Métodos y Algoritmos para resolver problemas de Corte unidimensional en entornos realistas. Aplicación a una empresa del Sector Siderúrgico.

Tesis Doctoral

Departamento de Organización de Empresas

Universidad Politécnica de Valencia

Autor: Carlos P Gracia Calandín
Director: Dr. Carlos Andrés Romano
Director: Dr. Luís I Gracia Calandín

En primer lugar, agradezco a los directores de la tesis Drs. Carlos Andrés y Luis Gracia su valiosa ayuda, estímulo permanente y acertado asesoramiento a lo largo del trabajo.

En segundo lugar, agradezco a las personas vinculadas a la empresa de corte de perfiles mencionada en la tesis, su ayuda a la hora de proporcionarme un contexto real sobre el que aplicar las técnicas y algoritmos desarrollados en el trabajo.

En tercer lugar, vaya también mi agradecimiento a los compañeros del departamento de Organización de Empresas que de algún u otro modo han contribuido al desarrollo de la tesis.

A Inmaculada e Irene
A mis padres

RESUMEN

La presente tesis doctoral aborda el análisis y modelización de los problemas de programación en el corte de perfiles estructurales de acero, así como la propuesta de diferentes metodologías y algoritmos basados en técnicas heurísticas que permiten resolverlos de manera óptima. En concreto se profundiza en los siguientes temas:

- Se estudia la problemática concreta en el corte de vigas estructurales en una empresa de transformados metalúrgicos. Dicho estudio motiva y justifica todo el trabajo posterior, a la vez que proporciona un contexto concreto en el que aplicar de forma práctica los resultados obtenidos con los algoritmos desarrollados.

- Se modeliza matemáticamente el Problema del Corte de vigas a partir de perfiles estructurales.

- Se presenta una metodología que resuelve de manera eficiente, mediante el uso de patrones, el Problema del Corte para satisfacer la demanda de vigas en un periodo concreto. A tal efecto se desarrolla: un primer algoritmo genético que genera patrones de corte idóneos (fase 1); un segundo algoritmo genético que determina las frecuencias de uso de cada patrón para minimizar tanto el desperdicio como la sobreproducción (fase 2); y cuatro algoritmos adicionales que mejoran la solución obtenida en la fase anterior (fase 3).

- A fin de evaluar la metodología propuesta, se desarrolla un generador de problemas que a partir de unos parámetros de instancia obtiene distintos problemas de test.

- Se propone otro algoritmo genético para resolver el Problema multiobjetivo de Secuenciación de Patrones optimizando dos objetivos: minimizar las necesidades de espacio para el apilamiento de pedidos en curso y minimizar la extensión temporal requerida para procesar los pedidos.

- Finalmente se propone una metodología para la resolución del Problema Global de Corte y Secuenciación.

ABSTRACT

This PhD thesis deals with the analysis and modelling of the problems arising in the programming of cutting operations on structural steel profiles, and the proposal of several methodologies and algorithms based on heuristics techniques that solve them optimally. In particular, the thesis focuses on the following issues:

- Considers the specific problem arising in cutting structural steel beams in a national manufacturer. This study motivates and justifies all subsequent work, while providing a specific context in which applying the results obtained with the developed algorithms.

- The Cutting Stock Problem is modelled and identified in the national manufacturer of metal profiles.

- An efficient methodology, based on the use of cutting patterns, is described in order to solve the Cutting Stock Problem so that all costumers´ demands are satisfied for a period of time. So far, we develop the following: a genetic algorithm that generates efficient cutting patterns (phase 1); a second genetic algorithm which solves in a first step the cutting stock problem by determining the frequencies of use of each pattern (phase 2); and four additional algorithms which improve the solution obtained in the previous phase (phase 3).

- In order to evaluate the proposed methodology, a problem generator is developed, so that from certain parameters of an instance the generator gives rise to specific test problems.

- Another genetic algorithm is proposed to solve the multi-objective Sequencing Problem of cutting patterns so that two objectives are accomplished simultaneously: the minimization of space requirements to pile work of in process orders.

- Finally, it is proposed a methodology to solve the Global Cutting and Sequencing Problem

RESUM

La presente tesi doctoral aborda el anàlisi i modelització dels problemes que apareixen en la programació de la producció de perfils estructurals tallats, així como la proposta de diferents metodologies y algoritmes que ens permeten la seua resolució satisfactòriament. En concret s'aprofundix en els temes següents:

- S'estudia la problemàtica concreta en el tall de bigues estructurals en una empresa de transformats metal.lúrgics. Aquest estudi motiva i justifica tot el treball posterior, alhora que proporciona un context concret en el qual aplicar de forma pràctica els resultats obtinguts amb els algoritmes desenvolupats.

- Es modelitza matemàticament el Problema del Tall de Bigues obtingudes a partir de perfils estructurals.

- Es descriu una metodologia eficient que mitjançant L'ús de patrons de tall permet resoldre el Problema del Tall per tal que estiguen satisfetes les demandes de bigues durant un període de temps. Per això es desenvolupa el següent: un algoritme genètic que permet generar patrons de tall eficients (fase 1), un algoritme genètic que resol el problema de la determinació de les freqüències D'ús de cadascú dels patrons (fase 2) i quatre algoritmes que milloren a solució obtinguda en la fase anterior (fase 3).

- A fi D'avaluar la metodologia proposta, es desenvolupa un generador de problemes tal què partint D'uns paràmetres de instància s'obtinguen diferents problemes de test.

- Es proposa un algoritme genètic que resol el Problema multi objectiu de Seqüenciació de Patrons de forma que es permeta simultàniament optimitzar dos objectius: minimització de les necessitats de espai per tal D'apilar les comandes en curs i minimització de la extensió temporal de les comandes en curs.

- Finalment es proposa una metodologia per a la resolució del Problema Global de Tall i Seqüenciació.

ÍNDICE GENERAL

1 Introducción……..15

1.1 Objetivo y estructura de la memoria presentada……..16

2 Problemática en el proceso del corte de perfiles estructurales…….. 19

2.1 El sector del metal……20

2.1.1 Perspectiva general…… 20

2.1.2 El sector en la Comunidad Valenciana……... 21

2.2 Caso real en una empresa ……..…….22

2.2.1 Descripción del proceso de corte de vigas……...23

2.3 Datos y volúmenes del proceso del corte de perfiles (1d msscsp) ……33

2.4 Planteamiento del problema a resolver……..…35

3 Los problemas de corte y empaquetado…….. 37

3.1 Descripción de los problemas de corte y empaquetado……....39

3.1.1 Estructura básica……...39

3.1.2 Clasificación de los problemas de corte y empaquetado…… 40

3.2 Resolución de los problemas de optimización combinatoria…….47

3.2.1 Métodos exactos……...47

3.2.2 Heurísticas……50

3.2.3 Metaheurísticas……51

3.3 Resolución de los problemas de corte y empaquetado…….. 55

3.3.1 Problema de empaquetado con ítems idénticos (IIPP)……..55

3.3.2 Problema de emplazamiento (PP) ……...…….57

3.3.3 Problema de la mochila (KP) ……58

3.3.4 Problema de dimensión abierta (ODP) ……..…….59

3.3.5 Problema de la caja de embalaje (BPP) ……..….60
3.4 Aplicaciones de los problemas de corte y empaquetado en la industria.62
3.4.1 Industria papelera…….62
3.4.2 Industria maderera……..….63
3.4.3 Industria del vidrio…….…..63
3.4.4 Industria textil……...…63
3.4.5 Industria de pieles y calzado…….…… 64
3.4.6 Industria del metal……....64

4 El problema de corte unidimensional (1d CSP) ……...….67
4.1 El problema de corte unidimesional (1D CSP) formulaciones matemáticas…….…….…… 69
4.1.1 Modelo de asignación……..71
4.1.2 Modelo basado en patrones de corte……..…. 71
4.1.3 Modelo de corte único…….…..73
4.1.4 Modelo basado en grafos…….……...…..74
4.2 Métodos para la resolución del 1d CSP……... 75
4.2.1 Procedimientos basados en el uso de patrones……. 75
4.2.1.1 Enfoques basados en programación lineal…….…….. 75
4.2.1.2 Secuenciación heurística (enfoque constructivo)……80
4.2.1.3 Procedimientos híbridos…….……..81
4.2.2 Resolución del problema de corte sin el uso de patrones……..81
4.3 El problema de corte unidimensional con múltiples tamaños en stock (1d MSS CSP)……..…….…… 82
4.3.1 Modelo basado en patrones de corte……... 83
4.3.2 Modelo basado en teoría de grafos……...84
4.4 Resolución del problema de corte con múltiples longitudes……. 86
4.4.1 Particularización del algoritmo de generación de columnas…. 87
4.4.2 Enfoques basados en heurísticas de redondeo y problemas residuales……..…….. 88
4.4.3 Enfoques basados en métodos exactos……..……..88
4.4.4 Enfoques de secuenciación heurística……. 89
5 **Algoritmos genéticos para la resolución del problema 1dimensional MSSCSP**…….……..…….... 91

5.1 Modelización matemática del problema de corte de perfiles…..…….. 92
5.2 Métodos evolutivos en la resolución del problema de corte unidimesional……..……94
5.3 Una metodología basada en algoritmos genéticos para la resolución del problema de corte unidimesional…..……..101
5.3.1 Un algoritmo genético para la generación de patrones (fase 1) ……..103
5.3.2 Un algoritmo genético para la resolución del problema de corte (fase 2) …….. 111
5.3.3 Algoritmos para la mejora de las soluciones obtenidas (fase 3) ……..... 115
5.4 Implementación y resultados computacionales…..……...126
5.4.1 Generador de problemas de test……... 127
5.4.2 Generación de las Instancias…… 128
5.4.3 Evaluación del desempeño del generador de patrones…….130
5.4.4 Análisis de sensibilidad de los parámetros del algoritmo del problema de corte…….. 131
5.4.5 Resultados……134
5.4.6 Análisis del impacto de cada una de las fases de la Metodología sobre su desempeño…… 139
5.4.7 Coste computacional…… 143
5.4.8 Aplicabilidad de la metodología al caso real…....145
5.5 Ejemplo…….. 146

6 Algoritmos genéticos para la resolución del problema de secuenciación de patrones…….151
6.1 Los problemas de secuenciación de patrones……... 153
6.2 Algoritmos genéticos para la resolución de problemas de secuenciación de patrones en el corte de vigas……..... 159
6.2.1 El problema de minimización de la cantidad de paquetes abiertos (MOSP) …….. 160
6.2.2 El problema de minimización de la cantidad media de la extensión de pedidos (MORP) ……...162
6.2.3 Un algoritmo genético para la resolución independiente

del mosp y el morp en el corte de vigas……....164
6.3 Un algoritmo genético para la resolución conjunta del MOSP y el MORP en el corte de vigas (problema multiobjetivo) ……..…... 167
6.3.1 Optimización multiobjetivo…….....167
6.3.2 Un algoritmo genético para la resolución conjunta del MOSP y el MORP……. 171
6.4 Implementación y resultados computacionales……...175
6.4.1 Instancias……...175
6.4.2 Resultados……...177
6.4.3 Coste computacional…… 184
6.5 Una metodología para la resolución del problema global de corte y secuenciación…….. 185

7 Conclusiones y futuras líneas de trabajo……....189

Bibliografía……...191

Anexo Implementaciones en *®MATLAB*…….... 207
X.1 Implementaciones para la resolución del problema de corte……...207
X.2.1 Generador de problemas……....207
X.2.2 Generador de patrones……... 209
X.2.3 Algoritmo genético de resolución del problema de corte……...211
X.2.4 Algoritmo para la búsqueda de Patrones Incompletos de corte……...212
X.2.5 Algoritmo de Agrupamiento (A1) ……...213
X.2.6 Algoritmo genético residual (A2) ……...213
X.2.7 Algoritmo de longitudes menores (A3) ……... 214
X.2.8 Comparación de los algoritmo (A1, A2 y A3) ……..... 214
X.2 Implementaciones para la resolución del problema de secuenciación... 214
X.2.1 Implementación en *®MATLAB*……...214

ÍNDICE DE TABLAS

Tabla 2.1- Secciones estándar del perfil IPE…….. 24
Tabla 2.2- Volúmenes de corte……..35
Tabla 3.1- Categorías de problemas básicos de Corte y Empaquetado.....44
Tabla 5.1- Combinaciones con repetición de m+1 elementos tomados de N en N…….110
Tabla 5.2- Instancias para la experimentación……... 129
Tabla 5.3- Sensibilidad del algoritmo……..…..132
Tabla 5.4- Coeficientes de correlación de Pearson…….133
Tabla 5.5- Coeficientes de correlación de Pearson…….133
Tabla 5.6- Resultados obtenidos para Instancias de bajas demandas…...136
Tabla 5.7- Resultados obtenidos para Instancias de bajas demandas…...137
Tabla 5.8- Resultados porcentuales obtenidos para Instancias de bajas demandas…….. 138
Tabla 5.9- Resultados obtenidos para patrones de baja eficiencia…….....141
Tabla 5.10- Cantidad de patrones contenidos en solución y Contribución de la fase 3……..142
Tabla 5.11- Coeficientes de correlación por grupos de patrones..……143
Tabla 5.12- Coste computacional……...144
Tabla 5.13- Resultados porcentuales obtenidos de restos totales generados…….. 146
Tabla 5.14- Datos del problema ejemplo…….. 146
Tabla 5.15- Patrones Generados para el ejemplo…….147
Tabla 5.16- Soluciones obtenidas por el algoritmo genético de corte…..…149
Tabla 5.17- Soluciones finales obtenidas para el problema ejemplo……..150
Tabla 6.1- Cantidad media de patrones para instancias de bajas

demandas……..……..176
Tabla 6.2- Convergencia de los algoritmos genéticos para cada tipo de problema…….……. 177
Tabla 6.3- Relación entre cantidad máxima de paquetes abiertos y extensión media de pedido…….……...179

Tabla 6.4- Relación entre cantidad máxima de paquetes abiertos y extensión media de pedido…….……...180
Tabla 6.5- Desempeño del algoritmo para el problema Multiobjetivo……. 182
Tabla 6.6- Desempeño del algoritmo para el MOSP…….…….183
Tabla 6.7- Coste computacional …….…….184

ÍNDICE DE FIGURAS

Figura 2.1- Vigas IPE330 almacenadas…….....25
Figura 2.2- Imagen almacén de vigas de longitud estándar…….. 27
Figura 2.3- Imagen máquina de corte de perfiles estructurales…….....27
Figura 2.4- Imagen bancadas de entrada de material……..... 28
Figura 2.5- Imagen chatarra almacenada en contenedor……...29
Figura 2.6- Datos en el proceso de corte de perfiles……..... 30
Figura 2.7- Diagrama del proceso de corte de perfiles…….... 31
Figura 2.8- Identificación de material IPN……..... 33
Figura 2.9- Esquema de la filosofía 5S……..... 33
Figura 4.1- Modelo basado en grafos (Ejemplo)……....86
Figura 5.1- Representación basada en grupos……... 96
Figura 5.2- Representación basada en orden para un perfil en stock……...99
Figura 5.3- Representación basada en orden para varios perfiles…….. 99
Figura 5.4- Visión general del enfoque de resolución propuesto…….. 102
Figura 5.5- Codificación de los patrones de corte……... 105
Figura 5.6- Alternativa propuesta para la codificación de los patrones de corte……...106
Figura 5.7- Procedimiento de recombinación……... 114
Figura 5.8- Esquema general Fase 3: modificación soluciones……... 117
Figura 5.9- Representación del algoritmo de búsqueda de Patrones Incompletos……....119
Figura 5.10- Representación de un problema de corte y su solución…….120
Figura 5.11- Patrones Incompletos obtenidos para una solución del problema de corte……....121
Figura 5.12- *Patrones completos* e *incompletos* para una solución del

problema de corte…….. 122
Figura 5.13- Solución obtenida por algoritmo de agrupamiento……... 123
Figura 5.14- Solución obtenida por algoritmo genético residual……...124
Figura 5.15- Solución obtenida por algoritmo de longitudes menores…….126
Figura 5.16- Descripción de la Fase previa: Generación de problemas…….128
Figura 5.17- Histograma distribución patrones…….…. 130
Figura 5.18- Porcentaje de veces que cada método
alcanza la mejor solución……..…. 139
Figura 5.19- Comparación histograma distribución patrones……...141
Figura 5.20- Histograma distribución patrones ejemplo…….148
Figura 6.1- Ejemplo de *tour* en un grafo de 8 vértices……... 156
Figura 6.2- Solución mejorada a partir de un procedimiento 2 óptimo…… 158
Figura 6.3- Procedimientos de mutación de individuos…….. 167
Figura 6.4- Convergencia del algoritmo en cada tipo de problema…….. 178
Figura 6.5- Metodología para la resolución del Problema
Integrado (Corte & Secuenciación)……..…..186
Figura X.1- Entorno de trabajo ®MATLAB……... 216

CAPÍTULO 1

INTRODUCCIÓN

En el contexto de la Investigación Operativa y bajo el término problemas de *Corte y Empaquetado, Cutting and Packing Problems,* se engloban un conjunto de problemas de Optimización Combinatoria para una considerable variedad de aplicaciones industriales: metal, papel, madera, vidrio, textil o piel y calzado, por ejemplo.

El estudio de los problemas de *Corte y Empaquetado* se encuentra entre los primeros abordados por la Investigación Operativa, (Gilmore, Gomory 1961), (Gilmore, Gomory 1963), y (Kantorovich 1939). El interés de la comunidad científica en estos problemas, debido a sus aplicaciones prácticas y al significativo desarrollo tecnológico de herramientas para su modelado e implementación, ha llevado a un incremento considerable en el número de publicaciones sobre el tema en las últimas dos décadas. En este sentido, (Sweeney, Paternoster 1992) recogen, en su revisión bibliográfica, hasta cuatrocientas publicaciones de muy diversos orígenes en su revisión bibliográfica. Mientras que, (Wäscher et al. 2007) identifican entre 1995 y 2005, sólo en aplicaciones de problemas estándar, que excluyen variantes tipo multiobjetivo u *on line*, más de cuatrocientas.

La presente tesis se enmarca dentro de los mencionados problemas de *Corte y Empaquetado*, y fue motivada por los problemas vinculados a la planificación y

programación de las operaciones de corte en una **Empresa** de transformados metalúrgicos de la Comunidad Valenciana.

1.1 OBJETIVO Y ESTRUCTURA DE LA MEMORIA PRESENTADA

El objetivo de esta tesis doctoral es realizar un análisis y modelización de los problemas de programación de la producción en el contexto de una empresa de corte de vigas estructurales, así como la definición de diferentes metodologías y algoritmos que permitan resolverlos de manera óptima. En concreto se pretende:

- Desarrollar una metodología que permita resolver el **Problema de Corte.** Éste consiste en obtener, para un horizonte de planificación, una asignación eficiente entre los perfiles disponibles en stock y las vigas demandadas. En la asignación anterior se busca, por una parte la minimización de los restos generados por el corte (tanto de chatarra como de puntas de producción), y otra la reutilización de puntas de producción generadas en etapas anteriores.

- Resolver el **Problema de Secuenciación de Patrones** de manera que se optimicen simultáneamente dos objetivos: minimizar el espacio requerido para el apilamiento de pedidos en curso y minimizar la extensión temporal para procesar los pedidos.

La presente Memoria está organizada en siete capítulos, incluyendo éste.

En el **capítulo 2** se plantea el problema real del corte de perfiles estructurales que motiva la presente investigación. Se describe el sector industrial en el que se ubica la Empresa, sus principales líneas de negocio y su proceso productivo de corte de perfiles estructurales, identificando los principales problemas vinculados a la planificación de la producción de vigas. En este capítulo ya se propone una modelización matemática para el Problema del Corte de perfiles.

En los **capítulos 3 y 4** se revisa el estado del arte en el campo del trabajo. Ofrecen una mirada de conjunto a las distintas temáticas de la tesis: clasificación de los problemas de *Corte y Empaquetado*; modelos matemáticos; principales técnicas de resolución,… En

concreto, en el capítulo 3 se estudian los problemas de *Corte y Empaquetado* en su conjunto: estructura básica, principales aplicaciones y metodologías de resolución. Mientras que el Capítulo 4 se centra en la revisión de bibliografía específica que trata problemas de Corte similares al descrito en el capítulo 2.

La revisión bibliográfica se completa de forma pormenorizada en el resto de capítulos, fundamentalmente en sus introducciones, lo que permite encauzar el desarrollo de los mismos.

En **el capítulo 5** se aborda el **Problema de Corte,** en el que se generan patrones de corte eficientes y se determinan sus frecuencias de uso óptimas, de modo que queden satisfechas todas las demandas realizas por los clientes, a la vez que se minimiza el desperdicio y el exceso de vigas sobreproducidas. A tal efecto, se propone una metodología para su resolución basada en algoritmos genéticos y que consta de varias etapas. La dificultad de conocer a priori la validez de los métodos y técnicas hace necesario un análisis experimental de éstos para su evaluación. También se estudia la idoneidad de dicha metodología para el problema real.

En el **capítulo 6** se trata el problema **Secuenciación de Patrones de corte** con **naturaleza multiobjetivo.** Por un lado se busca minimizar el espacio requerido para el almacenaje de pedidos en curso, y por otro minimizar el tiempo requerido para procesar los pedidos. Las técnicas de resolución que se proponen también están basadas en algoritmos genéticos.

En el **capítulo 7** se presentan las conclusiones y se dejan abiertas distintas líneas de trabajo futuro que han ido surgiendo a lo largo del desarrollo de la presente investigación y que merecerán de una especial atención.

