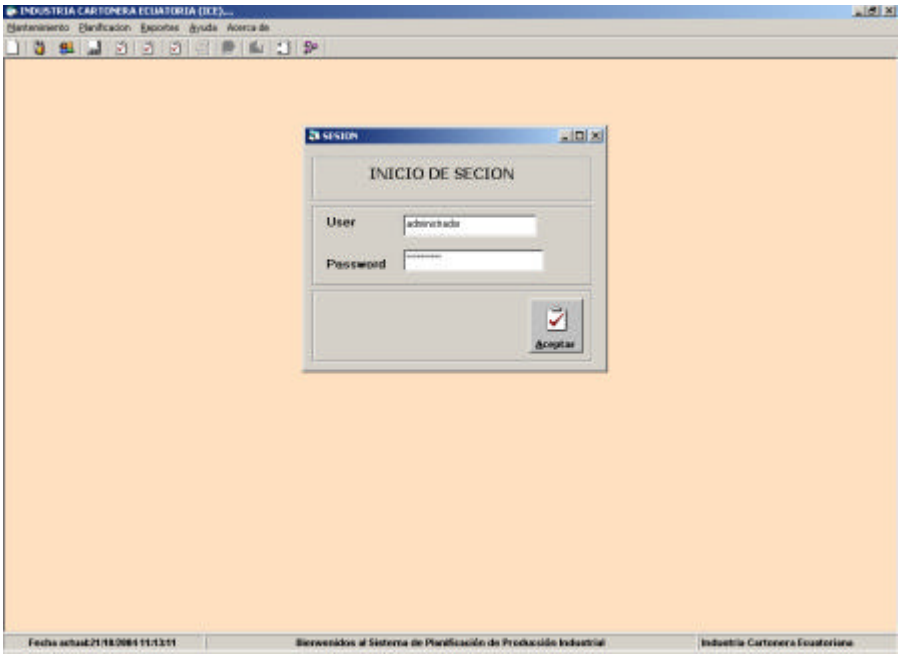

Figura 5.2. *Pantalla de Ingreso.*

En cuanto a la información de utilidad para el usuario Administrador, se encuentra en los formularios del menú de mantenimientos ya sea para:

- Cliente
- Encargado
- Producto
- Mercado

Todos los mantenimientos se efectúan de la misma manera así que mostraremos uno de ellos, el de Cliente

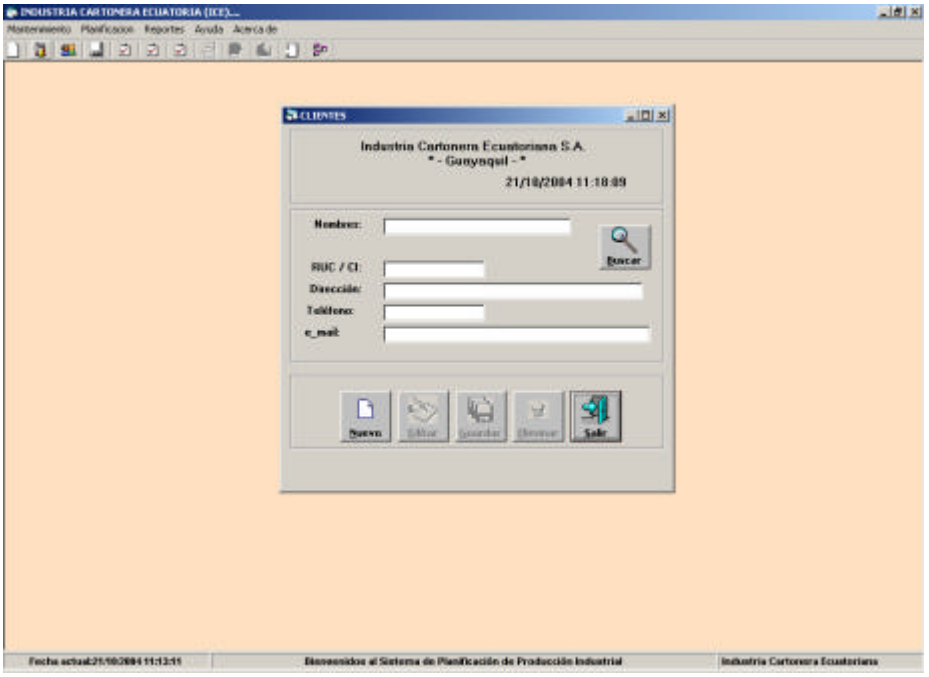

Figura 5.3. *Pantalla de Mantenimiento de Cliente*

Al hacer click en botón Nuevo le permite ingresar un cliente y al dar clic en el botón Guardar insertara en la base de datos el nuevo cliente.