Por último, en un **Anexo** se describen las principales funciones programadas para implementar los métodos expuestos tanto en el capítulo 5 como en el capítulo 6. El software utilizado para toda la implementación ha sido ®*MATLAB* en su versión *2007Rb*.

CAPÍTULO 2

PROBLEMÁTICA EN EL PROCESO DEL CORTE DE PERFILES ESTRUCTURALES

En este capítulo se plantea el problema real del corte de perfiles estructurales para el que en los capítulos siguientes se propondrán métodos de resolución eficientes.

Se describe el actual proceso productivo para el corte de perfiles estructurales de acero en una empresa de la Comunidad Valenciana, así como las restricciones a tener en cuenta a lo largo de la investigación. Se identifican también los principales problemas vinculados a la planificación de la producción de vigas: **Problema de Corte de perfiles** y **Problema de Secuenciación de patrones** y se relatan las cifras más significativas relativas a: volúmenes de producción, puntas de producción, niveles de chatarra generados, tipos de perfiles, vigas demandadas,… que serán de utilidad para la validación de los métodos planteados en los subsiguientes capítulos. Por último, se plantea el problema a resolver y se enumeran las distintas fases en las que se organizará la investigación.

2.1 EL SECTOR DEL METAL

2.1.1 Perspectiva general

La industria, los servicios y el comercio del Metal en España superan las 150.000 empresas y emplean a más de 1.500.00 personas. Su Valor Añadido Bruto en 2007 supuso alrededor del 9% del PIB (Confemetal).

La industria del metal constituye una de las industrias básicas más importantes en los países industrializados y su grado de madurez es un exponente claro del desarrollo industrial del país. La industrial del metal en España supone cerca del 40% de la producción industrial española, de cada 100 empleados en la industria española, 37 son del *Metal*. Anualmente la industria exporta en torno al 50% del total de las exportaciones españolas. De éstas, el 75% tienen como destino la UE. En España las empresas del *Metal* representan el 5% del total y de ellas 78.000 son establecimientos industriales. En el *Metal* las PYMES son mayoría: un 86% son empresas de menos de 10 asalariados y un 98% con menos de 100 (Confemetal).

Se incluyen en este sector diferentes actividades industriales, que se pueden diferenciar por los productos que obtienen. En cuanto a la estructura del sector, habitualmente la industria se divide en dos grandes grupos: industrias básicas e industrias de transformación (Cavallé 1975). Se consideran como industrias metalúrgicas básicas aquellas de obtención de hierro, aceros especiales, semiproductos y primeros laminados (estructurales, comerciales y planos). Son industrias metalúrgicas de transformación las que se dedican a las siguientes actividades:

- Laminación en frío. A partir de laminados fabrican chapas, flejes y bandas.
- Forja y estampación.
- Trefilerías. Fabricación de alambres y sus derivados: telas, clavos, cables,...
- Calibrados: laminados de precisión.

Dentro de la segunda categoría incluiríamos empresas de fabricación de:

- Estructuras metálicas (puentes metálicos, marcos, puertas, ventanas).
- Calderería (calderas, accesorios, depósitos de agua).
- Herramientas manuales.

- Talleres metálicos.

2.1.2 Referencia al sector del Metal en la Comunidad Valenciana

Históricamente en la Comunidad Valenciana el desarrollo de otras industrias, así como la evolución del tráfico ferroviario y marítimo, favoreció la implantación y posterior expansión del sector metalúrgico, con modernos centros en Valencia y Alcoy. La Primera Guerra Mundial estimuló la creación y ampliación de empresas en el sector de maquinaria y de transformados metálicos. En esta época se instaló la siderurgia de Sagunto (1922) y se creó la Unión Naval de Levante (1924). En el periodo posterior a la guerra civil se crearon numerosos talleres y a partir de 1953 se produce un rápido proceso de recuperación industrial, propiciado en gran medida por el *Programa de Ayuda Americana*. Durante el periodo de 1964 a 1973 se produce un nuevo desarrollo del sector, por la instalación de la factoría de automóviles de Ford e IBM, no sólo por el número de puestos de trabajo directos, sino por la cantidad de industrias auxiliares que han ido surgiendo como consecuencia (Dalmau et al. 1993)

El sector en la Comunidad Valenciana es muy heterogéneo, englobando un conjunto de actividades económicas muy dispares. Sin embargo existe una característica común a todas ellas: fabricar, reparar o instalar productos de metal, lo cual permite a dichas empresas el uso de tecnologías similares. Éste es uno de los principales sectores productivos de la Comunidad Valenciana, generando el 21% del Valor Añadido Bruto de la Industria, ocupando alrededor del 25% de los empleados; y representando el 31% del Comercio exterior. La participación del sector metalúrgico en la Comunidad Valenciana es apreciablemente superior a la de las industrias tradicionales valencianas (muebles, calzado, azulejo, textil y juguete).

Una de las características que siempre se han señalado como básicas del sector metalúrgico es la pequeña dimensión de sus unidades productivas, y como consecuencia de ello, la escasa tecnificación de los procesos productivos utilizados y la baja necesidad de inversión para la puesta en marcha de las empresas. Al frente de estas empresas predomina el operario emancipado más que el directivo o el empresario propiamente dicho. Si se tiene en cuenta que en gran medida los clientes de estos operarios van a ser

prácticamente los mismos que los de su empresa originaria, se puede prever el elevado nivel de competencia existente en el sector. Así se explica, en gran parte, por qué las principales estrategias de «marketing» se basan en la guerra de precios, a costa de la consecuente reducción de los márgenes comerciales. (Sánchez et al. 1999)

El subsector metalmecánico en la Comunidad Valenciana también es a su vez muy heterogéneo, pero utiliza tecnologías similares en sus procesos basados en los conocimientos de la metalurgia y la mecánica. Según (Dalmau et al. 1993), el tamaño de las empresas varía dentro de límites muy amplios. Por lo que se refiere a la dimensión de las empresas del subsector, se constata el claro dominio de la pequeña y mediana empresa.

2.2 CASO REAL EN UNA EMPRESA DE LA COMUNIDAD VALENCIANA

La empresa cuyo proceso de transformación de perfiles estructurales ha sido objeto de estudio en la presente tesis, inició su actividad empresarial en 1954, con la apertura de un almacén de hierro en Valencia. Desde entonces ha sufrido un crecimiento constante en cuanto a sus volúmenes de negocio, ámbito geográfico y facturación, a la vez que ha experimentado una diversificación de su actividad. Actualmente ofrece una amplia gama de servicios relacionados con las siguientes líneas de negocio:

- Distribución de productos siderúrgicos.
- Actividad de ferralla.
- Suministro industrial.
- Centros de servicio del acero: servicio sobre chapa.
- Centros de servicio del acero: servicio oxicorte y corte por plasma.
- Centros de servicio del acero: servicio sobre perfiles estructurales.

Así pues, las principales actividades de la empresa son: la Distribución de productos siderúrgicos; el Suministro industrial y de maquinaria; y la Distribución y transformación de los siguientes tipos de productos:

- **Productos planos**. Servicios de corte longitudinal y trasversal de bobinas de chapa en toda su gama de anchos, largos, desarrollos, espesores, calidades y acabados superficiales.
- **Oxicorte y cizallado de chapa industrial**. Oxicorte de piezas de chapa gruesa a medida según las necesidades de los clientes a través de sistemas de CAD/CAM en distintas calidades y espesores.
- **Corte por plasma de alta definición**. Corte de piezas de espesores entre 0,5 y 15mm, con tolerancias muy ajustadas y superficie de corte perpendicular, que permite lograr una deformación de la pieza mínima.
- **Perfiles estructurales**. Corte, taladro granallado y pintado de perfiles estructurales en todos sus tipos y largos.
- **Producto de construcción**. Construcción metálica: calderería y estructuras de acero.
- **Actividad de ferralla**.

En concreto, el proceso que se estudia en esta tesis es el corte de vigas de distintos perfiles estructurales. La actividad de la planta en la que se realiza dicho corte, abarca el almacenaje y corte de perfiles, así como el suministro de otros productos laminados, conformados y de acero corrugado en una amplia gama de espesores, calidades y tamaños.

Únicamente los perfiles estructurales de acero laminado son los que sufren un proceso de transformación geométrica en la planta (corte en su eje longitudinal). Para la realización de dicha operación de corte, las instalaciones cuentan con dos maquinas de corte de 9 y 4kW de potencia motriz respectivamente.

2.2.1 Descripción de su proceso de corte de vigas

El corte de vigas se realiza bajo pedido, ya que se trata de un producto personalizado y muy heterogéneo, con grandes diferencias entre unos pedidos y otros, tanto en cuanto a las cantidades demandadas, como en cuanto al tipo de perfiles y sus longitudes. Los clientes solicitan unas cantidades concretas de vigas estructurales de una determinada longitud, obtenidas a partir del corte longitudinal (recto o en ángulo) de vigas más largas disponibles en stock de longitudes estándar.

En esta empresa se cortan cinco tipos de perfiles distintos (IPE, IPN, UPN, HEA y HEB). Cada perfil está disponible en almacén en sus tamaños de sección comercial y en entre tres y seis longitudes estándar. El corte de perfiles se puede realizar hasta longitudes de 24m de acuerdo a las siguientes características:

- Corte recto 90º: 1.250 x 600mm^2 (ancho x alto).
- Corte en ángulo 45º: 814 x 600mm^2 (ancho x alto).
- Corte en ángulo 60º: 620 x 600mm^2 (ancho x alto).
- Corte en ángulo 60º: 620 x 600mm^2 (ancho x alto).

Longitud en corte recto (m)	Hasta 8	de 8 a 15	Más 15
Tolerancia estándar inferior a la indicada en la norma UNE. ENV 1090-1 (mm)	+/- 2	+/- 3	+/- 4

A modo de ilustración, en la siguiente tabla se indican las secciones estándar para el perfil IPE. En la siguiente figura, se muestra una fotografía de la empresa en la que están almacenadas y correctamente identificadas vigas del tipo IPE330. Se observa que junto a las vigas de longitud estándar 7420mm también hay apiladas dos puntas de producción de longitudes 7140 y 7070mm.

Medidas	Kg/m	Sección en mm (h)	Sección en mm (b)	Sección en mm (e)	Sección en mm (e1)
80	6,15	80	46	3,8	5,2
100	8,3	100	55	4,1	5,7
120	10,66	120	64	4,4	6,3
140	13,22	140	73	4,7	6,9
160	16,2	160	82	5	7,4
180	19,27	180	91	5,3	8
200	22,96	200	100	5,6	8,5
220	26,86	220	110	5,9	9,2
240	31,47	240	120	6,2	9,8
270	37	270	135	6,6	10,2
300	43,26	300	150	7,1	10,7
330	50,33	330	160	7,5	11,5
360	58,53	360	170	8	12,7
400	67,96	400	180	8,6	13,5
450	79,54	450	190	9,4	14,6
500	92,97	500	200	10,2	16
550	108,65	550	210	11,1	17,2
600	125,05	600	220	12	19

Tabla 2.1- Secciones estándar del perfil IPE

Figura 2.1- Vigas IPE330 almacenadas

El plan de producción se realiza cada día en función de los pedidos confirmados por el departamento comercial. Lo componen distintas órdenes de fabricación a partir de los pedidos. Un pedido está definido por la siguiente información:

- Cantidad de vigas demandadas de un perfil específico (tipo y dimensiones de la sección).
- Longitud de corte de dichas vigas.
- Identificador del cliente y plazo de entrega del pedido.

En cada orden de fabricación, sin embargo, se considera solamente un único tipo de viga que podrá pertenecer a uno o más pedidos y por tanto al mismo o a diferentes clientes. Diariamente se realiza una reunión entre la dirección y los mandos intermedios donde se concreta la lista de órdenes de fabricación a ejecutar ese día, sobre la base de los pedidos demandados y sus plazos de entrega. A lo largo del día se revisa hasta tres veces el programa definido y se van añadiendo o modificando nuevas órdenes de fabricación si fuera necesario. Así se define un calendario de lanzamientos, basándose en criterios de disponibilidad de material y plazos de entrega.

Desde que el departamento comercial de la empresa realiza la oferta a un cliente, asociada a un pedido demandado, hasta que éste se expide, la secuencia de pasos que se siguen en los distintos departamentos de la empresa se podría sintetizar en los siguientes puntos (descritos en más detalle en la figura 2.7) :

- Recepción del pedido de vigas

- Estudio de la pieza y presupuesto
- Anidado en una punta que cumpla las medidas y la calidad deseadas
- Preparación del programa de corte
- Corte realizado mediante una sierra propulsada por motor eléctrico
- Preparación del material
- Distribución al cliente

Como se ha comentado anteriormente para la operación de corte de perfiles, se dispone de dos maquinas de corte por cinta. Una de ellas es capaz de cortar perfiles estructurales de hasta 24 m gracias a su bancada, y es la utilizada para cortar grandes paquetes de vigas, ya que sus rodillos son capaces de arrastrar transversalmente mas de 5 toneladas. La sierra de la otra máquina es más pequeña que la de la anterior, y posee una bancada capaz de alojar vigas de hasta 15-16 m. Esta máquina se utiliza para cortar perfiles pequeños, para perfiles que requieren un corte angular y para pequeños paquetes. Su apertura para entrada de material llega a los 620 cm de ancho x 350 cm de alto.

En general las características de las sierras de ambas máquinas permiten el corte en inglete en ambos sentidos, tanto para secciones huecas como macizas. La guía de la sierra se realiza por una combinación de rodillos y está endurecida de tungsteno para permitir una mayor precisión en el corte. Se dispone de una rueda motorizada instalada separadamente del motor con rodamientos robustos, que permite el desplazamiento de las vigas a lo largo de la máquina. Con el fin de asegurar el pisado de la pieza en el corte, la máquina dispone de una doble unidad de pisado, acanalado y endurecido. Las cintas de sierra tienen una duración de entre una y dos semanas de trabajo, dependiendo de los tipos de corte que se realicen con ella. En la figura 2.2 se muestra una fotografía de una de las máquinas de corte, la cual es capaz de cortar varios paquetes a la vez.

Por otro lado, para la manutención de material se dispone de varios puentes grúa que permiten la manipulación y el transporte de las vigas entre las distintas zonas de la nave. En lo que se refiere a los perfiles estructurales, se distinguen las siguientes zonas:

- Almacén de materia prima en la que se encuentran ordenados los perfiles en pasillos, como se muestra en la siguiente figura 2.3.

- Zona cercana a la línea de corte para ubicación y selección del material (vigas) que va a ser cortado a continuación.
- Líneas de corte con rodillos para la alimentación de la sierra de corte.
- Zona para la clasificación del material cortado a la salida de la máquina.
- Contenedor para depositar los restos de chatarra que sobran del corte
- Zona de almacenaje de pedidos listos para su expedición.

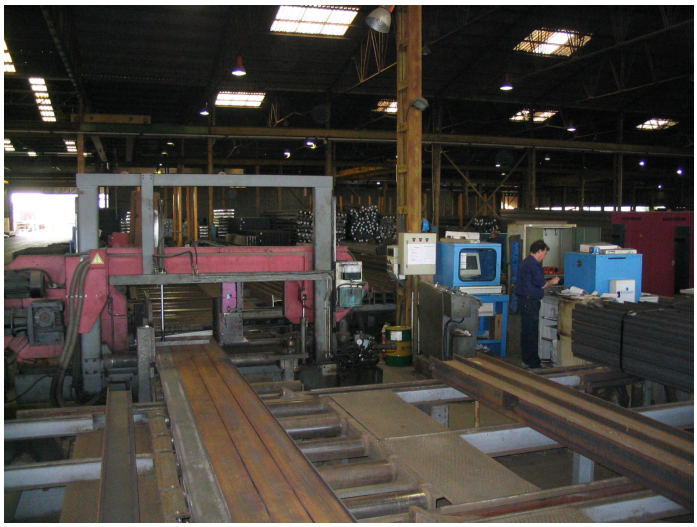

Figura 2.2- Maquina de corte de perfiles estructurales

Figura 2.3- Almacén de vigas de longitud estándar

Una vez el operario tiene la orden de fabricación, en la que se indica la cantidad, longitud, perfil y sección del material que se va a cortar, los operarios ejecutan las

siguientes etapas que constituyen el propio proceso de corte. En función de la carga de trabajo, las etapas las realiza uno o varios operarios. En épocas de altas demandas hasta cuatro operarios trabajan en las dos líneas de corte: uno alimentando las líneas, dos cortando el material y el cuarto transportando el material de la salida de la línea hasta la zona contigua de playas de expedición):

- Se localizan, en el almacén de materia prima, las vigas que se van a cortar en la orden de producción. Por medio de un puente grúa, éstas se trasladan hasta la zona de corte y se depositan en un espacio cercano a la máquina, donde se selecciona la pieza o piezas que se vayan a cortar. En la siguiente fotografía se muestran las bancadas de entrada de material a la máquina de corte (figura 2.4).

Figura 2.4- Bancadas de entrada de material

- Mediante otro puente grúa, el material se traslada a la línea de corte, depositándolo sobre unos rodillos que lo trasladarán a la entrada de la sierra. Para que el material quede completamente anclado, se dispone de unas mordazas metálicas automáticas que lo centran y fijan, y por medio de un sistema de cadena y engranajes se va alimentando la sierra.

- La operación de corte propiamente dicha se programa desde el panel de mando de la sierra, especificando la longitud de la pieza que se va a cortar y la velocidad del corte (variable).

- Una vez la pieza se ha cortado, mediante un sistema de rodillos análogo al de la alimentación de la línea el material, se desplaza hasta la salida de la línea.

- A través de un puente grúa, el material cortado se clasifica a la salida de la línea en una zona auxiliar disponible al tal efecto. Muchos perfiles largos son utilizados para producir el producto de varios clientes, por lo que las vigas que van saliendo de la cortadora las debe separar el operario por pedido. La chatarra se deposita en un contenedor, ver la figura 2.5, y en caso de haber generado puntas reutilizables en un futuro, estás se ubican en el almacén de materia prima.

Figura 2.5- Chatarra almacenada en contenedor

- Por último, las vigas se apilan en la zona de expedición mediante el uso de un puente grúa en espera de que se carguen en camiones para el envío al cliente.

Para el proceso de corte de perfiles se identifican los siguientes datos como entradas (ver figura 2.6):

- Datos sobre materiales: Información asociada al pedido (cliente, plazos de entrega, cantidades y longitudes de las vigas, secciones, etc.) y datos sobre disponibilidad de material en almacén y en proveedores (stocks de puntas, stocks de perfiles estándar)
- Medios: Procedimientos, normas, guías, condiciones, etc.

- Recursos con los que se cuenta para la preparación del pedido: maquinaria, mano de obra,…

Como salidas se tienen (ver figura 2.6):

- Pedido: vigas cortadas a medida y la información asociada
- Datos de control: medida de los recursos utilizados e indicador de nivel de satisfacción del cliente

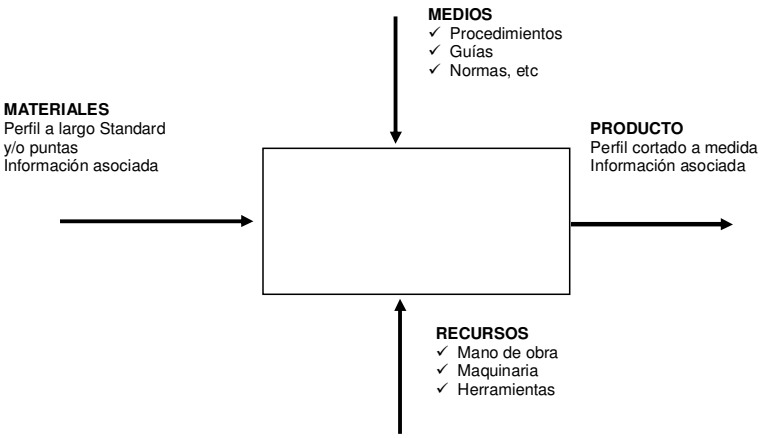

Figura 2.6- Datos en el proceso de corte de perfiles

En la figura 2.7 se representa, mediante un diagrama de flujo, el proceso del corte de vigas tal como se lleva a cabo actualmente en la empresa, de forma que se identifican las principales actividades y secuencias del proceso.

Del procedimiento utilizado, cabe destacar el hecho de que los pedidos pasan por una fase previa en la que no son pedidos en firme. De modo que a lo largo de esta fase previa se estudia la viabilidad técnica del posible pedido: disponibilidad de perfiles con la sección demandada, tanto en almacén propio como de proveedores; disponibilidad de longitudes de dichos perfiles; plazos de entrega, etc. Con los datos obtenidos (viabilidad, presupuesto y plazos), el departamento comercial emite una oferta al cliente y éste decide confirmarlo o no. Una vez el pedido se materializa y se convierte en pedido en firme, éste pasa al listado de órdenes de fabricación manejado por el jefe de almacén, el cual en base a su experiencia selecciona un material u otro para el corte.