El botón Buscar podrá visualizar una lista de clientes que constan en la base de datos le permitirá seleccionar uno y le mostrara los datos de dicho cliente en el formulario ya sea tan solo para consultar la información, o actualizar algún dato con el botón Actualizar o eliminar al cliente con el botón Eliminar

En este menú de Mantenimientos, si el usuario es No Administrador tendrá ciertas restricciones que no le permiten ingresar nuevos registros actualizar o eliminar registros existentes, tan solo consultar . En el menú de Reportes este usuario tendrá acceso a cada una sus opciones ya que solo son reportes.

La Programación de Ordenes existentes solamente le compete al usuario Administrador que son los encargados y responsables de planificar la programación de las ordenes dentro de una Producción. Esta sección se encuentra en la opción Programar dentro del Menú Planificación del Sistema.

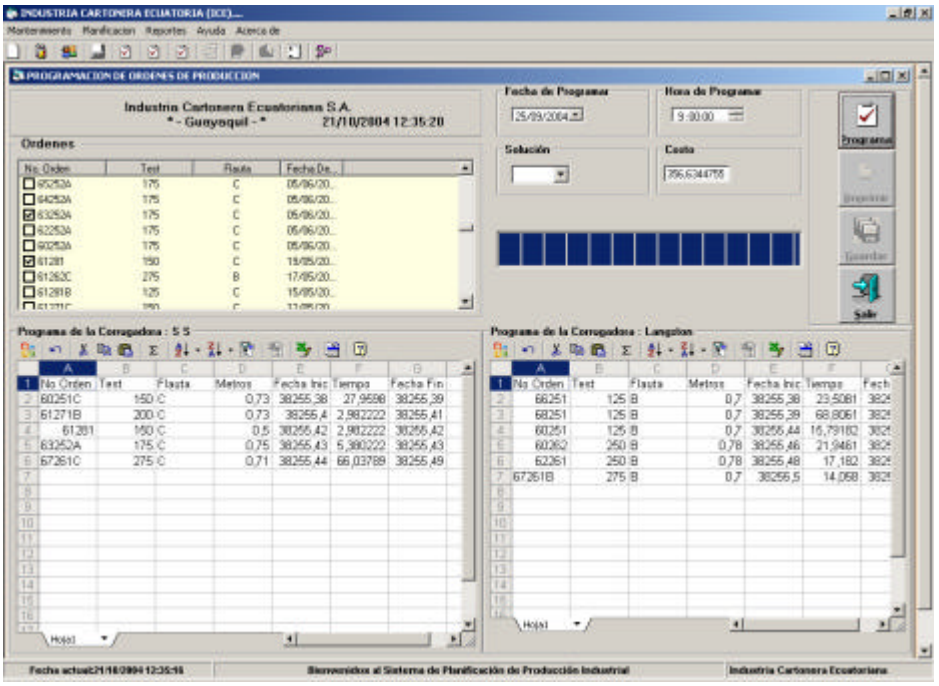

Figura 5.4. *Pantalla de Programación de Ordenes*

El usuario podrá visualizar en un list view todas las órdenes que dentro de la Base tienen un estado de Abiertas, es decir que aun no han sido programadas, de manera que el usuario tiene la opción de escoger las ordenes que desea programar, una vez seleccionadas, debe ingresar la fecha y hora en que desea que inicie la programación, finalmente da click en el botón Programar y los resultados de cada máquina denominados Solución Optima serán visualizados en los Web Components correspondientes a cada máquina; y además podrá observar el costo o tiempo que demorara en producir las ordenes bajo esa programación

Pero además el usuario tiene la opción escoger las otras soluciones que fueron aceptadas antes de la optima, según el algoritmo. Así el podrá determinar que solución le conviene de acuerdo al costo. Y dando clic en el botón Imprimir podrá visualizar las soluciones impresas y guardarlas en la Base de Datos con el botón Guardar. De manera que si posteriormente desea revisar las programaciones anteriores puede hacerlo en la opción Programas del Menú Reportes del Sistema.

5.2. DESARROLLO DEL SISTEMA

En esta sección se va a explicar los detalles concernientes a la implementación del sistema, la cual incluye lo referente a los lenguajes de programación utilizados tanto en VB como en el manejador de base de datos (Procedimientos almacenados).

5.2.1 Programación del lado de la aplicación

Cabe resaltar que éste sistema esta desarrollado en Visual Basic 6.0

Conexión a la base de datos

Puede utilizar la opción Orígenes de Datos ODBC de Herramientas Administrativas del Panel de control (Administrar Orígenes de Datos ODBC) para agregar, configurar o quitar DSN de usuario, DSN de sistema, DSN de archivo; para el desarrollo de este sistema hemos agregado un DSN de sistema con el nombre de DSN_ICE que esta configurado con una conexión a la base de datos del sistema que tiene el nombre de ICE, con user = sa y sin password., la misma que debe ser restaurada con el backup que se adjunta en el CD del sistema de la tesis que tiene el nombre de backup_ICE.

Para la conexión a la Base de Datos se crea un objeto recordset, este no es mas que una forma de obtener registros de la fuente de datos, para poder identificar el origen de la información

Para crearlo se debe agregar la referencia Microsft Data Formatting Object en VB dentro del menú Project en References

En el Anexo A se encuentra la programación modular que contiene los objetos recordset y de conexión, las variables publicas y todos los procedimientos y funciones, que se implementaron para el desarrollo de este trabajo.

Para el desarrollo de esta tesis hemos trabajado también en memoria para la implementación del Algoritmo Recocido Simulado, es decir que tan solo la Solución Inicial que el algoritmo utiliza para empezar la simulación fue implementada en procedimiento almacenado del motor de la BD. Pero en VB se implemento las Listas Doblemente Enlazadas dentro de una programación modular de Clases, la misma que se encuentra en el Anexo B.

5.2.2 Programación del lado de la base de datos

Este tipo de programación se la realiza en el motor de base de datos, en este caso SQL Server 2000, mediante la utilización de procedimientos almacenados.

Procedimientos almacenados

Los procedimientos almacenados son conjuntos de sentencias en lenguaje Transact SQL que pueden almacenarse en el propio servidor. Los procedimientos almacenados de SQL Server, son mas potentes, porque permiten almacenar funciones y procedimientos compuestos por varias instrucciones, introducir saltos, bucles, etc.

Ejecutar un procedimiento dentro del Query Analayzer

Par ejecutar el procedimiento dentro del Analizador de Consular Query Analayzer se utiliza el comando EXECUTE o su abreviación EXEC.

```
EXEC asigna_programacion
```

Ejecutar un procedimiento dentro de VB

Para ejecutar un procedimiento almacenado desde VB, a través de un objeto conn (variable de conexion), se utilizan las siguientes sentencias:

```
conn.Execute ("asigna_programacion")
```

Llamar a un procedimiento dentro de VB

Para llamar a un procedimiento almacenado desde VB, a través de un objeto Recorsedt, se utilizan las siguientes sentencias:

```
Set rst = conn.Execute("programacion 1")
```

En el Anexo C, se muestra la creación de uno de los procedimientos almacenados que se implementaron para el desarrollo de este proyecto de tesis

5.3. PRUEBAS DEL SISTEMA

En esta sección se muestra la fase de prueba por la que ha tenido que pasar el proyecto de la tesis, ya que es debido esta fase el sistema ha ido mejorando tanto en su interfaz (para que esta sea amigable y fácil para el usuario), como en las soluciones (para que estas sean soluciones óptimas en las que el usuario pueda confiar).