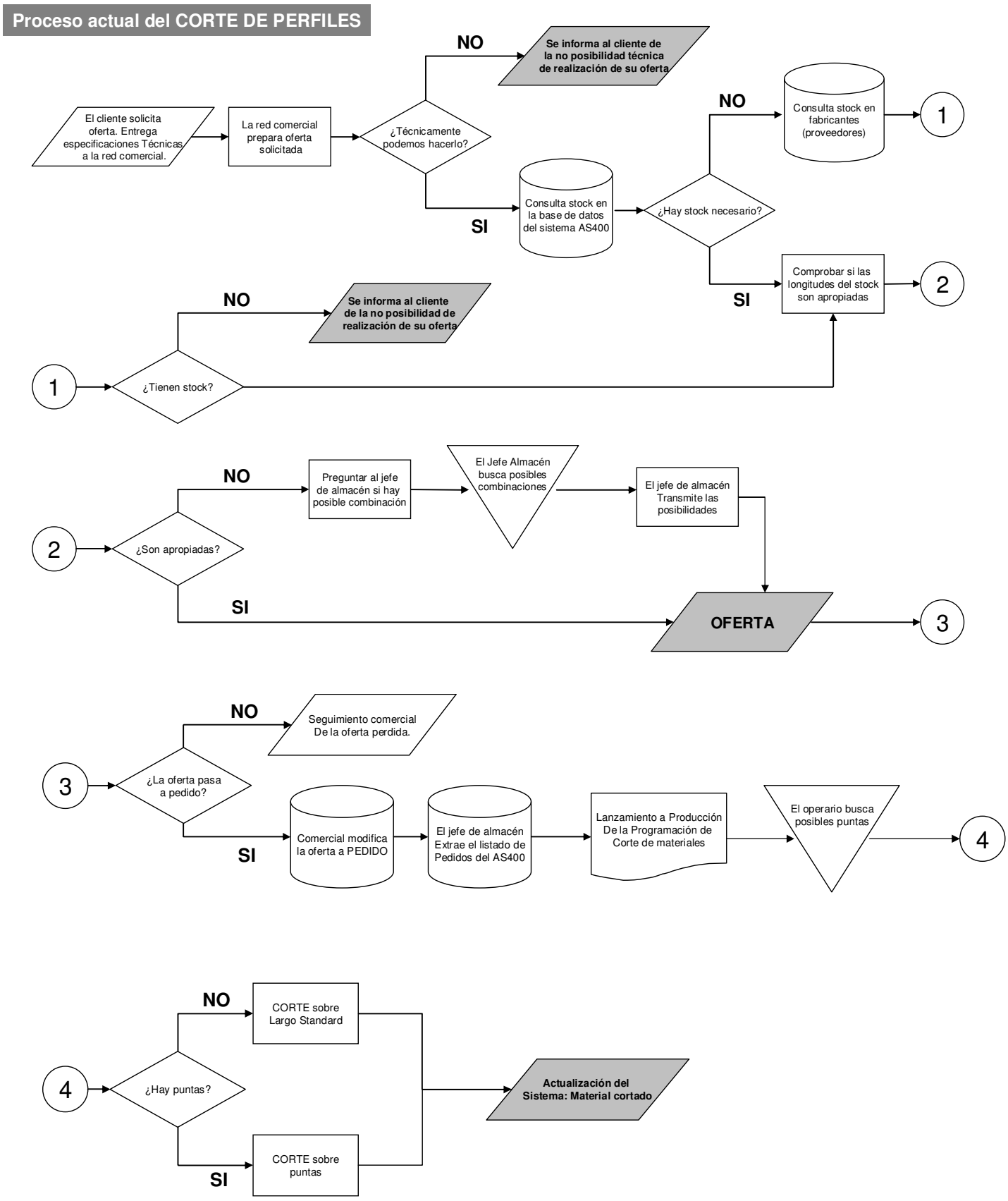

Figura 2.7- Diagrama del proceso de corte de perfiles

Podríamos destacar las siguientes imperfecciones en el proceso:

- La oferta la realiza el departamento comercial por lo que en determinadas ocasiones se pierde información operativa de mucho valor.

- No existe una programación de la producción en base a patrones de corte eficientes sino una optimización efectuada por la "buena memoria" del Jefe de Almacén y del operario de la máquina.

- El listado de órdenes de fabricación en curso lo manejan los operarios de almacén, que buscan las puntas y/o el material de largo estándar.

- La búsqueda de puntas se realiza a veces directamente por el operario asignado a la máquina de corte, lo cual reduce el número de horas de corte efectiva de máquina, aumentando el tiempo de ciclo. Incluso en ocasiones este mismo es quien tiene que suministrarse el material a la maquina con el puente.

- Existen dificultades relacionadas con el almacenaje y la expedición de material, principalmente en aspectos de control e identificación (tanto de la materia prima como del producto terminado).

Así pues podemos identificar dos **problemas** básicos que habría que resolver en lo que se refiere al proceso del corte de perfiles estructurales:

- **Necesidad de determinar un programa de producción eficiente**: que consistirá en el establecimiento de patrones de corte óptimos, sus frecuencias de uso adecuadas y las secuencias de corte apropiadas, de modo que: se evite el desperdicio de material; se utilice mejor el espacio disponible; y se reutilicen las puntas de producción. Todo el proceso de optimización anterior **constituye el objeto de estudio en la presente tesis**. De modo que, a partir de la modelización del problema de corte y secuenciación se propondrán algoritmos eficientes que permitan obtener el programa de corte para un horizonte de planificación.

- **Mejora de los procedimientos** de almacenaje, identificación y búsqueda de materiales en los almacenes. Se propone la aplicación de la metodología japonesa de las 5"s": *clasificar, ordenar, limpiar, estandarizar* y *disciplina*, (Hirano 1998), ligada a la Fábrica Visual y que permita resolver los siguientes problemas identificados en el caso de la empresa respecto al almacenaje tanto de materias primas como de productos terminados:
    - Su "clasificación" no permite una visualización rápida de sus características.
    - Infrautilización del espacio disponible.
    - No existe un sistema de claro de identificación de materiales (en la imagen siguiente se muestra el sistema mejorado para la identificación del material)
    - Poca limpieza por flejes, papeles y suciedad en los almacenes

Figura 2.8- Identificación de material IPN

Figura 2.9- Filosofía 5S

2.3 DATOS Y VOLÚMENES DEL PROCESO DEL CORTE DE PERFILES (1D MSS CSP)

A continuación se indican algunos datos que se manejan en el problema real de la empresa y que permitirán determinar hasta qué punto la metodología planteada es eficiente.

Como se ha indicado anteriormente la cantidad de **tipos de largos estándar** disponibles para sección (tipo y tamaño) oscila entre 4 y 6. Por otro lado, la política actual es la de generar pocas puntas de fabricación, por lo que apenas se tiene un par de longitudes adicionales en stock debido a puntas de fabricación. De modo que la cantidad de perfiles diferentes de una misma sección que hay disponible en stock está entorno a **cinco** y ocasionalmente mayor, llegando en raras ocasiones a diez.

La cantidad de **vigas diferentes demandadas** de una misma sección (tipo y dimensiones) para un el horizonte del programa de producción (un día) de una misma sección es baja: al rededor de **cinco** y pocas veces llegando a diez. Esto es debido, a pesar de que en un día se procesan bastantes órdenes de producción, a la diversidad de tipos de secciones (IPE, IPN, HEA, HEB y UPN) y todos sus tamaños disponibles de cada una.

Además, la **diversidad en las longitudes demandas de las vigas es, en general, muy elevada**, demandándose por igual vigas pequeñas y vigas grandes, si bien es cierto que para algunos perfiles concretos se demandan mucho longitudes específicas.

Cabe destacar que en épocas de mucho volumen de trabajo, el espacio de la zona de expedición ha sido insuficiente para alojar todos los pedidos demandados.

De acuerdo con el histórico de datos se ha estimado el volumen de chatarra (puntas de longitud menor a 400mm) generado por el proceso de corte en un 2,6%, sin contar las puntas generadas mayores de 400mm que se devuelven a almacén de materia prima en espera de poder ser reutilizadas. El volumen medio de puntas reutilizables generado es del 12% del volumen total cortado. A modo de ejemplo ilustrativo se muestran los datos disponibles en el histórico de la empresa de uno de los últimos años.

	Enero	Feb	Marzo	Abril	Mayo	Junio	Julio	Agosto	Sep	Oct	Nov	Dic	Media
Volumen Corte (t)	609.82	926.29	959.00	780.48	577.63	725.84	548.95	280.49	621.38	577.55	627.71	594.96	660.746
Volumen Puntas generadas (t)	53,259	47,77	63,106	43,016	41,323	43,47	40,335	42,334	40,923	40,508	72,202	42,560	76,003
Volumen Chatarra (kg)	7.830	33.580	16.450	20.140	19.900	15.670	12.060	6.670	24.910	18.660	16,708	20.050	17.587
Puntas (%)	10%	5%	7%	6%	7%	6%	7%	15%	7%	7%	12%	7%	12%
Chatarra (%)	1,3%	3,5%	1,7%	2,5%	3,3%	2,1%	2,1%	2,3%	3,9%	3,1%	2,62%	3,4%	2,6%

Tabla 2.2- Volúmenes de corte

2.4 PLANTEAMIENTO DEL PROBLEMA A RESOLVER

El problema de establecer un programa de producción eficiente para el corte de perfiles lo podemos dividir a su vez en dos partes claramente diferenciadas, y que se abordarán de forma sucesiva en los capítulos 5 y 6 de la presente tesis. Por un lado, encontramos el **Problema de Corte** (*Cutting Stock Problem*) propiamente dicho. Por otro el **Problema de Secuenciación de los Patrones de Corte** (*Pattern Sequencing Problem)*.

Como consideración previa, hay que distinguir entre las **vigas demandadas y las vigas de largo estándar** disponibles en stock en el almacén, y que se usan para el corte. A las primeras las denominaremos **vigas** y a las segundas **perfiles**. De forma que el apelativo perfil en este caso no va a ir vinculado a una forma o dimensiones específicas de sección sino a una longitud.

El **Problema de Corte** consiste en obtener una asignación entre los perfiles disponibles en stock y las vigas demandadas para un horizonte de planificación, persiguiendo objetivos como la minimización de restos (tanto de chatarra como de puntas de producción) y la reutilización de puntas de producción generadas en etapas anteriores. La asignación se realiza mediante el uso de **patrones** de corte, por lo que el problema se reduce a determinar las **frecuencias de corte** de los patrones. Dichas frecuencias de corte deberán estar sujetas a restricciones relativas a la satisfacción de demanda, a la no

superación de los stocks disponibles y al cumplimiento de los plazos de entrega establecidos.

El **Problema de Secuenciación de Patrones** buscará **una secuencia óptima** de los mismos de manera que se optimicen simultáneamente dos objetivos (**problema multiobjetivo**): minimizar las necesidades de espacio para el apilamiento de pedidos en curso de acuerdo con restricciones espaciales y minimizar la extensión temporal requerido para procesar los pedidos.

Así pues, para abordar el problema globalmente se seguirán las fases que se detallan a continuación:

- A partir de la descripción del problema industrial real realizada en este capítulo, se procederá a hacer una búsqueda bibliográfica en la literatura, a fin de recopilar los desarrollos realizados relacionados con el problema objeto de estudio. Se trata, por tanto, de detectar el interés que suscita el problema tanto dentro del entorno académico como del industrial (capítulo 3).

- Se describirán las principales formulaciones matemáticas que han aparecido en la literatura en el contexto de los *Problemas de Corte*, así como los principales métodos y estrategias de resolución que se han utilizado para resolver este tipo de problemas (capítulo 4)

- Se va a desarrollar una metodología basada en técnicas metaheurísticas (algoritmos genéticos) que permita resolver el Problema de Corte de manera eficiente. Se realizará un estudio experimental para evaluarla y compararla con otros métodos que han venido utilizándose en la resolución de problemas de corte del mismo tipo (capítulo 5).

- Se va a desarrollar una metodología basada en algoritmos genéticos que permita resolver el problema multiobjetivo de la secuenciación de patrones, obteniendo los puntos óptimos del frente de Pareto (capítulo 6)

- Por último se mostrarán los resultados y conclusiones obtenidos así como líneas futuras de trabajo que abre esta tesis (capítulo 7).

CAPÍTULO 3

LOS PROBLEMAS DE CORTE Y EMPAQUETADO

En el presente capítulo se realiza una revisión de lo que en el contexto de la Investigación Operativa y bajo el término *Problemas de Corte y Empaquetado, Cutting and Packing Problems* (C&P) engloba un conjunto de problemas de Optimización Combinatoria con una considerable variedad de aplicaciones de tipo industrial. El problema del corte de perfiles, estudiado en el capítulo anterior, pertenece a este tipo de problemas *de Corte y Empaquetado*. En primer lugar se identificará los principales tipos de problemas de este tipo, se describirá la estructura básica común a todos ellos y posteriormente se revisarán los principales métodos utilizados en su resolución.

En un problema de optimización, el objetivo es encontrar la mejor solución posible. Se ha de lograr un valor para las variables de decisión del problema de entre un conjunto finito de soluciones, de forma que satisfaciendo unas restricciones a las que están sujetas, se alcance un valor máximo o mínimo para una función objetivo definida.

El estudio de los problemas de *Corte y Empaquetado* se encuentra entre los primeros abordados por la Investigación Operativa, (Gilmore, Gomory 1961), (Gilmore, Gomory 1963) y (Kantorovich 1939). El interés de la comunidad científica (internacional), debido a sus aplicaciones prácticas, junto con el significativo desarrollo tecnológico de herramientas para su modelado e implementación, ha llevado a un incremento

considerable en el número de publicaciones sobre el tema en las últimas dos décadas. (Sweeney, Paternoster 1992) recogen hasta 400 publicaciones en su revisión bibliográfica, y (Wäscher et al. 2007) identifican entre 1995 y 2005 más de 400 publicaciones sobre problemas standard (se excluyen extensiones o variantes tipo multiobjetivo, on-line, problemas estocásticos,…)

A lo largo de los años, en la literatura, el término *Corte y Empaquetado* se ha convertido en sinónimo de una amplia gama de investigaciones de muy diversos orígenes, lo que en muchos casos ha supuesto que un mismo problema apareciera en las publicaciones bajo diferentes nombres. A primera vista y por el hecho de que han aparecido bajo diferentes denominaciones, podría pensarse que se trata de problemas independientes y sin ninguna relación. Pero en el fondo todos ellos poseen una misma estructura lógica común que los caracteriza. En el desarrollo del presente capítulo, y a medida que se vaya tratando la clasificación de los problemas de *Corte y Empaquetado,* se estudiarán sus principales tipos básicos de acuerdo a sus propiedades y estructura.

Los problemas de *Corte y Empaquetado* pueden modelizarse en términos de programación matemática. En cuanto a su resolución y de acuerdo con la Teoría de la Complejidad Computacional la mayoría de ellos son del tipo NP-Completos, (Garey, Jonson 1979). Es decir, si bien la verificación de una solución al problema puede efectuarse en tiempo polinomial (propiedad NP), muy **probablemente** en el caso de la resolución del problema no ocurra así (NP-Completo $\cap$ P = 0), y por tanto se trata de problemas de “difícil” solución.

Soluciones óptimas a problemas de *Corte* con variables enteras únicamente se encuentran para tamaños muy pequeños y alejados de lo que es habitual en las decisiones reales de planificación industrial, así que los métodos basados en procedimientos de aproximación heurística son en muchos casos las técnicas empleadas para su resolución

3.1 DESCRIPCIÓN DE LOS PROBLEMAS DE CORTE Y EMPAQUETADO

3.1.1 Estructura Básica

En todos los problemas pertenecientes al *Corte y Empaquetado* se identifica una estructura lógica común que consta de los siguientes elementos:

- Dos grupos de datos básicos cuyos elementos consisten en "figuras" geométricas definidas en una, dos, tres o incluso más dimensiones:
    - Un conjunto de piezas grandes que se ha dado en llamar **objetos** (*objects*), se encuentran disponibles en stock (*input*)
    - Un conjunto de piezas más pequeñas o **ítems** (i*tems)* con unas demandas que hay que satisfacer (output)

La denominación de objetos e ítems es la utilizada en la clasificación de (Dyckhoff 1990).

El problema consiste en obtener diferentes subconjuntos mediante la agrupación de todos o algunos de los ítems de forma que estos subconjuntos se asocien a todos o algunos de los objetos sin violar las condiciones geométricas (los ítems de un subconjunto deben “caber” en el objeto al que se asocian) de forma que se optimice una función objetivo dada. Una consideración a tener en cuenta es que dependiendo del tipo de problema, la solución puede utilizar/contener todos o algunos de los ítems y todos o algunos de los objetos.

(Wäscher et al. 2007) identifican cinco subproblemas como consecuencia de esta estructura y que deben resolverse de manera simultánea para poder obtener el óptimo global. Estos son:

- Problema de selección de los objetos
- Problema de selección de los ítems
- Problema de agrupamiento de los ítems en subconjuntos
- Problema de asignación de los subconjuntos a los objetos

- Problema de distribución de los ítems en cada uno de los objetos de acuerdo a sus restricciones geométricas

3.1.2 Clasificación de los Problemas de Corte y Empaquetado

La gran variedad de aplicaciones referenciadas en la literatura llevó a que (Dyckhoff 1990) desarrollara una tipología que permitió clasificar los problemas de *Corte y Empaquetado*.

Una tipología en este caso consiste en organizar los problemas en categorías homogéneas de acuerdo con un conjunto de criterios dados. Por otro lado debe ser capaz de suministrar una base consistente para un análisis estructural de cada uno de los tipos de problemas, y permitir tanto la definición de problemas estándar como el desarrollo de modelos y algoritmos e incluso generadores de instancias.

A tal efecto y de acuerdo con la estructura identificada, establece una clasificación basada en el uso de cuatro características:

- Dimensionalidad: Número mínimo de dimensiones necesarias para describir el problema (*Dimensionality)*

- Tipo de asignación entre ítems y objetos (*Kind of assignment*):
  - (B) Se asignan todos los objetos a una selección de ítems (*Beladeproblem*, en alemán)
  - (V) Se asignan todos los ítems a una selección de objetos (*Verladeproblem*, en aleman)

- Surtido de **objetos** grandes (*Assortment of large objects*)
  - (O) Un único gran objeto
  - (I) Muchos objetos, todos ellos iguales
  - (D) Diferentes objetos

- Surtido de **ítems** pequeños (*Assortment of small items*)

- (F) Pocos ítems de diferentes dimensiones
- (M) Muchos ítems de muchas dimensiones
- (R) Muchos ítems de relativamente pocas dimensiones
- (C) Muchos ítems todos iguales

Cada tipo de problema tiene por tanto una codificación de acuerdo a estas características. Como ejemplo ilustrativo, si se considera el clásico problema en el que se han de cortar barras grandes (objetos) disponibles en stock (todas ellas idénticas) en una de sus dimensiones con el fin de satisfacer las demandas a lo largo de un periodo de tiempo de barras más pequeñas (ítems), el problema se denotaría como 1/V/I/R.

Si bien en un principio la clasificación de Dyckhoff supuso un hito en la investigación de los problemas de *Corte y Empaquetado*, en tanto en cuanto relacionaba bajo la misma denominación dos áreas de la investigación que hasta el momento habían sido independientes (el *Corte* y el *Empaquetado*), con el paso de los años su uso no se ha extendido de manera generalizada entre la comunidad científica como podría esperarse inicialmente. Algunas de las deficiencias más significativas y que probablemente hayan sido la causa del escaso éxito de esta tipología son:

- En algunos casos el mismo problema se puede codificar de maneras diferentes. Un ejemplo de esto ocurre en el clásico problema de la mochila. En este caso las tres primeras características del problema son 1/B/O. Se debe realizar una selección entre los ítems (B), sólo hay un objeto grande (O). Sin embargo a la hora de decidir el surtido de ítems, el problema puede indistintamente referirse con F, M e incluso R.

- Dos problemas diferentes y que se resuelven de manera diferente se codifican de igual forma. (Gradisar et al. 2002) indican lo inconveniente de esta situación en el caso concreto de problemas de corte unidimensionales de tipo 1/V/D/R en los que en función de las particularidades del surtido de objetos el problema se resuelve mediante una técnica u otra.

A la vista de lo anterior (Wäscher et al. 2007) presentan una nueva caracterización tipológica de los problemas de *Corte y Empaquetado*. En este caso proponen cinco criterios para la definición de cada clase de problemas. Aprovechan

algunos de los utilizados en la anterior clasificación y revisan otros para salvar las inconsistencias de la clasificación anterior.

- Dimensionalidad: Número de mínimo de dimensiones necesarias para describir el problema (*Dimensionality)*

- Tipo de asignación entre ítems y objetos (*Kind of assignment*):
  - *Maximización del output*: Todos los objetos se utilizan y por tanto la selección se realiza sobre los ítems, lo que persigue maximizar el valor de la asignación (equivale a la B de Dyckhoff)

  - *Minimización del ínput*: A diferencia del caso anterior, se debe elegir entre un conjunto de objetos sobre los que acomodar todos los ítems de forma que se minimicen las pérdidas (equivale a la V de Dyckhoff)

- Surtido de objetos grandes (*Assortment of large objects*)
  - Un único gran objeto
    - todas sus dimensiones fijas
    - una o mas dimensiones variables
  - Muchos objetos (todas dimensiones fijas)
    - Idénticos
    - Surtido débilmente heterogéneo
    - Surtido fuertemente heterogéneo

- Surtido de ítems pequeños (*Assortment of small items*)
    - Todos idénticos
    - Surtido débilmente heterogéneo
    - Surtido fuertemente heterogéneo

- Forma de los ítems pequeños, sólo para problemas de más de dos dimensiones (*Shape of the small items*)
    - Forma regular
    - Forma irregular

Posiblemente la mayor diferencia respecto a la clasificación de (Dyckhoff 1990), que haga que esta nueva categorización se consolide en el tiempo, consiste en que mientras (Dyckhoff 1990) se limitaba a identificar algunos de los problemas más frecuentes con su codificación, (Wäscher et al. 2007) establecen toda una "jerarquía" en la categorización de problemas: puros, básicos, intermedios, refinados, estándar de primer nivel, estándar de segundo nivel, problemas especiales, problemas extensivos y variantes. El procedimiento que siguen es el siguiente:

Primero establecen la diferencia entre problemas puros y problemas extendidos (no contemplados por la clasificación). Los problemas extendidos incluyen aspectos adicionales más allá del propio problema de corte. Ejemplos de problemas extendidos son el de secuenciación de patrones estudiada por (Rinaldi, Franz 2007) o el del problema de minimización de patrones de (Umetani et al. 2003).