Como se mencionó en el capítulo 3, para lograr que las soluciones sean óptimas es muy importante experimentar con los parámetros que el algoritmo Recocido Simulado requiere, es decir ajustar o calibrar los parámetros.

5.3.1 Calibración de los Parámetros

Los parámetros a calibrar del Algoritmo Recocido Simulado, la explicación y demostración de sus fórmulas se mencionaron en el capítulo 3. A continuación se nombra los parámetros, utilizados para la implementación del algoritmo:

- Temperatura Inicial $T_0 = 0.005 * C(S_{inicial})$
- Velocidad de Enfriamiento (Número de iteraciones) $L(T)$, éste parámetro es el más importante, ya que es el criterio de parada del algoritmo. Para obtener una solución óptima, $L(T)$ debe ser al menos igual a $card(\Omega)^2$, es decir que el algoritmo haría una búsqueda exhaustiva; pero el tiempo de ejecución sería demasiado largo, y el objetivo de este trabajo es implementar un algoritmo que busque una solución rápida y óptima, de manera que hemos experimentado con $L(T)$ hasta ajustarlo a nuestra conveniencia.

$$L(T) = 5 * n \quad ; n = \text{Número de Pedidos}$$

$L(T)$	$C(S_{óptima})$
$2*n$	321.7356395
$3*n$	321.7356395

$4*n$	321.73563955
$5*n$	321.7356395

- Decrecimiento de la Temperatura $a(T) = kT, 0 < k < 1$; para nuestro caso hemos escogido un $k \in [0.8,8.99]$ aleatorio para cada iteración
- Temperatura Final $T_f = \frac{e}{\ln(card(\Omega))}$; pero así mismo como se menciono en el parámetro $L(T)$ estaríamos haciendo una búsqueda exhaustiva; de manera que hemos ajustado este parámetro a la siguiente aproximación:

$$T_f = \approx \frac{e}{\ln(n!)} \approx \frac{e}{n/2}$$

CONCLUSIONES Y RECOMENDACIONES

- Como resultado del desarrollo de este proyecto de tesis tenemos un sistema de información que permite planificar una producción de una Industria, cumpliendo con ciertas restricciones de tiempo, requerimientos y especificaciones de la Industria. Es decir que este sistema permite la asignación óptima de los pedidos a las unidades de trabajo en la Producción Industrial.

- La Teoría de Optimización Combinatoria mediante la minimización de la función de costo que para nuestro caso es una función de tiempo; permite encontrar una solución factible para la asignación de pedidos en la Producción Industrial. La solución factible será una solución Inicial para la Implementación del Algorítmo Meta-Heurístico Recocido Simulado.

- La Algorítmo Recocido Simulado, necesita como parámetros: una solución Inicial, el numero de iteraciones, una solución final aproximada, un error estimado; mediante la calibración exacta de estos parámetros el algoritmo puede buscar una mejor solución hasta encontrar una solución óptima.

- El desarrollo del sistema de esta tesis requiere del almacenamiento de la información en una base de datos de SQL Server 2000 y la implementación de procedimientos almacenados en el motor de la base de datos, lo que permite interactuar de una manera fácil y rápida en los procesos de Ingreso, Eliminación, Actualización y Consultas.

- Se recomienda que la implementación del algoritmo Recocido Simulado sea desarrollada en memoria, mediante la utilización de Clases (Listas Doblemente Enlazadas); debido a que por el número de iteraciones al que está sujeto el algoritmo, el desempeño de la implementación en el motor de la base de datos es mínimo y totalmente contrario al desempeño de la implementación en memoria.

- La implementación de la interfaz del sistema fue desarrollada bajo la aplicación de Visual Basic 6.0; pero pudo desarrollarse en cualquier aplicación.

- Este proyecto podría constituirse en una herramienta de ayuda para los encargados de programar y planificar la Producción en la Industria.