Después establecen las hipótesis que han de cumplir los problemas estándar: problemas de objetivo único, contexto determinista, periodo único de fabricación,…. En caso de no cumplirlas el problema pasa a ser una variante (tampoco contemplada en la clasificación). Ejemplos de problemas variantes son los problemas multiobjetivo como los estudiados por (Yang et al. 2006), los problemas con demandas variables o los problemas en línea o los problemas n-dimensionales.

A partir de dos características ("tipo de asignación" y "surtido de ítems pequeños") identifican cinco problemas básicos de *Corte y Empaquetado*:

- Problema de Empaquetado con Ítems Identicos (*Identical Item Parking Problem*)
- Problema de Emplazamiento (*Placement Problem*)
- Problema de la Mochila (*Knapsack Problem*)
- Problema de Dimensión Abierta (*Open Dimension Problem*)
- Problema de Corte (*Cutting Stock Problem*)
- Problema de la Caja de Embalaje (*Bin Packing Problem*)

	surtido de ítems / tipo de asignación	Idénticos	débilmente heterogéneos	fuertemente heterogéneos
todas las dimensiones fijas	maximización del output	Problema de Empaquetado de Ítems Idénticos (IIPP)	Problema de Emplazamiento (PP)	Problema de la Mochila (KP)
	minimizacón del input		Problema de Corte (CSP)	Problema de la Caja de Embalaje (BPP)
alguna dimensión variable		Problema de Dimensión Abierta (ODP)		

Tabla 3.1- Categorías de problemas básicos de Corte y Empaquetado

Sobre la base de los seis problemas básicos anteriores se va construyendo la clasificación. Al añadir las posibilidades de la característica “surtido de objetos grandes” aparecen los problemas intermedios. En el estadio de problemas intermedios se identifican hasta trece problemas.

Como ejemplos, en el caso de Problema de la Mochilla, al contemplar las diferentes posibilidades del surtido de objetos, nos encontramos con los siguientes problemas intermedios: el Problema de la Mochila Única (*Single Knapsack Problem*) donde sólo hay un único objeto grande (“mochila”), el Problema de Múltiples Mochilas Iguales (*Identical Knapsack Problem*) en el que hay más de una “mochila” pero todas ellas iguales y el Problema Heterogéneo de la Mochila (*Heterogeneous Knapsack Probles*) con varias “mochilas” diferentes entre sí.

Si se presta atención al tipo básico Problema de Corte, al añadir las posibilidades relativas al surtido de objetos, nos encontramos con el Problema de Corte con un Único Tamaño en Stock (*Single Stock Size Cutting Stock Problem*), donde todos los objetos son idénticos; el Problema de Corte con Múltiples Tamaños en Stock. (*Multiple Stock Size Cutting Stock Problem*), en el que el surtido de objetos es débilmente heterogéneo; y el Problema de Corte Residual (*Residual Cutting Stock Problem*).

Los problemas intermedios se identifican por medio de sus iniciales. Por ejemplo el Problema de Corte Residual será RCSP.

Al incorporar las características de "dimensionalidad" y "forma de los ítems pequeños" a los problemas intermedios, aparecen los problemas de tipo refinado. En el caso anterior, se hablaría de Problema unidimensional de Corte Residual o Problema bidimensional de Corte con Múltiples Tamaños.

A diferencia de la codificación de Dyckhoff, la identificación de los problemas se realiza mediante las iniciales del problema precedidas por su dimensión y forma del ítem, así pues en el caso anterior, el Problema unidimensional de Corte Residual se denotaría por *1dimensional RCSP* o en el caso del Problema bidimensional de Corte de piezas rectangulares con Múltiples Tamaños, su denominación sería: *2dimensional rectangular MSSCSP*.

Si a estos problemas refinados no se les añaden otro tipo de restricciones constituyen los problemas estándar de primer nivel. En caso de añadir nuevas restricciones o características al problema más allá de las utilizadas para la categorización, se habla de problemas estándar de segundo nivel.

A continuación se describen algunos de los problemas más frecuentes vinculados al *Corte y Empaquetado.*

- ***Empaquetado en cajas (Bin Packing)***

  Este problema tiene que ver con la determinación del número mínimo de cajas ("objetos") que se necesitan para empaquetar un conjunto de "ítems". Se han descrito varias versiones diferentes del problema y los "objetos" pueden modelizarse

bajo una o más dimensiones. El problema surge en diferentes aplicaciones como pueden ser la carga vehículos (*Vehicle Loading*), el *Scheduling* o la planificación de rutas (*Vehicle Routing*) (Coffman et al. 1997).

- ***Problema de la mochila (Knapsack Problem)***
  El problema consiste en un "objeto" (la mochila) con una capacidad fija y una serie de "ítems", que tienen un tamaño (y por tanto consumen parte de esa capacidad) a la vez que poseen una cierta utilidad. El objetivo que se persigue es la obtención de un subconjunto de "ítems" de forma que se consiga la máxima utilidad a la vez que se satisface la restricción de capacidad de la mochila. Este problema está estrechamente relacionado con el empaquetado en cajas (White 1992)

- ***Empaquetado Ortogonal (Orthogonal Packing- Strip Packing)***
  Este problema busca ubicar piezas rectangulares sobre otro rectángulo de ancho fijo y longitud variable. El empaquetado de rectángulos se realiza con sus lados orientados siempre paralelos a los ejes x e y (sólo se permiten rotaciones de 90 º). El problema se puede dar con altura limitada o ilimitada, en el caso de altura ilimitada se denomina *Strip Parking*. En (Álvarez-Valdés et al. 2008) se ha estudiado este problema recientemente

- ***Problema de anidación (Nesting – Marker Making Problem)***
  Este problema tiene que ver con el corte o marcado de piezas irregulares bidimensionales sobre una hoja o bobina de material de manera que se minimicen las pérdidas de material debidas al corte. Es un problema similar *al Empaquetado Ortogonal* salvo que en este caso, los ítems son de contorno irregular, un ejemplo de este tipo de problema se estudia en (Babu, Babu 2001). Una aplicación se encuentra en las industrias de ropa y calzado en las que en el proceso productivo para el conformado de la prenda, se manejan una gran cantidad de piezas de formas irregulares. En el caso concreto en el que el problema se circunscribe a la industria textil, el problema se ha denomina como *Marker Making Problem* precisamente por el marcado de las piezas que se deben realizar sobre las telas.

- ***Problema de Corte (Cutting Stock- Trim loss)***
  Este problema consiste en satisfacer las demandas de unas determinadas piezas que se deben obtener mediante el corte o partición de un objeto más grande

disponible en stock en unas determinadas cantidades. La asignación generalmente se realiza persiguiendo el objetivo de minimizar los recortes o piezas residuales generadas por el corte.

- ***Problema de Carga (Loading Problem)***
  El *Problema de Carga* se utiliza para el empaquetado de piezas regulares tridimensionales o de cajas, de modo que éstas deben ubicarse en el interior de un contenedor maximizando el volumen usado por la carga (o minimizando el espacio libre en el contenedor). Dependiendo de la industria de la que se trate pueden aparecer restricciones y nuevos objetivos que compliquen el problema (fragilidad de las cajas a la hora de ser apiladas, uniformidad en la repartición de pesos,…)

- ***Asignación de espacio - Asignación de la Capacidad (Space Allocation – Capacity Allocation)***
  Los problemas de asignación del espacio (o capacidad) están estrechamente relacionadas con los problemas de la mochila y del empaquetado de cajas. Distribuyen un espacio entre un conjunto de artículos (por ejemplo la asignación de puestos de trabajo en una oficina), mientras se respetan restricciones adicionales.

3.2 RESOLUCIÓN DE LOS PROBLEMAS DE OPTIMIZACIÓN COMBINATORIA

En esta sección se presentan las técnicas más utilizadas con carácter general para la resolución de problemas de optimización combinatoria, con la finalidad de que sirva de base para un posterior estudio de los procedimientos específicos en la resolución de problemas de Corte y Empaquetado, así como su particularización al MSSCSP.

3.2.1 Métodos Exactos

La resolución de problemas de optimización combinatoria (encontrar un procedimiento que permita encontrar la solución óptima) en la mayoría de los casos es una tarea difícil. La dificultad surge del hecho de que a diferencia de lo que ocurre en programación lineal, en el caso del problema combinatorio la región factible de soluciones no es con conjunto convexo. Así como en programación lineal se garantiza que cualquier

solución local es un óptimo global, en el caso de la programación lineal entera un problema puede poseer muchos óptimos locales, lo que obliga a tener que demostrar mediante otro tipo de argumentos, diferentes de los utilizados en programación convexa, que una determinada solución es mejor que el resto. Hay un número considerable de enfoques utilizados para resolver problemas de programación lineal entera, los más significativos serían: enfoques enumerativos; técnicas de descomposición lagrangiana; y algoritmos basados en planos de corte. Por último reseñar que una de las tendencias actuales consiste en combinar estos métodos mediante procedimientos híbridos con el fin de utilizar las ventajas de cada uno de ellos.

- **Técnicas enumerativas**

  Parece lógico que la forma más sencilla de resolver un problema de programación entera consiste en enumerar todas las soluciones factibles y elegir la mejor. Sin embargo la explosión de posibilidades como consecuencia de aumentar el tamaño del problema hace que la enumeración completa no sea viable. Así que una opción consiste en eliminar muchas de esas posibilidades argumentando aspectos de factibilidad y dominación. La técnica de Ramificación y Acotación (*Branch and Bound*) es la más utilizada de los enfoques de enumeración. Consiste en generar un árbol de soluciones en la que cada una de las ramas (proceso de enumeración por ramificación) que salen de un nodo conduce a una solución posterior a la del nodo. El método es capaz de detectar qué ramas no son óptimas (acotación) y las elimina del árbol (poda de la rama). (Linderoth, Savelsbergh 1998) describen diferentes estrategias utilizadas en la literatura dentro del marco de la ramificación y acotación.

- **Descomposición y relajación Lagrangiana**

  Frente a la clásica relajación lineal de los problemas de programación entera que consiste en “relajar” las restricciones de integridad de las variables, otro tipo de relajación es el que se basa en un enfoque lagrangiano. En este caso se parte de un conjunto de las restricciones "complicadas" del problema que se añadirán a la función objetivo con una serie de multiplicadores de Lagrange que se irán modificando de forma iterativa. Este enfoque se conoce como la relajación de Lagrange. Al eliminar las restricciones más complicadas del problema, el consiguiente subproblema es a menudo mucho más fácil de resolver. Los subproblemas deben resolverse repetidamente hasta que se alcancen los valores óptimos para los multiplicadores. Otra técnica relacionada con la relajación

lagrangiana es la descomposición lagrangiana. Este método consiste en aislar subconjuntos de restricciones de forma que se obtengan problemas sencillos de resolver para cada uno de estos subconjuntos. En cuanto a su aplicabilidad, comentar que todos los enfoques lagrangianos son fuertemente dependientes del problema y por tanto son de difícil aplicación o extrapolación general. Las publicaciones de (Nemhauser, Wolsey 1988) y (Martin 1998) proporcionan una excelente bibliografía sobre métodos lagrangianos.

- **Algoritmos basados en planos de corte**

  El método de los planos de corte fue introducido por (Gomory 1963) en su algoritmo del plano secante. La filosofía de estos algoritmos de tipo algebraico, consiste en modificar el conjunto convexo del espacio de soluciones de la relajación lineal del problema, de forma que todos sus puntos extremos lleguen a ser enteros. Pese a los cambios en los límites del espacio de soluciones, éstos deberán seguir proporcionando conjuntos convexos. En un primer estadio se resuelve la relajación lineal del problema y posteriormente se resuelve el problema subsiguiente de encontrar un plano (desigualdad lineal) que corte la parte fraccional de la solución lineal a la vez que se asegura que todos los puntos enteros factibles la cumplen. Un plano de corte será una restricción válida deducida del conjunto de restricciones del problema.

  El algoritmo termina cuando: o bien se encuentra una solución entera al problema y por tanto éste se ha resuelto; o bien la relajación lineal del problema no es factible y por tanto tampoco lo es el problema entero; o bien las últimas generaciones de planos de cortes no han mejorado la función objetivo como para seguir con el proceso; o bien no se es capaz de definir ningún plano de corte.

  Un algoritmo híbrido relacionado con los planos de corte es el algoritmo que se denominado como Ramificación y Corte (*Branch and Cut*) en el que se incorpora el enfoque de los planos de corte dentro de un algoritmo de acotación. Los planos de corte se generan a lo largo de todo el árbol, no solamente en su parte más alta

- **Algoritmo de Generación de Columnas**

  Como se ha visto, una de las técnicas más utilizadas en los procedimientos para resolver problemas de optimización combinatoria consiste en examinar la estructura del problema y encontrar una relajación o descomposición del problema que sea más sencilla de resolver. A partir de ahí se intenta consolidar esta aproximación, ya sea

por la adición de restricciones o modificando los coeficientes en las restricciones del problema o de la función objetivo.

Una descomposición que ha tenido gran éxito en los últimos años es la descomposición de Dantzing Wolfe a menudo llamada “generación de columnas” o “*branch and price*”. Se basa en el hecho de que cualquier punto factible se puede representar como una combinación lineal de los puntos extremos de la región factible.

Para la mayoría de los problemas reales un conjunto de puntos factibles es demasiado grande para enumerarlo, así que se comienza con la generación de columnas de manera que sea suficiente como para garantizar una solución viable (al menos para la relajación lineal del problema). A partir de aquí se van identificando columnas adicionales que mejoren la solución por medio de un algoritmo que utiliza la información del problema dual asociado para generar nuevas columnas. Se vuelve a resolver el problema primal añadiendo la nueva columna y se repite el ciclo hasta que no existe ninguna columna más que mejore la solución del problema lineal. Conocido el óptimo lineal se aplica una técnica de ramificación que modifique la solución.

3.2.2 Heurísticas

En los últimos años el desarrollo de procedimientos que utilizan heurísticas para resolver los problemas de optimización ha sido enorme. Un método heurístico frente a uno exacto suministrará una solución al problema que se considerará buena aunque no necesariamente sea la óptima. A diferencia de los métodos exactos que proporcionan una solución óptima del problema.

Algunos de los motivos que pueden justificar esta incorporación de las técnicas heurísticas a la resolución de problemas de optimización pueden ser: su flexibilidad a la hora de abordar aspectos de complicada modelización; su bajo coste computacional para implementarlas; la posibilidad de dar soluciones factibles en problemas en los que se desconoce un método exacto; que es capaz de suministrar buenas soluciones iniciales al problema que sirvan como entrada al desarrollo de otros procedimientos. Los procedimientos heurísticos exigen de una especialización elevada a la hora de establecer las especificaciones concretas de un problema. Esto los hace altamente dependientes del

problema concreto al que se están aplicando y dificulta poder establecer una clasificación completa. Una forma de agrupar las técnicas heurísticas podría realizarse utilizando las siguientes categorías:

- Métodos de descomposición: el problema se va descomponiendo en problemas más sencillos.

- Métodos Inductivos: A partir de los casos más simples del problema, se identifican buenas técnicas y por inducción se generalizan al problema completo.

- Métodos de Reducción: Añaden nuevas restricciones al problema de forma que limitan el espacio de soluciones. Estas restricciones se formulan a partir de las propiedades comunes identificadas sobre conjunto de buenas soluciones al problema.

- Métodos Constructivos: consisten en ir añadiendo elementos en cada una de las iteraciones hasta llegar a la solución. En cada iteración se elige el elemento que mejor rendimiento ha obtenido.

- Métodos de Búsqueda Local: Exploran el entorno de una solución original mediante operaciones llamadas movimientos. Buscan las soluciones al problema dentro del entorno de la solución inicial y las evalúan. La mejor pasa a ser la nueva solución. Mientras la solución pueda mejorarse el algoritmo continua.

La combinación de los métodos heurísticos de búsqueda local y los constructivos constituyen la base inicial sobre la que se desarrollaron algunos métodos metaheurísticos.

3.2.3 Metaheurísticas

Las Metaheurísticas están diseñadas para hacer frente a problemas complejos de optimización, donde otros métodos de optimización no han podido ser eficaces o eficientes. Estas técnicas han llegado a ser reconocidas como uno de los mejores enfoques para resolver muchos problemas complejos como es el caso en la mayoría de

los problemas reales de tipo combinatorio. La principal ventaja de las metaheurísticas radica tanto en su eficacia como en su aplicabilidad general. Originalmente, se desarrollaban complejas heurísticas muy específicas para resolver un determinado problema de optimización combinatoria, y esta especialización no siempre permitía su extrapolación a otro tipo diferente de problemas. Sin embargo, al ser las metaherurísticas estrategias de resolución de carácter general, el principal desafío en su uso, ha sido la adaptación concreta de la metaheurística para un determinado tipo de problema. Lo que normalmente requiere mucho menos trabajo que el desarrollo de una heurística específica para una aplicación específica. Para más información, (Glover, Kochenberger 2003) ofrecen una buena introducción y referencia general a muchas de los más populares metaheurísticas.

La idoneidad de utilizar metaheurísticas frente a otros métodos de optimización radica en el hecho de que éstas son capaces de encontrar buenas soluciones en problemas que contienen muchos óptimos locales y que no poseen una estructura capaz de orientar la búsqueda del óptimo global. El procedimiento que utilizan las metaheurísticas comienza por obtener una solución inicial (o un conjunto inicial de soluciones) y, a continuación, iniciar una búsqueda de soluciones mejores guiada por ciertos principios.

La estructura de búsqueda tiene muchos elementos comunes para numerosas metaheurísticas. En cada etapa del algoritmo de búsqueda, se parte de una solución (o un conjunto de soluciones), lo que representa el estado actual del algoritmo. En la etapa siguiente se evalúa un nuevo candidato (o conjunto de candidatos) dentro del entorno de la solución (o conjunto de soluciones) de la etapa anterior. Esta evaluación consiste en calcular o estimar el rendimiento del candidato o candidatos y compararlo con el rendimiento de la etapa previa, e incluso a veces entre sí. Basándose en esta evaluación, el candidato/s puede ser aceptado, entra a formar parte de la solución (o conjunto de soluciones) de esa etapa, o rechazada, en cuyo caso la solución (o conjunto) se mantiene. El proceso se repite hasta que se cumple un determinado criterio de fin.

Frente a las heurísticas, las metaheurísticas especifican estrategias generales de orientación de la búsqueda sin concretar todos los detalles relativos a la búsqueda. Por ejemplo, la estrategia de la búsqueda tabú es utilizar una lista de soluciones llamada lista

tabú, que garantiza que la búsqueda no adopte soluciones recientes y por tanto se quede atrapada en óptimos locales. En el caso de los algoritmos genéticos, se contemplan como conjunto de soluciones posibles en una etapa las soluciones que se pueden obtener mediante la combinación de las soluciones actuales a través de determinados operadores. En el recocido simulado, se determina un criterio para decidir la aceptación o el rechazo de soluciones que le permita escapar de óptimos locales.

Por tanto cada metaheurística establece ciertos elementos pero a la vez deja libre otros de forma que se adapten a la aplicación particular. Si bien esto dota a la técnica de gran flexibilidad, hace necesario el usuario debe especifique ciertas concreciones, lo que en determinados casos puede ser complicado. A continuación se describen algunas de las metaheurísticas más utilizadas en Optimización Combinatoria:

- **Los Métodos Multiarranque (*Multistart*)**

  Básicamente consisten en aplicar de forma reiterada un procedimiento de búsqueda a partir de diferentes soluciones iniciales construidas. Se distinguen dos fases en cada iteración. Primero se construye una solución para después mejorarla mediante una heurística de búsqueda. Numerosas aplicaciones combinatorias se pueden en encontrar (Martí 2000)

- **El Recocido Simulado (*Simulated Annealing*)**

  Esta fue una de las primeras metaheurísticas (Kirkpatrick et al. 1983). Su nombre debe su origen al proceso de recocido de los materiales, si bien en este caso este enfoque permitirá especificar un método para decidir si una solución debe ser aceptada. El recocido simulado es una búsqueda solución a solución, es decir, en cada etapa se evalúa un único candidato. El procedimiento de aceptación y rechazo se realiza de la siguiente forma. Se valúa el candidato, si éste es mejor que el anterior, se acepta y lo sustituye. Si el candidato es peor en lugar de directamente rechazarlo, puede ser aceptado con una probabilidad que depende de los rendimientos de las soluciones y de un parámetro que se denomina temperatura y que va descendiendo de acuerdo con un plan de “enfriamiento” preestablecido a medida que se van realizando las iteraciones del algoritmo.

- **La Búsqueda Tabú (*Tabu Search*)**

  Al igual que en el caso del recocido simulado, se trata de una técnica solución a solución. En cada iteración del algoritmo hay una lista de soluciones que se han visitado en las iteraciones anteriores y por tanto son tabú. El algoritmo busca en el entorno de las soluciones no tabú y elige la mejor (Glover 1989 y 1990).