Anexos

A. Programación Modular

Variables conection y recordset

```
Public conn As New ADODB.Connection

Public rst As New ADODB.Recordset
```

Procedimiento para Abrir la Conexión de la BD

```
Public Sub Abrir_coneccion()

conn.Open "DSN=DSN_ICE;uid=sa;pwd="

End Sub
```

Procedimiento para Cerrar la Conexión de la BD

```
Public Sub Cerrar_Coneccion() ‘ Permite Cerrar la conexión

conn.Close

Set conn = Nothing

End Sub
```

Procedimiento para Llenar una Lista

```
Public Sub LLenar(ByVal L1 As Lista, ByRef L2 As Lista)
 Dim el As New ElementoLista
 Set el = L1.First
 While (el.Target <> 0)
  L2.Add el.FechaDespacho, el.corrugadora, el.NumeroOrden,
   el.Cantidad, el.Test, _
    el.Flauta, el.Metros, el.LagoP, el.AnchoP, el.OrdenA, _
    el.FechaIActual, el.FechaFActual, el.OrdenP, _
    el.FechaIPropuesta, el.FechaFPropuesta, el.Tiempo
  Set el = el.After
 Wend
End Sub
```

Procedimiento del Recocido Simulado

```
Public Sub simulated_annealing(ByVal Tinicio As Double, ByVal
alfa As Double, ByVal iteracciones As Double, ByVal Tfinal As
Double, ByVal l_actual As Lista, ByVal capacidad As Integer,
fecha As Date, hora As Date)

Dim T As Double
Dim delta As Double
```

```
Dim x As Long
Dim y As Long
Dim max As Long
Dim l_candidata As New Lista
Dim boltzmann As Double
'Dim ePointer As New ElementoLista
max = l_actual.GetLong
LLenar l_actual, l_candidata
 T = Tinicio
frm_programa_orden.barra.Min = 0
 frm_programa_orden.barra.max = iteracciones
 While T >= Tfinal
       For j = 1 To iteracciones
             Randomize
             x = CInt(Rnd * (max - 1) + 1)
             y = CInt(Rnd * (max - 1) + 1)
             l_candidata.Permutar x, y

             delta = costo_solucion(fecha, hora, l_candidata,
              capacidad) - costo_solucion(fecha, hora, l_actual,
              capacidad)
             boltzmann = Exp(-(delta / T))
```

```
            If (Rnd < boltzman) Or (delta < 0) Then
               LLenar l_candidata, l_actual
            End If
         frm_programa_orden.barra.Value = j
        Next j
        Randomize
        T = (Rnd * (0.99 - 0.88) + 0.88) * T
    Wend
End Sub
```

Procedimiento para Generar un Lista Doblemente Enlazada

```
Public Sub genera_lista(n As Integer)
Dim i As Integer
  For i = n To 1 Step -1
        solucion_candidata(i, 1) = i
        solucion_candidata(i, 2) = i * 15
        solucion_candidata(i, 3) = i * 150

     Next i
End Sub
```

Function de Costo Solución

```
Public Function costo_solucion(fecha As Date, hora As Date,
ByVal l As Lista, ByVal capacidad As Integer) As Double
 Dim i As Integer
 Dim max As Long
 Dim suma As Double
 Dim sol As Double
 Dim sol1 As Double
 Dim el1 As New ElementoLista
 Dim mts As Double
 Dim Test As Integer
 Dim Flauta As String
 Dim FechaInicio, FechaFin As Date
 Dim TiempoCProg, TiempoProd, TiempoProd1 As Double
   Set el1 = l.First
   mts = el1.Metros
   Test = el1.Test
   Flauta = el1.Flauta
   FechaDespacho = el1.FechaDespacho
   FechaFin = hora
   suma = 0
     While (el1.Target <> 0)
```

```
 If mts <> el1.Metros Or Test <> el1.Test Or Flauta <>
el1.Flauta Then
    TiempoCProg = 15
Else
    TiempoCProg = 0
End If
If fecha >= FechaDespacho Then
   ' MsgBox "Penalidad por Fecha Despacho"
End If
FechaInicio = DateAdd("n", TiempoCProg, FechaFin)
                TiempoProd1 = CDbl(el1.Cantidad) *
CDbl(el1.LagoP) / 1000
    ' MsgBox "No Orden = " & el1.NumeroOrden & " /
Cantidad = " & CDbl(el1.Cantidad) & " / Largo = " &
CDbl(el1.LagoP) / 1000
TiempoProd = TiempoProd1 / capacidad
If TiempoProd < 1 Then
    seg = TiempoProd * 60
    FechaFin = DateAdd("s", seg, FechaInicio)
```

```
        Else
            Min = CInt(TiempoProd)
            seg = TiempoProd - Min
            FechaFin = DateAdd("n", Min, FechaInicio)
            FechaFin = DateAdd("s", seg, FechaFin)
        End If

        el1.Tiempo = TiempoProd
        el1.FechaIActual = FechaInicio
        el1.FechaFActual = FechaFin

        suma = suma + TiempoProd + TiempoCProg

        mts = el1.Metros
        Test = el1.Test
        Flauta = el1.Flauta

        Set el1 = el1.After
        '  MsgBox "Tiempo Cambio Programa = " & TiempoCProg
& " / Fecha Incicio = " & FechaInicio & " / Tiempo Produccion = "
& TiempoProd & " / Fecha Fin = " & FechaFin
    Wend
```

```
  costo_solucion = suma
End Function
```