- **Algoritmos GRASP** (**Greedy Randomized Adaptive Search Procedure**)

  Es otra metaheurística muy utilizada que consiste en iniciar la búsqueda desde diferentes puntos de arranque forma que ésta sea más global. En su versión más básica cada iteración consta de dos partes: una primera en la que se busca una solución buena aunque no sea la óptima local y una segunda fase en la que dentro del entorno de esa solución se busca su óptimo local. El procedimiento se repite hasta que se alcanza un criterio de fin (Feo, Resende 1989 y 1995).

- **Métodos Evolutivos**

  Se basan en conjuntos o poblaciones de soluciones. Así pues en cada iteración no se tiene una única solución como en las metaheurísticas anteriores sino que se tiene un conjunto de éstas.

  - Los **Algoritmos Genéticos (*Genetic Algorithms*)** constituyen una de las metaheurísticas que evalúa a toda una población o conjunto de individuos en cada iteración. Forman parte de los algoritmos que se denominan evolutivos y que se basan en la idea de la selección natural de forma que el conjunto de soluciones a lo largo de las iteraciones va evolucionando de acuerdo con unos operadores genéticos: recombinación, clonación y mutación (Goldberg 1989). Una de las ventajas de los algoritmos evolutivos es que permiten explorar el espacio de soluciones de forma rápida e "inteligente".

  - La **Búsqueda Dispersa (*Scatter Search*)** es otro tipo de metaheurística evolutiva. Al igual que en los algoritmos genéticos se parte de un conjunto de soluciones pero a diferencia de éstos en lugar de partir de poblaciones con un elevado número de individuos, en la búsqueda dispersa se suele trabajar con poblaciones de 10 individuos. Éstos se recombinan, pero en este caso la recombinación se realiza utilizando combinaciones lineales.

Desde un punto de vista espacial esta combinación lineal del conjunto de referencia puede entenderse como una forma de generar caminos lo que lleve a una visión más amplia que es la que se introduce en el Re-encadenamiento de trayectorias (*Path relinking*) (Glover 1999), (Glover et al. 2000) y (Resende et al. 2009)

- Los **Algoritmos Meméticos (*Memetic Algoritms*)**
  Son técnicas de optimización que combinan aspectos de otras metaheurísticas como son, el uso de poblaciones en los algoritmos evolutivos y la mejora local utilizada en Búsqueda Tabú o Recocido Simulado (Moscato, Cotta 2003)

Hay otras técnicas metaheurísticas importantes (la Optimización con Colonia de Hormigas; la Optimización por Cúmulo de Partículas; o Redes Neuronales) que en este apartado no se han considerado por no encontrarse entre las más frecuentes utilizadas para los problemas de optimización combinatoria. Éstas se tratarán en el capítulo 4 donde se desarrolla una metodología de resolución basada en la aplicación de técnicas metaheurísticas para el 1dimensional MSSCSP.

3.3 RESOLUCIÓN DE LOS PROBLEMAS DE CORTE Y EMPAQUETADO

Con el fin de dar una visión general sobre las metodologías y procedimientos de resolución propuestas en la literatura para resolver problemas de Corte y Empaquetado, en este apartado se hará una revisión de las principales técnicas desarrolladas para cada uno de los problemas básicos del *Corte y Empaquetado*. Se excluye el tipo básico CSP ya que éste será objeto de estudio en mayor profundidad en el capítulo siguiente.

3.3.1 Problema de Empaquetado con Ítems Idénticos (*IIPP*)

Este tipo básico de problema posee las características de tener un único objeto grande y un conjunto de ítems idénticos que deben disponerse de forma que se maximice el número de ítems que “caben” en el objeto. Tres particularizaciones relativas a este tipo se han abordado en la literatura. Uno es el clásico Problema del Pallet (*Manufacturer´s Pallet Loading Problem*) o MPLP en el que se debe cargar un pallet con el máximo

número de cajas iguales, si se asume que las cajas se apilan en capas iguales se trata de un problema *2dimensional rectangular IIPP*. En el caso de que los ítems fueran cilíndricos (de base circular) se ha estudiado en la literatura bajo el nombre del *Cylinder Packing Problem,* se trata de un *2dimensional circular IIPP*. La tercera aplicación práctica que aparece en la literatura se nombraría como *3dimensional rectangular IIPP* y lo hace bajo el nombre del Problema del Contenedor con Cajas Iguales (*Single Box Container Parking Problem*). En este caso se debe llenar un contenedor completo intentado que el volumen libre sea el mínimo posible.

Los metodologías de resolución propuestas en la literatura para la resolución de problemas del tipo IIPP han sido diversas. En los primeros años 80 se aplicaron técnicas heurísticas como son: el algoritmo de (Steudel 1979), que de forma recursiva busca maximizar el uso del perímetro del objeto; o los algoritmos llamados de cuatro y cinco boques desarrollado respectivamente por (Smith, De Cani 1980) y (Bischoff, Dowsland 1982), en el que cuatro bloques se ubican en las esquinas del objeto buscando la mejor orientación de éstos y en el caso del quinto, este se coloca en la zona central del objeto. Otras heurísticas son al algoritmo G4 de (Scheithauer, Terno 1997), que ha dado muy buenos resultados para determinadas instancias, y más reciente la de (Lins et al. 2003) basada en cortes en L que definen el concepto de la estructura. En cuanto a la aplicación de métodos exactos, (De Cani 1979) aplica por primera vez un algoritmo de Ramificación y Acotación y (Dowsland 1985 y 1987) y (Lins et al. 2002) desarrollan algoritmos exactos basados en la correspondencia entre la teoría de grafos y el PLP. Más recientemente (Cui 2006) propone un algoritmo de Ramificación y Acotación para la generación de patrones óptimos. Las primeras metaheurísticas que se aplicaron sobre este tipo de problemas se basaban en el recocido simulado y en la búsqueda tabú (Dowsland, Dowsland 1992) y (Dowsland 1996). Sobre recocido simulado aplicado al problema del cilindro destacan las publicaciones de (Correia, et al. 2001). Posteriormente se fueron desarrollando otras basadas en algoritmos evolutivos, o en oscilación estratégica (Amaral, Wright 2001). Recientemente (Álvarez-Valdés et al. 2007) desarrollan un algoritmo de búsqueda tabú que incluye procedimientos de intensificación y diversificación basados en memoria a largo plazo.

Respecto al interés de investigación respecto al problema MPLP (Wäscher et al. 2007) hacen notar el descenso considerable en la última década del número de

publicaciones relativas a este tema. Lo que parece indicar que el problema se ha resuelto satisfactoriamente y las investigaciones en problemas IIPP tienden hacia una versión más complicada de este problema como es el caso de los problemas de emplazamiento tipo SLOPP

3.3.2 Problema de Emplazamiento (*PP*)

Este tipo básico de problema posee las características de tener un surtido de ítems débilmente heterogéneo en el que se debe maximizar la asignación de éstos sobre el/los objetos. Esta categoría admite los tres tipos de surtido de objetos lo que lleva a tres problemas intermedios Problema de Colocación en un objeto (SLOPP), Problema de Colocación en Múltiples Objetos Idénticos (MILOPP), Problema de Colocación en Múltiples Objetos Heterogéneos (MHLOPP). De entre los tres tipos el que más interés despierta es el SLOPP en el que a único objeto se deben asignar los máximos ítems posibles. La forma más común es 2dimensional regular SLOPP en la que sobre una plancha rectangular grande se deben cortar piezas rectangulares. Originalmente descrito por (Haims, Freeman 1970) bajo el nombre *Template Layout Problem* y se han encontrado aplicaciones con ítems de base rectangular como circular (Kopardekar, Mital 1991) Su aplicación en una dimensión son los problemas denominados *Bounded Knapsack Problem y Unbounded Knapsack Problems* estudiados por (Andonov et al. 2000) o (Zukerman et al. 2001). Un ejemplo de 3dimensional rectangular SLOPP es el Problema de Carga de un Único Contenedor estudiado en (Bischoff et al. 1995) y (Davies, Bischoff 1999)

Al igual que ocurría con el problema IIPP, en el caso de los problemas de PP a lo largo de los años se han ido aplicando multitud de técnicas para su resolución: sólo entre los años 1995 y 2005 (Oliveira, Wäscher 2007) identifican más de 60 publicaciones internacionales referidas a los problemas de emplazamiento. Entre los enfoques exactos se encuentra la aplicación de algoritmos de Ramificación y Corte para el problema de dos dimensiones (Hifi 1997) que incorpora técnicas de programación dinámica que permiten determinar cotas superiores e inferiores para realizar la poda de las ramas. Posteriormente (Cung et al. 2000) y (G et al. 2003) incorporan al algoritmo de Hifi una heurística utilizada por (Fayard et al. 1998) que les permite aumentar inicialmente la cota

inferior y así limitar el espacio de búsqueda, a la vez que tratan de mejorar el límite superior en cada nodo interno del árbol. El uso de metaheurísticas también se ha utilizado como metodología para la resolución de problemas de emplazamiento, especialmente con técnicas basadas en procedimientos híbridos: (Daza et al. 1995) desarrollan técnicas híbridas a partir de algoritmos genéticos; (Alvarez-Valdés et al. 2002) aplican la búsqueda tabú a problemas bidimensionales de gran escala; (Hifi, M´Hallall 2004) desarrollan soluciones basadas en el recocido simulado para problemas con ítems circulares; y (Mack et al. 2004) emplean un algoritmo híbrido basado en búsqueda local para el problema 3 dimensiones.

3.3.3 Problema de la Mochila (*KP*)

Los problemas intermedios dentro del KP son de características análogas a los de Emplazamiento pero con la salvedad que en el caso de la mochila el surtido de ítems es fuertemente heterogéneo. Así pues se dan las siguientes posibilidades: Problema de la Mochila Simple (SKP) que corresponde al clásico Problema de la Mochila en el que hay que introducir objetos con un peso y valor dentro de una mochila de forma que no se sobrepase su capacidad y se maximice el valor del contenido de ésta. Particularizaciones de este problema son el *Subset-Sum Problem, que* es un problema de tipo 0-1 en el que el peso del ítem es exactamente igual que su valor, y el problema de la Mochila Multirestricción (Drexl, 1988), en el que se incluyen otras restricciones aparte de la capacidad.

Distintos algoritmos de ramaficación y acotación se han propuesto desde hace más de tres décadas para resolver el problema de la mochila, entre otros se pueden destacar a (Hung, Fisk 1979) o (Chistofides et al. 1979). El primero se ajusta bien a los problemas en los que los objetos son idénticos, mientras que el segundo se adapta mejor en el caso en el que haya múltiples tipos de mochilas. La resolución del problema de la mochila utiliza cotas superiores e inferiores. Numerosas son las cotas superiores que se han presentado en la literatura y se basan en variedad de técnicas que incluyen: linealización, relajación lagrangiana, obtención de planos superiores y técnicas de reformulación. Como ejemplo, (Martello, Toth 1981 y 1990) proponen algoritmos del tipo "*bound and bound*" en el que en cada uno de los nodos del árbol, no sólo obtienen una cota superior sino

también una inferior, y su aplicación es muy útil en problemas en los que es difícil verificar la factibilidad de la cota superior

3.3.4 Problema de Dimensión Abierta (*ODP*)

El problema de dimensión abierta corresponde a la clase de problemas de minimización en los que todos los ítems se deben acomodar sobre un/os objeto. La particularidad del problema consiste en que al menos una de las dimensiones del objeto es variable, por tanto el problema nunca será de tipo 1dimensional. Las aplicaciones referenciadas sobre este problema usualmente describen el problema para un único objeto. Problemas de esta clase con una dimensión variable son el *Orthogonal Packing y el Strip Packing* descritos más arriba y que se denotarían como 2dimensional rectangular ODP. Otro ejemplo es el problema del *Marker Making Problem* o *Nesting Problem* también descrito al inicio del capitulo y que se da en las industrias del calzado y ropa. En este caso el problema es del tipo 2dimensional irregular ODP. Problemas con objetos que posean dos dimensiones variables los tratan (Hifi, Ouafi 1998).

De todas las categorías de problemas básicos, el ODP es la que en los últimos 15 años mayor número de publicaciones internacionales ha arrojado. A lo largo de los años se han ido aplicando diversas metodologías de resolución para estos problemas. A continuación y de manera muy somera se describen las principales para los problemas regular ODP e irregular ODP.

En el 2dimensional regular ODP, (Coffman et al. 1980) aplicaron con éxito heurísticas basadas en clásicos algoritmos de aproximación utilizados para otros problemas de corte. Tanto el *Next Fit Decreasing Height* (NFDH) como el *First Fit Decrasing Height* (FFDH) agrupan los ítems por niveles. Otro de los enfoques clásicos, pero que no agrupa los ítems en niveles, es el algoritmo BL (*Bottom Left*) que definieron (Barker et al. 1980). (Kenyon, Rémila 2000) proponen un esquema completo de aproximación polinomial asintótica basado en la relajación lineal del problema. También se han aplicado técnicas basadas en metaheurísticas: (Dowsland 1993) propone un algoritmo basado en el recocido simulado; y (Jakobs 1996) y posteriormente (Gomez, de la Fuente 2000) desarrollan algoritmos genéticos mediante la codificación de los patrones de

empaquetado que permite la aplicación eficiente de las operaciones "genéticas" de recombinación y mutación. En cuanto al empleo de métodos exactos de resolución, (Martello et al. 2000) proponen un enfoque de enumeración en el que los ítems inicialmente se ordenan en altura decreciente y un procedimiento de reducción determina el óptimo. (Hifi 1998) desarrolla algoritmos exactos para la resolución del *Strip Problem* basados en procedimientos de ramificación y acotación, que combinados con algoritmos heurísticos limitan la longitud inicial del objeto. Recientemente (Álvarez-Valdés et al. 2009) también han desarrollado un nuevo algoritmo de ramificación y acotación basado en (Martello et al. 2003) para el *Strip Problem*.

Existen numerosos métodos de resolución para los irregular ODP (*Nesting Problems*). Revisiones bibliográficas al respecto se pueden encontrar en (Dowsland, Dowsland 1995). Las Metaheurísticas son una de las herramientas más populares para la resolución de este tipo de problemas (Bennell, Dowsland 1999 y 2001). Los enfoques más interesantes dividen el problema en tres subproblemas. Por un lado el subproblema de la representación geométrica de ítems irregulares, para lo que se han técnicas basadas en el concepto de anidación (*nesting*), donde los ítems se van "anidando" en bloques de forma que se representan contenidas en un polígono regular (rectángulos (Freeman, Shapira, 1975), hexágonos (Dori, Ben-Bassat, 1983 y 1984),…). Por otro lado, del subproblema de la ubicación de los ítems sobre el objeto mediante lo que se han dado en llamar métodos básicos, divididos entre los que permiten y los que no permiten superposiciones en el proceso de búsqueda. Los métodos básicos son algoritmos que generan aleatoriamente el ordenamiento inicial de las ordenes demandas y luego o bien utilizan heurísticas como la *first/best fit* para ir ubicando las sobre el objeto (Gomes, Oliveira 2002) o van moviendo los ítems por medio de rotaciones (Bennell, Dowsland 2001). Por último, el subproblema de escapar de posibles óptimos locales, en el que se utilizan metaheurísticas. (Gomes, Oliveira 2006) utilizan un algoritmo basado en recocido simulado con muy buenos resultados. Los algoritmos de tipo evolutivo también se han aplicado pero con un éxito limitado (Burke, Kendall, 1999)

3.3.5 Problema de la Caja de Embalaje (*BPP)*

Al contrario de lo que ocurre con le CSP, en el caso de los problemas tipo BPP el surtido de ítems es fuertemente heterogéneo y deben ser asignados de forma que se

minimice el número de objetos que se han de utilizar. Surgen tres problemas intermedios en función de la característica “surtido de objetos grandes”.

El primero es el Problema de la Caja de Embalaje de Tamaño Único (SBSBPP), en el caso en el que todas las cajas (objetos) sean iguales. En su formato unidimensional ha aparecido bajo los nombres de Problema de Carga de Vehículos (*Vehicle Loading Problem*) y Problema de Corte Binario (*Binary Cutting Stock Problem*). El 2dimensional SBSBPP se ha estudiado tanto con ítems de contorno rectangular (two-dimensional Finite Bin Packing Problem) como circular (Cylindrical Bin Packing), ver (Lodi et al. 2002).

En el caso en el que las cajas no sean iguales, el problema se denomina como Problema de la Caja de Embalaje con Múltiples Tamaños (MBSBPP). Su estudio en muchos casos (Kang, Park 2003) ha restringido los supuestos de: disponibilidad ilimitada de cajas; y que cada una de éstas cajas viene caracterizada por su coste y sus dimensiones.

Por último, en caso de que el tamaño de las “cajas” sea fuertemente heterogéneo, da lugar al Problema Residual de las Caja de Embalaje (RBPP) que es equivalente al RCSP. Una aplicación unidimensional se estudia en (Kos, Duhovnik 2002) en donde se aplica un algoritmo híbrido genético para resolver el problema de minimización del desperdicio.

Los Problemas BPP han sido ampliamente estudiados en los últimos años y al igual que en el resto de los problemas básicos de Corte y Empaquetado, se han referenciado para su resolución tanto métodos exactos como métodos heurísticos y metaheurísticos. A continuación se citan los más significativos para el 2dimensional SBSBPP. Algunos autores plantean enfoques híbridos utilizando de manera conjunta reglas heurísticas como el *First-Fit* o *el Next-Fit Decreasing* o el *Best-Fit Decreasing* (Frenk y Galambos, 1987) y (Berkey y Wang, 1987). (Lodi et al. 1999, 2004) proponen también un algoritmo que llaman Suelo-Techo (FC) basado en diferentes pautas heurísticas para la orientación del empaquetado (izquierda a derecha con la base en el suelo o de derecha a izquierda con la parte superior tocando el techo) y agrupándolo en diferentes niveles. En cuanto al uso de Metaheurísticas, (Lodi et al. 1999) desarrollan algoritmos en los que: partiendo de una heurística constructiva aplican la búsqueda tabú para escapar de óptimos locales. La principal característica de estos algoritmos consiste en que adoptan un esquema de búsqueda que permite aplicarlos a cualquier tipo de variante tanto del Problema de Cajas

de Embalaje bidimensional como tridimensional. En cuanto a los métodos exactos, (Martello et al. 2000) utilizan esquemas de ramificación que se basan en adaptaciones de algoritmos de ramificación y corte para resolver el 2 y 3dimensional SBSBPP. Un enfoque de tipo enumerativo es el que proponen (Martello, Vigo 1998) para la resolución exacta del problema bidimensional y que se basa en un esquema de ramificación en dos niveles a la vez que utiliza un procedimiento de reducción.

3.4 APLICACIONES DE LOS PROBLEMAS DE CORTE Y EMPAQUETADO EN LA INDUSTRIA

El gran abanico de posibilidades dentro de los problemas de Corte y Empaquetado hace que su aplicabilidad se de en numerosos sectores industriales. En esta sección se identifican los sectores más significativos a la vez que se relacionan con su correspondiente literatura académica.

3.4.1 Industria Papelera

El primer estudio sobre corte y el empaquetado se llevó a cabo en los años 50 e identificaba los problemas que surgían en la industria del papel. Esta primera investigación buscaba encontrar la manera óptima en que se podía cortar una bobina en pedazos más pequeños y utilizó técnicas de programación lineal (Eisemann, 1957). Ejemplos más reciente de investigaciones basadas en la industria del papel se presentan en (Johnson et al. 1997) donde se describe una heurística y un método exacto para resolver el problema del corte de bobinas de papel. (Harjunkoski et al. 1998) también presentan dos algoritmos de programación entera para el corte de bobinas en stock en bobinas comerciales que aplicaron a una paperera finlandesa. (Aboudi, Barcia 1998) además de utilizar métodos basados en programación lineal para el problema corte de papel, se ocupan del problema que surge cuando aparecen áreas defectuosas debidas al proceso de fabricación. Una zona defectuosa no se puede utilizar en el corte de bobinas. (Hahn 1968) ya estudia el problema de zonas defectuosas en las bobinas de papel y produce un algoritmo basado en óptimo (Gilmore, Gomory 1966) para eliminar las hojas que contienen múltiples zonas defectuosas. (Menon, Schrage 2002) proporcionan un enfoque basado en programación entera para resolver el problema de asignación del

orden en el caso de múltiples máquinas de corte en paralelo. También el caso de múltiples máquinas paralelas es el centro de (Giannelos, Georgiadis 2001) que utilizan técnicas exactas para su resolución.

3.4.2 Industria Maderera

La industria de corte de madera también ha sido objeto de investigación. (Morabito, García 1998) abordan un problema similar al de la industria del papel, salvo que en este caso el corte en pequeñas piezas rectangulares se realiza mediante de sierras circulares. Se compararon dos técnicas para resolver instancias reales de la industria maderera brasileña. La primera utiliza programación dinámica y la segunda es una simple extensión del algoritmo de (Gilmore, Gomory 1963). La producción en la industria del mueble es el tema de (Morabito, Arenales 2000). Los autores prestan especial atención al equilibrio entre la producción con poco desperdicio y la producción con diseños más sencillos que permitan cortes con menos complicaciones. El coste unitario del material se pondera mediante algoritmos que tienen en cuentas estos dos objetivos.

3.4.3 Industria del Vidrio

El corte de vidrio es un proceso en el que en primer lugar el vidrio debe ser marcado y, a continuación se corta sobre un eje plano. (Puchinger et al. 2004) desarrollan algoritmos evolutivos para resolver problemas reales de corte de vidrio. Ellos compararon diferentes heurísticas y metaheurísticas y sugieren que los algoritmos evolutivos dan buenos resultados bien en la práctica.