Function de Factorial

```
Public Function Factorial(n As Integer) As Double
Dim s As Double
Dim i As Integer
 If n > 1 Then
  s = 1
  For i = 0 To n - 1
     s = s * (n - i)
  Next i
    Factorial = s
  Else
    If n = 0 Or n = 1 Then
      Factorial = 1
    End If
  End If
  End Function
```

B. Programación Modular de Clases

Elementos de la Lista

```
Option Explicit

  Public FechaDespacho As Date

  Public corrugadora As String

  Public NumeroOrden As String

  Public Cantidad As Long

  Public Test As Long

  Public Flauta As String

  Public Metros As Double

  Public LagoP As Double

  Public AnchoP As Double

  Public OrdenA As Long

  Public FechaIActual As Date

  Public FechaFActual As Date

  Public OrdenP As Long

  Public FechaIPropuesta As Date

  Public FechaFPropuesta As Date

  Public Tiempo As Double
```

Procedimientos y Funciones de la Lista

```
Option Explicit

Private Longitud As Long

Public First    As New ElementoLista

Public Last     As New ElementoLista

Private Pointer  As ElementoLista 'TheApp

Private tPointer As New ElementoLista  'TheApp

Public Target As Long

Public Before    As Object

Public After     As Ob ject

Public Target As Long

Public Before    As Object

Public After     As Object
```

C. Creación de un Store Procedure

```
CREATE proc programacion @maq int
as
 select o.fechadespacho,c.nombre, o.no_orden,o.cantidad,
t.test,
f.flauta,o.mts_2,o.largo_plancha,o .ancho_plancha,o.indice,o.ord
en_actual,
       o.fechainicio_actual,
o.fechafin_actual,o.orden_propuesto,o.fechainicio_propuesto,
o.fechafin_propuesto,o.tiempo
 from ordenes o, tests t, flautas f, testsflautas tf,corrugadoras c
 where tf.id_flauta = f.id and tf.id_test = t.id and tf.id =
o.id_testflauta and o.id_corrugadora = @maq and O.estado= 'P'
and o.id_corrugadora=c.id
 order by o.fechadespacho,o.fechainicio_actual
```

GLOSARIO

Store Procedures: Procedimientos Almacenados de SQL Server 2000

Triggers:

Clustering:

List View: Objeto de Visual Basic donde se listan o muestran datos funciona igual que una matriz donde cada fila representa un registro

Web Components

Recordset:

Microsft Data Formatting Object

References

Backup: (copia de respaldo, copia de seguridad) Copia de ficheros o datos de forma que estén disponibles en caso de que un fallo produzca la perdida de los originales. Esta sencilla acción evita numerosos, y a veces irremediables, problemas si se realiza de forma habitual y periódica.

DBMS

ODBC: Objeto de conexión de la base de datos

DSN

Archivo adjunto: Archivo que acompaña un mensaje de e-mail. Es apropiado para el envío de imágenes, sonidos, programas y otros archivos grandes.

ASCII: American Standard Code of Information Interchange: Código normalizado estadounidense para el intercambio de la información. Código que permite definir caracteres alfanuméricos; se lo usa para lograr compatibilidad entre diversos procesadores de texto. Se pronuncia "aski".

Attachement: Archivo adjunto.

Base de datos: Conjunto de datos organizados de modo tal que resulte fácil acceder a ellos, gestionarlos y actualizarlos.

Bit: Abreviatura de binary digit (dígito binario). El bit es la unidad más pequeña de almacenamiento en un sistema binario dentro de una computadora.

Byte: Unidad de información utilizada por las computadoras. Cada byte está compuesto por ocho bits.

CD-ROM: Compact Disk - Read Only Memory. Disco compacto de sólo lectura. Tiene una capacidad de almacenamiento de hasta 650 megabytes, mucho mayor que la de un disquete.

Data: Datos, información.

Database: Base de datos.

Hardware: Componentes físicos de un ordenador o de una red, en contraposición con los programas o elementos lógicos que los hacen funcionar.

MTS: Servidor de transacciones de Microsoft. Es un administrador de transacciones que permite que aplicaciones cliente incluyan varios orígenes de datos en una transacción. Actualmente en Windows 2000 o superior se denomina MS DTC (Coordinador de transacciones distribuidas de Microsoft)

Script: Porción de código que contiene instrucciones en un lenguaje determinado en cual puede ser interpretado para ejecutar una acción o función especifica.

Scripting: Desarrollo de código basado en scripts.

Software: Programas o elementos lógicos que hacen funcionar un ordenador o una red, o que se ejecutan en ellos, en contraposición con los componentes físicos del ordenador o la red.

SQL: Structured Query Language. Lenguaje de programación que se utiliza para recuperar y actualizar la información contenida en una base de datos. Fue desarrollado en los años 70 por IBM. Se ha convertido en un estándar ISO y ANSI.

Windows 2000: Versión del sistema operativo Windows, cuyo lanzamiento ha sido anunciado por Microsoft para el año 1999.

Windows 95: Sistema operativo lanzado por Microsoft en agosto de 1995.

Windows 98: Sistema operativo lanzado por Microsoft en 1998, como sucesor de Windows 95. Una de las más visibles diferencias con el anterior consiste en la integración del sistema operativo con el navegador Internet Explorer. Esta característica dio pie a un juicio por monopolio.

Windows CE: Sistema operativo basado en Windows. Fue diseñado para dispositivos móviles o pequeños. Viene incorporado en varias marcas de handheld.

Windows NT Server: Windows NT diseñado para máquinas que proveen servicios a computadoras conectadas a una LAN.

Windows NT Workstation: Windows NT diseñado especialmente para empresas, se lo considera más seguro y estable que Windows 95 y 98.

Windows NT: Sistema operativo Windows de Microsoft diseñado para usuarios avanzados y empresas. En realidad se trata de dos productos: Windows NT Workstation y Windows NT Server.

World Wide Web: (W3 - Malla Mundial, Telaraña Mundial, WWW) Sistema de información distribuido, basado en hipertexto, creado a principios de los años 90 por Tim Berners-Lee, investigador en el CERN, Suiza. La información puede ser de cualquier formato (texto, gráfico, audio, imagen fija o en movimiento) y es fácilmente accesible a los usuarios mediante los programas navegadores. Es preciso destacar el hecho poco habitual de que tanto Berners-Lee como el CERN renunciaron a la explotación comercial de este extraordinario invento.

Workstation: estación de trabajo. Computadora personal conectada a una LAN. Puede ser usada independientemente de la mainframe, dado que tiene sus propias aplicaciones y su propio disco rígido.

Apéndices

BIBLIOGRAFÍA

- BRIAN SILER – JEFF SPOTTS (1999), “Edición Especial; Visual Basic 6.0”, Prentice Hall

- Tesis de Grado: Estudio y Mejoras del Proceso de Planeamiento de la Producción Industrial
  Autor: Ana Fabiola Terán Alvarado

- WILLIAM R. VAUGHN (1999), “Edición ; Programación de SQL Server 7.0 con Visual Basic 6.0”

- ADENSO DIAZ (1996), “Edición Paraninfo; Optimización Heurística y Redes Neuronales”