3.4.4 Industria Textil

La industria textil también es un buen banco de pruebas en el que aplicar investigaciones sobre el empaquetado automatizado. El corte de prendas de vestir a partir de bobinas de tela, ofrece muchas dificultades y matices. Estos problemas pueden incluir dibujos en las bobinas y, por tanto, limitaciones en la rotación de las piezas por las

orientaciones en las que se puede cortar la ropa (alineación de patrones,..). Además, a menudo se trata de formas muy irregulares y muchas veces el material presenta zonas defectuosas. Al principio del presente capítulo se ha descrito el *Marker Making Problem* que aparece en la industria de ropa. Aunque el problema de corte irregular tiene sus origen en (Adamowicz, Albano 1976), y probablemente por la dificultad exigida a los modelos (elevada irregularidad tanto de ítems como objetos) hasta los últimos años mucho interés en el problema textil. Algunos ejemplos recientes (Takahara et al. 2003) que proponen un enfoque basado en heurísticas de agrupamiento y metaheurísticas adaptativas. Otra referencia al problema se puede encontrar en (Babu, Babu 2001)

3.4.5 Industria de Pieles y Calzado

La industria de corte de piel también ha sido objeto de algunos de los últimos trabajos de corte y en embalaje. (Crispin et al. 2003) tratan el problema del diseño de cortes en la industria del calzado. El cuero posee una considerable variabilidad en su resistencia y en la calidad de diferentes áreas. Estos factores suelen restringir las posibles ubicaciones de las piezas. Por otro lado la variabilidad en los tipos de pies obligan a la supervisión directa de un operario bien durante todo el proceso o bien para identificar las zonas resistentes. Los autores utilizan algoritmos genéticos para generar los ángulos de posición de las piezas así como su dirección de anidación. Otras publicaciones que también estudian la planificación de la predicción en la industria de pieles se pueden encontrar en (Bounsaythip et al. 1995) y en (Heistermann, Lengauer 1995)

3.4.6 Industria del Metal

El corte metálico contiene un gran número de aplicaciones reales. Por ejemplo, los costes de material pueden variar considerablemente, desde el corte de hojas finas de aluminio hasta perfiles de metal que pueden tener espesores de varios centímetros. Por lo general, esto tiene un impacto en la forma de abordar el problema, así por ejemplo el caso del aluminio la importancia de generar un diseño óptimo es probablemente secundaria frente a la producción de una solución rápida que permita el ahorro de costes de producción y mantenga la producción en funcionamiento. Sin embargo, en el caso de

metales de gran espesor el coste por unidad de material es alto y por tanto la optimización del corte supone un ahorro significativo en costes. La casuística del corte metálico es enorme; corte unidimensional, corte bidimensional; piezas regulares rectangulares, piezas circulares; corte con perfiles o bobinas iguales; corte con perfiles diferentes y stock limitado,… Esto ha supuesto un número muy considerable de publicaciones sobre el tema, recientes estudios de problemas planteados en la industria de acero y metal se pueden encontrar en (Cui et al. 2009) y (Cui, Lu 2009) los primeros proponen un algoritmo *greedy* para el solucionar el *Strip problem* con ítems circulares para la embutición de tazas y los segundos desarrollan un algoritmo basado en técnicas recursivas y de programación lineal para el corte bidimensional de piezas metálicas para la industria de construcción de puentes metálicos.

CAPÍTULO 4

EL PROBLEMA DE CORTE UNIDIMENSIONAL (1D CSP)

El problema identificado en la empresa de perfiles metálicos forma parte del grupo de problemas que (Wäscher et al. 2007) denominan por las siglas **CSP**. En el presente capítulo se estudia más a fondo este tipo de problemas y se describen las principales formulaciones matemáticas que han aparecido en la literatura así como los principales métodos y estrategias de resolución.

Como se ha indicado en el capítulo anterior, el problema de corte (CSP) es uno de los seis tipos básicos de la clasificación de (Wäscher et al. 2007). Se trata de una clase de problemas en la que la característica "tipo de asignación" busca la minimización del *input*, de forma que la disponibilidad de objetos sea lo suficientemente grande como para permitir que todas las necesidades de ítems queden satisfechas. El objetivo consistirá en minimizar la cantidad de objetos necesarios para cubrir las necesidades de los ítems. En cuanto a la característica "surtido de ítems", los CSP poseen un conjunto de ítems débilmente heterogéneo, es decir, hay relativamente pocas cantidades de ítems diferentes. Al analizar la característica "surtido de objetos" aparecen tres posibles alternativas que dan lugar a los problemas de corte siguientes: el SSSCSP, el MSSCSP y el RCSP.

- **Problema de Corte con un Único Tamaño en Stock (SSSCSP)**.
  Este caso es la particularización del CSP para varios objetos todos iguales y en la literatura se han descrito problemas de este tipo para diferentes dimensiones. Así por ejemplo el 1dimensional SSSCSP es el clásico problema de Corte estudiado por (Gilmore, Gomory 1963), en el que el corte se realiza en la dimensión del eje longitud de las barras (objetos) para obtener barras más pequeñas (ítems). El 2dimensional SSSCSP es análogo al anterior pero con en este caso se cortan piezas bidimensionales (rectangulares) sobre planchas (tablas) más grandes en las que sus dimensiones (ancho y largo) son valores dados, muchos son los autores que han estudiado este problema. Ejemplos del 3dimensional SSSCSP son el Problema de Carga Múltiple de Paletas/Contenedores (*Multi Pallet/Container Loading Problem*) estudiados en (Steudel 1984) o (Lins et al. 2002), el objetivo radica en cargar unas cantidades determinadas de productos sobre paletas (o en contenedores) de forma que se reduzca al mínimo el número de paletas necesario. El objetivo teórico de buscar la mejor utilización del espacio está normalmente supeditado a una lista de cuestiones prácticas de tipo tecnológico, de distribución del peso sobre la paleta, de estabilidad,…

- **Problema de Corte con Múltiples Tamaños en Stock (MSSCSP)**.
  Este caso es la particularización del CSP para objetos de relativamente pocos tipos en stock. Se trata de la ampliación lógica del problema anterior a más de un único tipo de objetos. En el caso del 1dimensional MSSCSP se pueden encontrar aplicaciones reales en (Holthaus 2002) y (Poldi, Arenales 2009). Para el caso 2dimensional MSSCSP se parte de un conjunto de piezas rectangulares disponibles en inventario de diferentes tipos. Las piezas son de dimensiones conocidas y hay que cortarlas en pedazos más pequeños de manera que den lugar a piezas también rectangulares con el fin de satisfacer una determinada demanda. El objetivo será reducir al mínimo la superficie total requerida para el corte. Aplicaciones prácticas se dan en la industria del mueble con el corte de madera, o en la industria del cartón a la hora de la fabricación de cajas (Álvarez-Valdés et al. 2002).

- **Problema Residual de Corte (RCSP)**.
  En este caso el tipo de objetos es fuertemente heterogéneo y por tanto hay mucha variedad de piezas almacenadas en stock. Suele aparecer en la industria como

consecuencia del mismo proceso de corte tanto en una como en dos dimensiones. Cuando los restos de material cortado son suficientemente grandes como para desecharse y por tanto se vuelve a inventariar, de ahí su denominación de problema residual (Gradisar et al. 2002).

Así pues, podemos decir que **el problema identificado en la empresa de corte** de perfiles será o bien del tipo 1 dimensional MSSCSP o bien del tipo 1 dimensional RCSP. Unidimensional debido a que el corte se realiza en una sola dimensión del perfil. Si en almacén no encontramos muchos largos diferentes (largos estándar) podríamos decir que se trata del problema MSSCSP, ahora bien el número de perfiles procedentes de puntas de fabricación es elevado hablaríamos de un problema de tipo RCSP. El método de resolución buscará la eficiencia en el corte y minimizar así, tanto los restos como las puntas de producción con lo que **el problema de corte de perfiles estructurales será del tipo MSSCSP**.

4.1 EL PROBLEMA DE CORTE UNIDIMESIONAL (1D CSP) FORMULACIONES MATEMÁTICAS

El problema de corte unidimensional tradicionalmente ha aparecido en la literatura bajo el nombre del Problema de Corte Estándar Unidimensional (1D CSP). Lo definen los siguientes datos: $(m, L, l = (l_1, ..., l_m), d = (d_1, ..., d_m))$, L representa la longitud del perfil[1] en stock, m es el número de tipos de barras[1] que hay que producir $(i=1,...m)$, l_i es la longitud de cada barra y d_i su demanda. Un plan de corte consistirá en producir las barras demandados a partir de los perfiles disponibles en stock. El objetivo tradicional que se persigue es minimizar el número de perfiles utilizados para satisfacer las demandas, o lo que es lo mismo, minimizar la pérdida de material de los restos. Algunos autores como (Umetani et al. 2003) han estudiado otros objetivos que aparecen en los procesos de corte reales como son la minimización por cambios de partida o la posibilidad de apilar los perfiles en la operación de corte. En (Wäscher et al. 2007) se describen estos problemas como *extensiones* del problema.

[1] A partir de este punto y por ajustarse con su denominación en el sector metalmecánico, los objetos en stock se denominarán como perfiles y los ítems demandados como barras

En (Valério de Carvalho 2002) se realiza una revisión de las diferentes formulaciones de programación lineal del problema de corte unidimensional. Identifica el modelo de (Kantorovich 1939) para la minimización del número de bobinas cortadas. El modelo clásico es el que se presenta en (Gilmore, Gomory 1961 y 1963). Cada una de las variables de decisión corresponde a un conjunto de operaciones de corte que se realizan sobre el perfil para la obtención de las barras (patrón de corte).

En (Dyckhoff 1981) y en (Statdler 1985) se introducen formulaciones de modelos de corte único (*one-cut models*) en los que cada variable de decisión corresponde a una única operación de corte realizada sobre un perfil de manera que cada corte produce al menos una pieza de las demandadas. En este modelo un patrón de corte se expresa como una secuencia cortes únicos. Una de las características de este modelo es el hecho de que las piezas residuales (restos) son reutilizables para cortar más barras. En (Dyckhoff, 1981) se muestra cómo el modelo de (Gilmore, Gomory 1961,1963) y el modelo de corte único poseen soluciones enteras equivalentes y que además el número de variables del modelo de corte único es de tipo pseudo polimonial, es decir no crece de manera explosiva como ocurre en el modelo basado en patrones de corte.

Otra de las formulaciones que han aparecido en la literatura, es aquella que se basa en la asignación de posiciones indexadas como son los modelos basados en la teoría de grafos. Esta formulación consiste en indexar las variables que corresponden a las barras en función de la posición física que ocupan dentro del perfil que se va a cortar. Este tipo de formulación fue usada primero en (Beasley 1985) y posteriormente en (Valerio de Carvalho 1998).

Así pues, las formulaciones matemáticas basadas en programación lineal que hasta el momento se han propuesto en la literatura para modelizar el *1dimensional SSSCSP*, pueden dividirse en las siguientes categorías: modelo de asignación (Kantorovich, 1939); modelo basado en patrones de corte (Gilmore, Gomory, 1961); modelo de corte único (Dyckhoff, 1981); y modelo basado en grafos de flujo (Valério de Carvalho, 1998).

4.1.1 Modelo de asignación

En (Kantorovich 1939) aparece la primera propuesta de formulación basada en variables de asignación utilizando variables binarias para relacionar las barras con el material disponible en stock. El modelo se formula de la siguiente manera:

$$z = \min \sum_{j=1}^{n} y_j \tag{4.1}$$

s. a

$$\sum_{j=1}^{n} x_{ij} \geq d_i \ , \quad i = 1,...,m, \tag{4.2}$$

$$\sum_{i=1}^{m} l_i x_{ij} \leq L y_i \ , \quad \mathrm{j} = 1,...,n, \tag{4.3}$$

$$y_i \in \{0,1\}, \quad j = 1,...,n, \tag{4.4}$$

$$x_{ij} \geq 0 \, y \text{ entero}, \, i = 1,...,m \; j = 1,...,n. \tag{4.5}$$

Se definen y_i como variables binarias que representan la elección del perfil j, x_{ij} el número de barras de longitud l_i asignados al perfil j. Donde (4.2) y (4.3) son las condiciones de demanda y de la "mochila" (la suma de las longitudes de las barras cortadas en un perfil no puede sobrepasar la longitud de éste). El modelo (4.1)-(4.5)crece muy rápidamente en tamaño en cuanto se aumentan los valores de n y m además presenta algunas deficiencias como son: una cota inferior muy "pobre"; el intercambio de barras entre dos perfiles conduce a soluciones diferentes en términos de los valores de las variables pero que en la práctica son exactamente iguales. Estos motivos hacen que la reformulación basada en la aplicación de métodos de descomposición se considere una alternativa.

4.1.2 Modelo basado en patrones de corte

Gilmore y Gomory (1961) proponen un modelo que en lugar de hacer una asignación directa entre las barras y los perfiles, utilizan el concepto de *patrón de corte*. Este modelo se obtiene mediante la aplicación de la descomposición de Dantzing-Wolfe. Un patrón de corte es una combinación factible de barras para cada perfil en stock, de forma que la suma de las longitudes de las barras producidas en un perfil no sobrepase la longitud

total del perfil. Un patrón j viene definido por el vector columna $a^j=(a_{1j},...,a_{mj})\in\mathbb{Z}_+^m$, $j=1,...,n$. Se dice que un patrón es factible si cumple la siguiente restricción (*condición de la mochila*)

$$\sum_{i=1}^{m} l_i a_{ij} \leq L \tag{4.6}$$

En (Nitsche et al. 1999) se utiliza el concepto de patrón "adecuado" (*proper*) cuando la cantidad de una barra cortada por un patrón no supera la demanda de esa barra, ya que para demandas pequeñas la búsqueda de soluciones en la relajación lineal del problema se reduce.

$$a_{ij} \leq d_i, \quad i=1,...,m \qquad j=1,...,n \tag{4.7}$$

El modelo basado en patrones de corte se formula de la siguiente manera

$$z_{PE} = \min \sum_{j=1}^{n} c_j x_j \tag{4.8}$$

s. a

$$\sum_{j=1}^{n} a_{ij} x_j \geq d_i, \quad i=1,...,m \tag{4.9}$$

$$\sum_{i=1}^{m} l_i a_{ij} + c_j = L \tag{4.10}$$

$$x_j \in \mathbb{Z}_+, \quad j=1,...,n \tag{4.11}$$

Donde x_j, $j = 1,\ldots, n$ son las frecuencias de cada patrón (el número de veces que se va a utilizar) y c_j el desperdicio asociado a cada patrón.

Frente al modelo de asignación, la reformulación del problema basada en patrones de corte es mejor en tanto en cuanto: no hay simetría en el espacio de soluciones (dos soluciones matemáticas no corresponden a una misma solución real); en la mayoría de los casos el "hueco" (*integrality gap*) entre el óptimo de la relajación lineal y el óptimo entero es menor que la unidad, algunos son mayores que 1 pero nunca mayores que 2 (Marcotte, 1986); su cota es bastante más fuerte. El gran inconveniente de este modelo tiene que ver con su tamaño (en la práctica ni se plantea una enumeración completa de los patrones de corte, éstos se generan de forma dinámica)

4.1.3 Modelo de corte único

En (Dyckhoff 1981) y en (Stadler 1988) se proponen modelos que trabajan con la aplicación de cortes únicos para la producción de los ítems demandados. Si el corte se realiza sobre uno de los perfiles en stock, aparece una pieza residual que dependiendo de su tamaño podrá ser considerada como desperdicio (*scrap*) *o podrá reutilizarse (normalmente son* mucho mayor que las que hay en los modelos basados en patrones). La formulación de Dyckhoff aborda el caso de múltiples longitudes en stock, el conjunto S representa las longitudes disponibles en stock para el corte, de forma que $S = (W_1, ..., W_K)$. El conjunto de piezas residuales cortadas lo denotamos como R. D representa el conjunto de longitudes demandadas. Las variables de decisión $y_{p,q}$ indica el número de veces que una pieza de longitud p se corta produciendo una pieza de longitud q y una pieza residual de longitud p - q. Las variables z_k indican el número de perfiles en stock de longitud W_k utilizados. El modelo queda de la siguiente manera:

$$Z_{one-cut} = \min \sum_{k=1}^{K} W_k z_k \tag{4.12}$$

s. a

$$z_k + \sum_{p \in D: p+q \in S \cup R} y_{p+q,p} \geq \sum_{p \in D: p<q} y_{q,p} \forall q \in S \tag{4.13}$$

$$\sum_{p \in S \cup R: p>q} y_{p,q} + \sum_{p \in D: p+q \in S \cup R} y_{p+q,p} \geq \sum_{p \in D: p<q} y_{q,p} + N_q \quad \forall q \in (D \cup R) \setminus S, \tag{4.14}$$

$$y_{p,q} \geq 0 \text{ y entero}, \; p \in S \cup R, \; q \in D, \; q < p, \tag{4.15}$$

$$z_k \geq 0 \text{ y entero}, \; k = 1, ..., K. \tag{4.16}$$

Las desigualdades (4.13) son la restricciones para definición de las variables w_k. En (4.14) N_q toma los valores de demanda si se trata de una barra y 0 si se trata de un resto. Para cortar una barra demandada se puede utilizar bien un perfil disponible en stock con longitud W_k o bien una pieza residual que satisfaga la factibilidad del corte (4.13). El cumplimiento de la demanda de una barra se garantiza mediante el corte de una pieza de

mayor dimensión o bien mediante la generación de un resto de la longitud de la barra (4.14). Al igual que ocurría con el modelo de Kantorovich, en este caso también se da simetría en el problema (dos valores diferentes en las variables para una misma solución real) a cambio el número de variables no es tan grande como en el modelo de los patrones de corte.

4.1.4 Modelo basado en grafos

Por último en (Valerio de Carvalho 1999) se propone una nueva formulación para el problema de corte estándar basado en un modelo de teoría de grafos. En este modelo una instancia del problema se como un grafo $G = (X, A)$ con $|X| = L + 1$ donde L es la longitud de los perfiles disponibles en stock (todos iguales). Cada uno de los nodos del grafo (X) representa una posición discreta dentro del objeto que se va a cortar, y cada uno de los arcos (h,j) el emplazamiento de un ítem de tamaño j-h donde la posición h está lo más a la izquierda posible del perfil. Por tanto se define un patrón de corte como una secuencia de arcos que empieza en el nodo o (vértice del objeto) más a la izquierda. Para un conjunto de arcos A y variables $x_{h,j}$ se puede formular el modelo de la siguiente manera

$$z_{grafo} = \min z \tag{4.17}$$

s. a

$$-\sum_{(h,j)\in A} x_{hj} + \sum_{(j,l)\in A} x_{jl} = \begin{cases} z, & \text{si } j = 0 \\ 0, & \text{si } j = 1,...,L-1 \\ -z, & \text{si } j = L \end{cases} \tag{4.18}$$

$$\sum_{(h,h+l_i)\in A} x_{h,h+l_i} \geq d_i, \quad i = 1,...,m, \tag{4.19}$$

$$x_{hj} \geq 0 \text{ y entero}, \quad \forall (i,j) \in A \tag{4.20}$$

Este modelo es equivalente al de patrones de corte. Al igual que le ocurre al modelo de corte único, éste también presenta simetría en las soluciones. Su principal debilidad es el número de restricciones que es pseudo-polinomial.

4.2 MÉTODOS PARA LA RESOLUCIÓN DEL 1D CSP

En (Dyckhoff 1991) se distingue entre dos tipos de enfoques a la hora de plantear la resolución del problema de corte. Por un lado el enfoque basado en patrones (*pattern oriented*) y por otro el enfoque basado en perfiles y barras (*object or item oriented*) en este caso la asignación entre barras y perfiles se realiza inmediatamente.

4.2.1 Procedimientos basados en el uso de patrones

En (Haessler, Sweeney 1991) se clasifican los métodos que utilizan patrones de corte en dos tipos de procedimientos: los enfoques de programación lineal; y los enfoques de secuenciación heurística.

4.2.1.1 Enfoques basados en programación lineal

Un procedimiento heurístico basado en Programación Lineal consiste en resolver la relajación del problema lineal LP para después modificar de alguna forma la solución lineal obtenida y convertirla en una solución entera. A su vez la obtención de la solución entera puede realizarse bien por métodos exactos, obtienen el óptimo entero, como son los planos de corte o la ramificación y acotación (Belov, Scheithauer, 2002), o bien mediante la aplicación de heurísticas residuales y de redondeo (Holthaus, 2002) y (Poldi, Arenales, 2009).

- **Obtención de la solución de la relajación lineal**

  Partiendo de la base de que la enumeración completa de todos los patrones de corte factibles es prácticamente imposible. Para resolver la relajación lineal del problema se tiene que aplicar un método basado en algoritmos de generación de columnas. De forma que el número de variables activas (patrones) es reducido y éstas se van incorporando al problema a medida que se van necesitando.

  Gilmore y Gomory (1961, 1963) fueron los primeros en aplicar generación de columnas al problema de corte. Mediante el *delayed pattern generation* que consiste en ir incorporando patrones al problema de programación lineal por medio de la resolución del problema asociado de la mochila, resolvieron la relajación lineal del

problema sin necesidad de enumerar previamente todos los patrones factibles. Para ello aplican el método símplex junto con un algoritmo de generación de columnas. Los nuevas patrones se van incorporándose a la matriz de patrones.

Aplicaciones posteriores de éste procedimiento parten de una matriz inicial de patrones (muchas veces la matriz identidad) y a medida que se van realizando iteraciones del algoritmo, sólo se almacenan los vectores columna correspondientes a las variables básicas del problema (Belov, Scheithauer, 2002).

La aplicación del *Delayed Pattern Generation* sería el siguiente: A partir de la relajación lineal del problema se plantea el problema dual. Donde u_i es la variable dual asociada a la restricción *i* del modelo primal basado en patrones de corte (4.9). El problema dual asociado para la relajación del modelo es

$$z_{dualLP} = \max \sum_{i=1}^{m} d_i u_i \tag{4.21}$$

s. a

$$\sum_{j=1}^{n} a_{ji} u_i \leq 1 \tag{4.22}$$

$$u_i \geq 0,\ \mathrm{i} = 1,...,m \tag{4.23}$$

Se platea el problema de la mochila asociado al problema dual. x_i representa el patrón solución del problema de la mochila

$$z_{mochila} = \max \sum_{i=1}^{m} u_i x_i \tag{4.24}$$

s. a

$$\sum_{i=1}^{m} l_i x_i \leq L \tag{4.25}$$

$$x_i \geq 0 \text{ y entero} \tag{4.26}$$

Si $Z \leq 1$, entonces la solución del problema dual es la óptima, si $Z > 1$, entonces la solución x_i pasa a formar parte de la matriz de patrones y se repite el procedimiento hasta encontrar la solución óptima.

- **Heurísticas residuales y de redondeo**

  Una vez obtenido el óptimo lineal, hay que obtener la solución entera (se asume que al menos uno de los componentes de la solución posee una parte fraccional). En (Poldi, Arenales 2009) se definen los siguientes conceptos previos a tener en cuenta a la hora de abordar la conversión de la solución continua en solución entera.

  Si se reescribe (por comodidad) la formulación matemática del modelo basado en patrones de corte, en forma matricial. Donde *A* es la matriz asociada a los patrones ($\{a_{ij}\}$).

$$\begin{aligned} &\min f(x) = c^t x \\ &Ax \geq d \\ &x \geq 0 \text{ y entero} \end{aligned} \tag{4.27}$$

  Definición 3.1 (*solución entera aproximada*). Sea x la solución de la relajación lineal del problema que cumple $Ax=d$. Sea y un vector de enteros próximos a x. Se dice que y es una aproximación entera a x si cumple / $Ay \leq d$, $y \geq 0$.
  Un ejemplo sería el vector $y=([x_1],[x_2],...,[x_n])$ que se obtiene de redondear una solución x a sus enteros inferior.

  Definición 3.2 (*problema residual).* Sea y una solución entera aproximada de x y sea $r=d-Ay$ la demanda que queda por satisfacer después de realizar los cortes con los patrones de y. Entonces el problema de satisfacer las demandas pendientes se llama problema residual.

  Precisamente el problema residual de corte (RCSP) apuntado en la clasificación de (Wäscher et al. 2007) debe su nombre a esta circunstancia, que se da después del redondeo de la solución continua del problema lineal. Nótese que el surtido de barras (ítems) en este caso es muy heterogéneo debido a que en cualquier caso se trata de demandas residuales.

  Entonces el marco de aplicación general que definen para la obtención de la solución entera consta de los siguientes pasos:

  Paso 1: sea k=0 (iteración) con $r^k=d$ (datos del problema original)

Paso 2: resolver la relajación lineal del problema residual (mediante generación de columnas). Sea x^k la solución continua del problema. Si es entera entonces parar.

Paso 3: determinar una aproximación entera a x^k. Sea y^k la solución entera aproximada obtenida mediante la aplicación de una heurística de redondeo. Si y^k = 0, entonces ir al paso final.

Paso 4: Actualizar los datos de demanda del nuevo problema residual $r^{k+1}=r^k - Ay^k$; ir al *paso* 2

Paso Final: resolver el problema residual.

La resolución del problema residual del *paso* final puede realizarse de diferentes maneras. Normalmente se utilizan algoritmos de tipo secuenciales como el FFD o algoritmos *greedy* (*ver siguientes secciones*).

La heurística de redondeo más evidente que se puede aplicar en el *paso* 3 del procedimiento descrito, sería el redondeo al siguiente entero inferior (*round down heuristic*) para las variables con valores no enteros. Otras heurísticas serían las de redondeo superior (*round up*) donde aplican técnicas de algoritmos *greedy*. En (Holthaus 2002) y (Lodi, Arenales 2009) proponen técnicas basadas en este tipo.

Otras heurísticas de redondeo son las desarrolladas en (Scheithauer, Terno 1995 y 1997) y en (Rietz et al. 2002), se basan en una modificación de la propiedad IRUP que hasta el momento poseen todas las instancias del problema de corte. Esta propiedad IRUP (*Integer Round Up Property),* algo así como propiedad del redondeo al entero superior, indica que el hueco entre las soluciones óptimas entera y continua nunca es superior a la dos y casi siempre es menor que uno.

- **Aplicación de métodos exactos**

  En los últimos años se han ido desarrollando procedimientos que aplican algoritmos de generación de columnas junto con métodos de exploración dirigida y planos de corte.

  En (Vance et al. 1994) se estudian técnicas que combinan la generación de columnas (*pricing*) con ramificación y acotación. A estos algoritmos se les denomina

branch and price. En (Vance et al. 1994) se analizan diferentes maneras de realizar la ramificación. La dificultad mayor con la que se encontraron fue que la resolución de los subproblemas (ramas) no estaba equilibrada en cuanto a la complejidad de su resolución. Mientras el subproblema acotado inferiormente era más pequeño y relativamente fácil de resolver, el subproblema acotado superiormente prácticamente era tan difícil de resolver como el inicial ya nuevas variables nuevas podían generarse. Esto llevó a Vanderbeck (1999) a proponer diferentes estrategias de ramificación, sugiere que la ramificación debe realizarse en aquellas columnas que compartan una propiedad común, una propiedad que pueda ser fácilmente identificable en subproblema. En el caso de que las barras se encuentren ordenadas en el patrón de corte, un buen criterio sería tomar la posición de la barra en el patrón.

Valerio de Carvalho (1998) introduce su modelo basado en teoría de grafos y establece reglas de ramificación basadas en las variables de su formulación. Recientemente en (Peeters, Degraeve 2006) se analizan reglas la ramificación sobre las variables del modelo de Gilmore y Gomory.

Otro tipo de procedimiento exacto son los algoritmos de ramificación y corte (***branch and cut***) son algoritmos de tipo ramificación y acotación pero basados en la relajación lineal del modelo. Los planos de corte ajustan la relajación lineal en los nodos del algoritmo de ramificación y acotación (Linderoth, Savelsbergh 1999).

Un tipo de algoritmo que también se ha propuesto en la literatura recientemente es el que combina los dos anteriores, es decir, la exploración dirigida con los planos de corte y la generación de columnas. Este algoritmo se denomina ***branch and cut and price*** (Belov, Scheithauer 2006). Recientemente en (Alvés, Valerio de Carvalho 2008) se ha propuesto un algoritmo de este tipo para la resolución exacta del problema de corte de múltiples dimensiones. Para garantizar el equilibrio en la fase de ramificación, utilizan un esquema basado en dos niveles. En el primer nivel ramifican en las variables fraccionales sobre el modelo basado en teoría de grafos y eligen aquella que corresponde al perfil más largo en stock. En caso de que ninguna de éstas sea fraccional, entonces la ramificación se realiza sobre los arcos asociados a las barras, se elige el más a la izquierda de los arcos fraccionales (el asociado a la barra más grande). Una vez hecha la ramificación, los subproblemas los resuelven

por programación dinámica. Los planos de corte factibles son el segundo tipo de desigualdad que utilizan para ajustar el problema. Sólo tienen en cuenta cortes factibles que dependen de un único tipo de barra.

4.2.1.2 Secuenciación Heurística (enfoque constructivo)

Uno de los inconvenientes que presentan las soluciones basadas en programación lineal, consiste en que el número de patrones tiende a igualarse al número de barras, resultando éste en la mayoría de ocasiones un número demasiado grande. Por otro, lado uno de los problemas extensivos del CSP sobre los que se está investigando en los últimos años es el problema de minimización de patrones o secuenciación de patrones que se justifica a partir de la necesidad de reducir costes por cambios de partida. Estos dos motivos han conducido al desarrollo de procedimientos de resolución basados en secuenciación heurística.

Estas técnicas generan secuencialmente los patrones de forma que se vayan satisfaciendo progresivamente las cantidades demandadas de barras. El procedimiento termina cuando todas las demandas están cubiertas.

Uno de los primeros procedimientos de secuenciación es el que se desarrolla en (Haessler 1971 y 1975). Se trata de un algoritmo heurístico del tipo *greedy* y que denomina SHP, el algoritmo va añadiendo nuevos patrones de corte a la solución hasta que todas las demandas son satisfechas. En cada paso éste genera candidatos de patrones de corte que satisfagan una porción de las demandas no completadas para posteriormente y de forma heurística seleccionar un patrón de la lista de candidatos cuya pérdida de material (*trim loss*) sea pequeña y su frecuencia alta. Otro ejemplo de procedimiento es el desarrollado en (Vasko et al. 1999) que consiste en un algoritmo jerárquico (*DYNACUT_CS*) para la generación de patrones que ejecuta repetidas veces un algoritmo de búsqueda basado en ramificación y acotación y que aplica en la industria acerera.

4.2.1.3 Procedimientos híbridos

En los últimos años han ido apareciendo procedimientos que podrían denominarse como híbridos y que consisten en integrar los dos enfoques arriba descritos (programación lineal y secuenciación) y conseguir soluciones mejores de las obtenidas individualmente por cada uno de éstos. En (Sweeney, Haessler 1990) se presenta un algoritmo híbrido que conjuga el SHP de Haessler con técnicas de programación lineal. Otros son los propuestos en (Johnston 1986) y en (Goulilms 1990), los algoritmos parten de una solución obtenida por técnicas de LP y van reduciendo el número de patrones por medio de la combinación de dos patrones en uno, de forma que las cantidades de producción que cubrían los dos patrones originales las cubre ahora el nuevo patrón. En (Foerster, Wäscher 2000) también se propone un algoritmo de estas características que llaman *KOMB*I que utiliza muchos tipos de combinaciones de patrones. (Vanderbeck 2000) propone un algoritmo exacto para la minimización del número de patrones del tipo ramificación y acotación. En esta línea se encuentran también (Becceneri et al. 2004).

Un apartado especial merece la reciente aplicación de meta heurísticas en la resolución del problema de corte. En (Liang et al. 2002) se propone un sencillo algoritmo evolucionado para la resolución del problema de corte con y sin contigüidad y en (Wagner, 1999) se desarrolla un algoritmo genético para la resolución del problema de corte unidimensional. En el caso de la resolución del problema de corte bidimensional, las publicaciones que proponen el uso de meta heurísticas cada vez son más frecuentes, algunos ejemplos de esto son; (Leung et al. 2001) que desarrollan un algoritmo genético, (Burke, Kendall 1999) que comparan tres diferentes algoritmos metaheurísticos: genético, búsqueda tabú y recocido simulado.

4.2.2 Resolución del problema de corte sin el uso de patrones

Los procedimientos que se han dado en denominar *object* o *item-oriented,* se caracterizan por un tratamiento individual de cada uno de los ítems que se han de cortar. Este enfoque se utiliza en el problema de corte residual (RCSP), donde el surtido de perfiles (objetos) es muy heterogéneo e incluso todos diferentes entre sí. No parece

oportuno determinar patrones de corte ya que la frecuencia de éstos sería de 1 y la asignación haría de manera directa.

Estos problemas se resuelven generalmente o bien por métodos exactos como son la ramificación y acotación y la programación dinámica, o bien por medio de algoritmos de aproximación. Los algoritmos de aproximación van estableciendo reglas de asignación que se van repitiendo hasta que la totalidad de las órdenes demandadas quedan satisfechas. En esta línea (Coffman et al. 1984) desarrollan diferentes algoritmos heurísticos; NF (*Next Fit*), FF (*First Fit*), BF (*Best Fit*), FFD (*First Fit Decreassing*), WF (*Worst Fit*), AWF (*Almost Worst Fit*), las diferencias entre ellos estriba en las reglas en las que se determinan tanto la secuencia del corte de perfiles en stock, como la secuencia en la que se deben satisfacer las demandas y la forma en la que se procesan los restos de material derivados del corte. El más difundido y con el que mejores resultados se obtuvieron fue el FFD, el algoritmo consiste en ordenar las barras en orden decreciente de longitud, seleccionar la primera barra y cortarla de un perfil en stock seleccionado de forma aleatoria y repetir el procedimiento hasta que queden satisfechas las órdenes demandadas.

Uno de los procedimientos de resolución del problema de corte sin el uso de patrones es el que utiliza la programación dinámica, (Antonio et al. 1999) proponen dos heurísticas basadas en programación dinámica para la resolución del problema de corte y que denominan HCUS y HCUSO (*Heuristic Constraining the Utilisation of the Stored Bars* y *Heurisstic Constraining the Utilisation of the Stored and Ordered Bars*). Otro ejemplo de la aplicación de programación dinámica lo constituye el algoritmo desarrollado en (Hifi, Ouafi 1997) para la resolución del problema de corte bidimensional con diferentes perfiles en stock. Otro de los métodos habituales empleados para la resolución exacta del problema de corte es el de ramificación y acotación.

4.3 EL PROBLEMA DE CORTE UNIDIMENSIONAL CON MULTIPLES TAMAÑOS EN STOCK (1D MSS CSP)

Como se ha comentado al principio del capítulo, **el problema identificado en la empresa de corte de perfiles estructurales es del tipo 1d MSSCSP** y por tanto

requiere una especial atención. Así pues, en esta sección se estudia el problema de corte unidimensional con perfiles de múltiples longitudes en stock. Se trata de una extensión natural del problema con perfiles iguales. Bien es cierto que no es la única ya que incluso dentro de diferentes longitudes se podrían también establecer distintos grados de calidad de los objetos como ocurre en problemas vinculados a aplicaciones en papel, madera y acero (Sweeny, Haesler 1990). El modelo matemático planteado en el capítulo segundo (apartado 2.3) para el corte de perfiles metálicos se basa en los modelos descritos en este apartado.

Parece lógico que al haber más diversidad de material en stock las soluciones al problema que cabe esperar deben ser mejores en términos de minimización de los desperdicios. Si bien esta diversidad de stocks también tiene un coste asociado en términos de coste computacional, el problema es más "difícil" de resolver además de que la propiedad de redondeo superior descrita en (Marcotte 1985) para los problemas de corte deja de cumplirse en el caso de múltiples longitudes con coste proporcional a sus dimensiones.

Frente al problema de SSSCSP, en el MSSCSP aparecen datos relativos al material almacenado. Ahora habrá un número de tipos distintos de perfiles, cada uno de ellos con una disponibilidad almacenada (en algunos estudios considerada ilimitada (Holthaus, 2002), con una longitud definida. Así pues el problema se caracteriza por los siguientes datos: $(p, m, L = (L_1, ..., L_p), e = (e_1, ..., e_p), p = (p_1, ..., p_p), l = (l_1, ..., l_m), d = (d_1, ..., d_m))$. Donde p es el número de tipos de perfiles disponibles ($k=1,\ldots,p$) en cantidades e_k y con un precio p_k m es el número de tipos de barras demandadas ($i=1,..,n$) en cantidades d_i. Cuando los precios de los perfiles son proporcionales a sus longitudes entonces nos encontramos con un problema de minimización de restos.

4.3.1 Modelo basado en patrones de corte

La extensión del modelo descrito en 4.1.2 para múltiples longitudes se podría plantear de la siguiente manera. Sea a $n = m + p$, un patrón de corte vendrá definido por

un vector columna $a=(a_1,...,a_n)^T \in \mathbb{Z}_+^n$ con n componentes que cumple las siguientes desigualdades

$$\sum_{i=1}^{m} l_i a_i \leq \sum_{i=1}^{p} L_i a_{i+m} \text{ y } \sum_{i=1}^{p} a_{i+m} = 1 \tag{4.28}$$

Los componentes de a_i entre 1 y m determinan cuantas barras de tipo i se cortan en el patrón. El componente a_{k+m} correspondiente al perfil k utilizado por el patrón será 1, para el resto de los componentes a_{i+m} donde $i \in \{1,...,p\} \backslash k$ su valor será 0.

Sea η el número de patrones corte. Sea la matriz $A=(a_{ij}) \in \mathbb{Z}_+^{\eta}$ donde cada columna representa un patrón de corte. El problema se puede formular de la siguiente manera. Determinar un vector de frecuencias de corte $x=(x_1,...,x_\eta)^T$ que minimice el coste del material para satisfacer la demanda de barras con las cantidades disponibles en stock:

$$z_{entera} = \min\left\{cx : x \in \mathbb{Z}_+^{\eta}\right\} \tag{4.29}$$

s.a

$$\begin{aligned} &\sum_{j=1}^{\eta} a_{ij} x_j \geq d_i, \quad 1 \leq i \leq m, \\ &\sum_{j=1}^{\eta} a_{ij} x_j \leq e_{i-m}, \quad m < i \leq n, \end{aligned} \tag{4.30}$$

Sea a un vector columna de la matriz A. Los coeficientes de la función objetivo $c:=(c(a))_{a \in A}$ y $c(a):=\sum_{i=1}^{p} p_i a_{i+m}$ son los costes del material utilizado en cada columna. En el caso de 1 único objeto sería 1. También se pueden considerar como coeficiente de la función objetivo los restos c_j que genera cada uno de los patrones de corte siguiendo la ecuación

$$\sum_{i=1}^{m} l_i a_{ij} + \sum_{i=1}^{p} c_j a_{i+m,j} = L_i a_{i+m,j} \tag{4.31}$$

4.3.2 Modelo basado en teoría de grafos

La extensión del modelo para múltiples longitudes basado en teoría de grafos que se ha descrito en 4.1.3 lo desarrollan Alves y Valério de Carvalho (2008). Dados unos

perfiles (k) con longitudes L_k, stocks e_k y una barras (i) de longitudes l_i y demandas d_i,.. La extensión del modelo para el problema de múltiples longitudes se definirá sobre un grafo $G = (X, A)$ donde X es el conjunto de nodos y A el conjunto de arcos. De forma que el grafo tiene tantos nodos como el mayor de las longitudes almacenadas en stock más uno $|X| = L_1 + 1$. Dentro del conjunto de arcos hay tanto arcos de barras como arcos de restos. Cada uno de los nodos del grafo (X) representa una posición discreta dentro del objeto que se va a cortar. Para cada uno de los arcos, sea (h,j), tenemos que $(h, j) \in A,\ si\ 0 \le h < j \le L_1\ y\ j - h = l_i,\ \forall 1 \le i \le m$. El grafo G tiene $O(mL_1)$ arcos y L_1+1 vértices. Por tanto se define un patrón de corte como una secuencia de arcos que empieza en el nodo o (vértice del objeto) más a la izquierda. Este modelo puede ser altamente simétrico, para reducir esta simetría se pueden aplicar diferentes criterios como los propuestos en (Valério de Carvalho 1999). La ordenación de las barras, por ejemplo, es decreciente y la ubicación de los arcos de restos al final del perfil. Para un conjunto de arcos A y variables $x_{h,,j}$ se puede formular el modelo de la siguiente manera

$$z_{grafo} = \min \sum_{k=1}^{p} L_k z_k \tag{4.32}$$

s. a

$$-\sum_{(h,j)\in A} x_{hj} + \sum_{(j,l)\in A} x_{jl} = \begin{cases} \sum_{k=1}^{p} z_k, & \text{si } j = 0 \\ -z_k, & \text{para } j = W_k,\ k = 1,\ldots,K, \\ 0, & \text{otro caso,} \end{cases} \tag{4.33}$$

$$\sum_{(h,h+l_i)\in A} x_{h,h+l_i} \ge d_i, \quad i = 1,\ldots,m, \tag{4.34}$$

$$z_k \le e_k,\ k = 1,\ldots,p \tag{4.35}$$

$$x_{hj} \ge 0 \text{ y entero}, \quad \forall (i,j) \in A \tag{4.36}$$

$$z_k \ge 0 \text{ y entero}, \quad k = 1,\ldots,p \tag{4.37}$$

El problema se formula como un problema de flujo mínimo en G, con restricciones adicionales de demanda de barras y disponibilidad de perfiles. Las variables del flujo se denominan por x_{hj} mientras z_k indica el número de perfiles de longitud L_k usados.

De acuerdo con el principio de descomposición de grafos de flujo, un conjunto de arcos de flujo en G puede descomponerse en un conjunto de caminos y ciclos de flujo. Los ciclos en G pueden definirse considerando K arcos adicionales que empiecen en las posiciones L_k, $k=1,...,p$ y acabando en el vértice 0. El siguiente ejemplo extraído de (Alves y Valério de Carvalho 2008) ilustra el modelo para una sencilla instancia.

Ejemplo. Sea una instancia con un conjunto de longitudes en stock W = (7,5) con disponibilidades B = (5,5) respectivamente, y un conjunto de longitudes demandadas w=(4,3,2) y demandas b = (2,5,2). El tamaño total de las demandas será $\sum_i w_i b_i = 27$ y en el caso del stock $\sum_k W_k B_k = 60$. En la siguiente figura se representa el grafo completo basado en el modelo de arcos de flujo obtido después de aplicar los criterios para la reducción de la simetría.

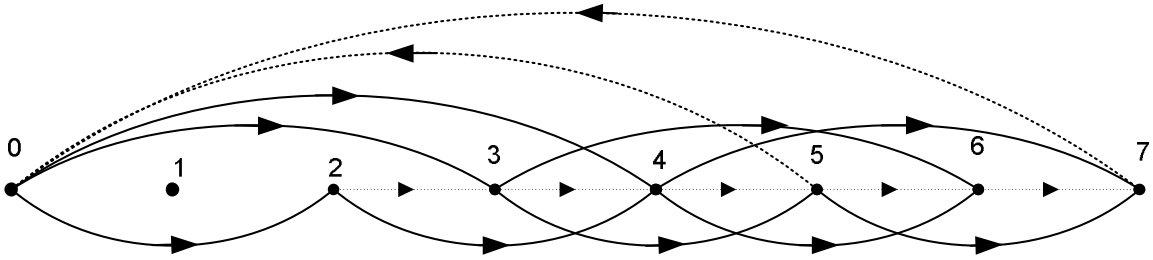

Figura 4.1- Modelo basado en grafo (Ejemplo)

4.4 RESOLUCIÓN DEL PROBLEMA DE CORTE CON MÚLTIPLES LONGITUDES

Los métodos que se han utilizado a lo largo de los años para la resolución del problema de múltiples longitudes se pueden enmarcar dentro de los enfoques descritos en el apartado 3.2. En esta sección se desarrolla la particularización para el problema de los conceptos generales descritos en el apartado 3.2, como puede ser la aplicación del la técnica de generación de columnas aplicada en los enfoques basados en programación lineal. Por otro lado se describen publicaciones recientes

4.4.1 Particularización del algoritmo de generación de columnas

La relajación lineal del problema entero planteado en 4.3.1 se obtiene eliminando la condición de integridad de las variables de decisión (4.2) e introduciendo las variables de holgura en las desigualdades (4.3)

$$z_{continua} = \min\left\{cx : x \in \mathbb{R}_+^{\eta}, s_i \in \mathbb{R}_+^1, 1 \le i \le n\right\} \tag{4.38}$$

s.a

$$\begin{aligned} &\sum_{j=1}^{\eta} a_{ij}x_j - s_i = d_i, \quad 1 \le i \le m, \\ &\sum_{j=1}^{\eta} a_{ij}x_j + s_i = e_{i-m}, \quad \mathrm{m}<i \le n, \end{aligned} \tag{4.39}$$

El valor de $z_{continua}$ será una cota inferior para el problema entero. El problema dual asociado será

$$z_{dual} = \max \sum_{i=1}^{m} d_i u_i + \sum_{i=1}^{p} e_i u_{i+m} \tag{4.40}$$

s. a

$$\sum_{j=1}^{\eta} a_{ij}u_i \le c_i, \quad i = 1,...,m,m+1,...,m+p \tag{4.41}$$

$$u_i \ge 0 \forall \mathrm{i} = 1,...,m, \quad u_i \le 0 \forall i = m+1,...,m+p \tag{4.42}$$

Se platea el problema de la mochila asociado al problema dual. x_i representa el patrón solución del problema de la mochila que se incorporará a la base.

$$z_{mochila} = \max \sum_{i=1}^{m+p} u_i x_i \tag{4.43}$$

s. a

$$\sum_{i=1}^{m} l_i x_i + \sum_{i=1}^{p} L_i x_{i+m} \le 0 \tag{4.44}$$

$$\sum_{i=1}^{p} x_{i+m} = -1 \tag{4.45}$$

$$x_i \ge 0 \text{ y entero} \forall i = 1,...,m \quad x_i = (0,-1) \forall i = m+1,...,m+p \tag{4.46}$$

Para simplificar los cálculos, en muchas ocasiones se propone resolver el problema de la mochila asociado a cada una de los perfiles por separado y evaluar la función objetivo para cada perfil. El problema de la mochila asociado a un perfil (k) sería

$$z^{k}_{mochila} = \max \sum_{i=1}^{m+p} u_i x_i \tag{4.47}$$

s. a

$$\sum_{i=1}^{m} l_i x_i + \sum_{i=1}^{p} L_i x_{i+m} \leq 0 \tag{4.48}$$

$$\sum_{i=1}^{p} x_{i+m} = -1 \tag{4.49}$$

$$x_i \geq 0 \text{ y entero } \; 1 \leq i \leq m, \quad x_{i+m} = \begin{cases} 0, & i < k \\ 1, & i = k \\ 0, & i > k \end{cases} \tag{4.50}$$

4.4.2 Enfoques basados en heurísticas de redondeo y problemas residuales

En (Holtahaus 2002) se estudia el caso de múltiples longitudes con disponibilidad ilimitad, resuelve la relajación lineal del problema mediante generación de columnas y usa tres procedimientos para redondear la solución hasta llegar al problema residual que resuelve mediante un ILP-solver . Recientemente en (Poldi, Arenales 2009), también se proponen tres diferentes procedimientos heurísticos de redondeo tipo *greedy* para la resolución de los problemas residuales: en el primer procedimiento se da prioridad a la hora de ordenar los patrones a aquellos con un uso más elevado (GRH1); en el segundo se priorizan los patrones en función del resto (GRH2); en el tercer procedimiento se ordenan las variables en función de su parte fraccional (GRH3).

4.4.3 Enfoques basados en métodos exactos

Probablemente debido a la mayor capacidad computacional para resolver problemas de optimización, en los últimos años ha crecido el número de publicaciones que

desarrollan algoritmos del tipo *branch and price and cut*. En (Belov, Scheithauer 2002) se propone un método exacto que combina los planos de corte de Chvatal y Gomory. Recientemente en (Alves, Valerio de Carvalho 2008) se estudia el uso de desigualdades duales dentro de un algoritmo tipo *branch and price and cut* para la convergencia de la técnica de la generación de columnas en todos los nodos del árbol.

4.4.4 Enfoques de secuenciación heurística

Probablemente una de las aplicaciones más interesantes de la secuenciación heurística aparece cuando se añade la restricción de minimizar el número de patrones debido a los costes por cambio de partida. A lo largo de los años se han ido proponiendo variantes de éste así como aplicaciones del algoritmo a diferentes problemas de corte (múltiples longitudes en stock,...).

Varios son los tipos de algoritmos que se han propuesto para la minimización del cambio de patrones en problemas con múltiples longitudes: El primero es una variante utilizada en (Gradisar 1999) del algoritmo SHP tipo greedy propuesto en (Haesler 1990). SHP añade de manera secuencial nuevos patrones de corte a la solución inicial hasta que todas las demandas quedan satisfechas. En cada paso, primero genera candidatos que satisfagan parte de la demanda remanente y luego mediante una heurística selecciona uno de los patrones de la lista de candidatos teniendo en cuenta su bajo desperdicio y su alta frecuencia de uso. (Vahrenkamp 1996) propone una variante del SHP en el que el nuevo patrón se genera mediante un simple algoritmo aleatorio. El segundo es un método heurístico llamado “combinación de patrones” como el que propone (Goulimis 1990). El método utiliza la solución continua y reduce el número de patrones mediante una heurística que combina dos patrones en uno. El tercero es un algoritmo exacto propuesto en (Vanderbeck 2000), PMP, primero se formula el problema como un problema entero cuadrático y después se descompone en múltiples problemas de la mochila acotados con “fuertes” relajaciones lineales, la ramificación y acotación se aplica en la técnica de generación de columnas. En (Umetanni et al. 2003) se propone un nuevo enfoque metaheurístico que combina un algoritmo de búsqueda local con la generación de patrones adaptativa mediante un algoritmo *greedy*. Recientemente en (Aktin, Özdemir 2009) se desarrolla un enfoque basado en dos modelos separados de programación lineal entera para generar patrones en un primer estadio y después generar el plan de corte.

En (Johnston, Sadinlija 2004) se ha propuesto un nuevo modelo matemático en el que las variables que generan los patrones de corte son binarias x_{ijk}, que toman el valor 1 si el ítem *i* se repite *k* veces en el patrón *j*. El número de patrones es limitado de antemano de forma que la minimización de patrones por cambios de partida así como su secuenciación, son dos de los criterios junto con el de minimización de los desperdicios. El problema de este modelo radica en la necesidad de generar miles de variables para instancias basadas en aplicaciones reales.

CAPÍTULO 5

ALGORITMOS GENÉTICOS PARA LA RESOLUCIÓN DEL PROBLEMA 1D MSSCSP

Como se ha descrito en el capítulo anterior, el problema de corte que surge en el corte de vigas metálicas pertenece al tipo unidimensional con múltiples longitudes disponibles en stock, y que según la terminología de (Wäscher et al. 2007) se denota como 1D MSSCSP. En el presente capítulo se desarrolla e implementa una metodología basada en el uso de algoritmos genéticos que permite resolver este problema con un buen nivel de desempeño.

A tal efecto, en el apartado 5.1 se presentará la modelización del problema de corte descrito en el capítulo 2 tomando como referencia los modelos descritos en el capítulo 4. A continuación, en el apartado 5.2 se revisarán las principales aportaciones bibliográficas que hasta el momento han propuesto el uso de técnicas evolutivas para la resolución del problema de corte. En el apartado 5.3 se describe la metodología propuesta basada en algoritmos genéticos para la resolución eficaz del problema de corte que se estructura en tres fases y consiste en la aplicación de: un algoritmo genético para la obtención de patrones de corte; un algoritmo genético que solucionará inicialmente el problema de corte; y cuatro algoritmos para la búsqueda de patrones incompletos y para la mejora de las soluciones iniciales (de agrupación, genético residual y longitudes menores). Por último en el apartado 5.4 se implementarán éstos y se mostrarán los resultados computacionales obtenidos que demuestran la eficiencia del método. En este contexto se hace necesario

desarrollar un generador de problemas de test para que a partir de determinadas instancias se pueda evaluar la efectividad de los algoritmos.

5.1 MODELIZACIÓN MATEMÁTICA DEL PROBLEMA DE CORTE DE PERFILES

A la hora de modelizar matemáticamente el Problema de Corte identificado en el capítulo 2 de la presente memoria, se tendrá en cuenta lo siguiente:

- El corte ser realiza en una sola de las dimensiones de las piezas (longitud).
- Consideraremos todos los cortes como si fueran rectos.
- En cada programa de producción consideraremos únicamente una sección de perfil estructural, ya que cada sección es independiente de las demás y cada una poseerá su programa de corte independiente.
- La asignación de las vigas demandadas a las longitudes disponibles en stock se realiza mediante el uso de patrones de corte.
- Hay más una longitud de perfil disponible en stock. Inicialmente el problema debería considerar los distintos tipos de sección que se pueden encontrar en el caso real. Sin embargo, dado que las secciones de las vigas no son intercambiables, el problema es separable. Por lo que al **hablar de tipos de perfiles se hará referencia a su longitud nominal**, **entendiéndose que la forma y tamaño de la sección ya está definida**.
- El plazo de entrega de los pedidos es suficientemente grande para garantizar que, el corte de los pedidos dados por el programa de producción obtenido se realiza a tiempo.
- Todos los pedidos son pedidos en firme.

Se distinguen 3 tipos de datos:

- Los que tienen que ver con los **pedidos y las órdenes de fabricación**
    - Viga demandada: perfil del que se trata, cantidad y longitud.

- Los que tienen que ver con el **material en stock**
    - Perfil en stock, cantidad y longitud

- Los que tienen que ver con el proceso de corte (**patrones de corte**)

- Matrices de relación de corte, entre vigas demandadas y vigas en stock
- Restos generados por cada patrón

Podemos caracterizar el Problema de Corte en base a los siguientes datos:

$$p,\ m,\ L=(L_1,...,L_p),\ e=(e_1,...,e_p),\ \mathrm{v}=(v_1,...,v_p),\ l=(l_1,...,l_m),\ d=(d_1,...,d_m)$$

donde el índice p es el número de tipos de perfiles disponibles ($k=1,...,p$) de longitudes L_k, en cantidades e_k y con un precio v_k. El índice m es el número de tipos de vigas demandadas ($i=1,..,m$) de longitudes l_i, en cantidades d_i.

Modelizamos[1] el Problema de Corte usando el concepto de patrón de corte, donde se tiene que $n = m + p$ y que un patrón de corte vendrá definido por un vector columna $a=(a_1,...,a_n)^T \in \mathbb{Z}_+^n$ con n componentes que cumple las n siguientes desigualdades

$$\sum_{i=1}^{m} l_i a_i \le \sum_{i=1}^{p} L_i a_{i+m} \text{ y } \sum_{i=1}^{p} a_{i+m} = 1 \qquad (5.1)$$

Los componentes de a_i entre 1 y m determinan cuántas vigas de tipo i se cortan en el patrón. El valor del componente a_{k+m} correspondiente al perfil k utilizado por el patrón será 1, y el del resto de los componentes a_{i+m}, donde $i \in \{1,...,p\} \setminus k$, su valor será 0.

Sea η el número de patrones corte. Sea la matriz $A=(a_{ij}) \in \mathbb{Z}_+^{\eta}$ donde cada columna representa un patrón de corte. Por tanto el Problema de Corte se formula de la siguiente manera:

Determinar un vector de frecuencias de uso de cada patrón de corte $x=(x_1,...,x_\eta)^T$ *que minimice el coste del material para satisfacer la demanda de vigas con las cantidades disponibles en stock*

$$z_{entera} = \min\{cx : x \in \mathbb{Z}_+^{\eta}\} \qquad (5.2)$$

s.a

[1] El modelo matemático aquí presentado es una particularización del que se describía en el apartado 4.3.1 del capítulo 4.

$$\sum_{j=1}^{\eta} a_{ij} x_j \geq d_i, \quad 1 \leq i \leq m,$$
$$\sum_{j=1}^{\eta} a_{ij} x_j \leq e_{i-m}, \quad \mathrm{m}<i \leq n, \tag{5.3}$$

donde los coeficientes c_j de la función objetivo son los restos r_j que genera cada uno de los patrones de corte, calculados con la ecuación:

$$r_j = \sum_{i=1}^{p} L_i a_{i+m,j} - \sum_{i=1}^{m} l_i a_{ij} \tag{5.4}$$

En caso de que se planteara bonificar aquellos restos que puedan ser reutilizables en futuros horizontes de planificación, se podría establecer un coeficiente γ_j que redujera el coeficiente de la función objetivo si el resto pasa a considerarse una punta almacenable para futuros programas de corte. En concreto, en el caso de la empresa objeto de estudio, un resto se considera chatarra para valores inferiores a *400mm* de longitud, así que de forma que el coeficiente de la función.

$$c_j = \begin{cases} r_j & \text{si } r_j<400 \\ \gamma_j * r_j & \text{si } r_j \geq 400 \end{cases} \tag{5.5}$$

5.2 MÉTODOS EVOLUTIVOS EN LA RESOLUCIÓN DEL PROBLEMA DE CORTE UNIDIMESIONAL

En comparación con otro tipo de técnicas, pocas son las publicaciones que proponen algoritmos evolutivos para la resolución del problema de corte. Esto es debido, entre otras cosas, a la dificultad en mantener la esencia del concepto de recombinación de los progenitores sin violar de las restricciones del problema al aplicarla directamente. Es decir, en la mayoría de los casos, cuando se realiza una recombinación directa de dos soluciones "padre", se obtienen soluciones "hijo" no factibles. En contrapartida, un algoritmo genético maneja grandes poblaciones de individuos o soluciones lo que permite ofrecer mayor diversidad cuando la toma de decisión implica múltiples objetivos (Wagner 1999)

Como se ha indicado en la sección 2.4 del capítulo 3, un algoritmo genético, evalúa toda una población o conjunto de individuos en cada iteración. Forman parte de los algoritmos que se denominan evolutivos y que se basan en la idea de la selección natural de forma que el conjunto de soluciones a lo largo de las iteraciones va evolucionando de acuerdo con unos operadores genéticos: recombinación, clonación y mutación. Una de las ventajas de los algoritmos evolutivos es que permiten explorar el espacio de soluciones de forma rápida e "inteligente".

Las principales referencias bibliográficas para la resolución del problema de corte unidimensional por medio de métodos evolutivos, como son los algoritmos genéticos (GA) o la programación evolutiva (EP), las encontramos en (Wagner 1999) en su aplicación a un problema de corte de paquetes de tablones de madera y en (Hinterding y Kahn 1995) y en (Liang et al. 2002). Éstos últimos también estudian la resolución del problema secuenciación de patrones corte, en el que hay que establecer la secuencia en la que se cortan las vigas teniendo en cuenta limitaciones de espacio para el almacenaje de los paquetes inacabados, y que aparece en la literatura bajo las denominaciones de *Contiguity, Pattern Sequencing* o *Open Stacks*

Revisando estas publicaciones, dos codificaciones diferentes son las que se han utilizado a la hora de plantear la estrategia de resolución. La primera, representa la solución al problema en términos de agrupamientos previos de vigas en patrones de corte y que se denomina "representación basada en grupos" (*group-based*). Este tipo de representación ha resultado adecuada tanto para problemas de corte con contigüidad como sin contigüidad. La segunda codifica la solución al problema como una lista ordenada de todas las vigas demandadas. En este caso, la secuencia de la lista diferencia una posible solución de otra y se denomina "representación basada en orden" (*order-based*). Su desempeño es adecuado sólo en el caso de problemas de corte con contigüidad. Con la representación basada en el orden se han utilizado tanto algoritmos genéticos como programación evolutiva (Liang et al. 2002)

- **Representación de una solución basada en grupos**

  El **cromosoma** (solución del problema) se codifica como una secuencia de *n* **genes**, que se corresponden con los grupos o patrones de corte. Para este tipo de

representación, es necesaria la definición previa de cada uno de estos genes/grupos/patrones a la aplicación del algoritmo de resolución del problema. Con este fin, en (Hinterding y Khan 1995), por ejemplo, se propone un algoritmo del tipo *First Fit* para la generación de los patrones de corte, de forma que se elige de manera aleatoria un perfil en stock y luego se aplica el algoritmo de modo que se minimice el desperdicio generado. En (Wagner 1999), sin embargo, se adapta un procedimiento de (Pierce 1964) en el que se generan todos los posibles patrones de corte eficientes. La eficiencia viene determinada por un valor máximo prefijado de desperdicio admisible, de esta forma además, se sugiere que implícitamente una solución al problema minimizará el desperdicio.

Como se muestra en la siguiente figura, cada grupo o gen lleva la siguiente información asociada: conjunto de vigas que incluye en el patrón de corte, tipo de perfil en stock que se corta y cantidad de desperdicio o resto generado por el corte.

Longitudes de vigas demandadas:
2, 5, 6, 7, 9 y 10

Longitudes de los perfiles en stock:
12, 13 y 15

Posibles grupos o patrones:

Grupo	Vigas que corta	Perfil	Desperdicio
A	(10)	12	2
B	(6 5 2)	13	0
C	(9 2)	12	1
D	(7 6)	13	0
E	(10 5)	15	0

Representación de una solución con 10 genes

AACBDDDEEE

Figura 5.1- Representación basada en grupos

Un grado distintivo de este tipo de representación es la variabilidad en el número de genes que pueden llegar a componer el cromosoma (solución del problema). Mientras. Mientras en (Hinterding y Khan 1995) se establece la cantidad de genes en un cromosoma como un valor variable, en (Wagner 1999) éste valor es fijo y se determina en función de las longitudes tanto de los perfiles en stock disponibles como de las vigas demandadas. Posibles problemas derivados de hacer fijo el número de genes en una

solución son la sobreproducción de vigas no demandadas y la no satisfacción de algunas de las demandadas.

En cuanto a la aplicación de los aspectos relacionados con los algoritmos genéticos en la representación basada en grupos, la implementación del operador de **recombinación** (*crossover*) es una tarea difícil ya que éste debe mantener la factibilidad de las soluciones "hijo", en caso contrario algún tipo de penalización se debe introducir en el algoritmo para potenciar las nuevas soluciones factibles. Wagner (1999), por ejemplo, selecciona un punto aleatorio de recombinación en los cromosomas padres y el nuevo hijo se genera a partir de la primera porción de un padre y la segunda del otro. En el caso de la implementación del operador de **mutación** para la representación basada en grupos, ha consistido en eliminar algunos genes (habitualmente el 3%) y sustituirlos por otros nuevos. De esta manera se intenta evitar una convergencia prematura del algoritmo. (Hinterding, Kahn 1995) eliminan los genes que generan más desperdicio mientras que (Wagner 1999) hace una selección aleatoria, ya que la eficiencia del patrón queda garantizada en el proceso previo de creación de patrones.

A la hora de abordar la **evaluación** de cada solución, encontramos diferentes propuestas en la literatura. Así por ejemplo, Wagner (1999) incorpora tres términos ponderados en su **función eficiencia**: minimización de la cantidad total de perfiles utilizados (lo que minimiza el desperdicio generado), minimización de los niveles de inventario finales (penalización por sobreproducción) y la minimización de las vigas demandadas no fabricados (penalización por "infraproducción") como se muestra en (5.6).

$$\min \sum_k w_k \sum_j X_{jk} + \sum_i w_i I_i^{1+} + \sum_i (I_i^{1-})^2 \qquad (5.6)$$

donde i, j, k son las vigas, patrones y perfiles respectivamente y w_k, w_i son los factores de ponderación de los perfiles totales cortados e inventarios finales. I_i^{1+}, I_i^{1-} Indican la sobreproducción y la "infraproducción" respectivamente. X_{jk} es el numero de veces que se cortará el patrón j vinculado al perfil k. En función del valor de la función objetivo, se asigna a cada solución una probabilidad de supervivencia, de modo que aquellos

cromosomas con mejores valores tienen una mayor probabilidad de sobrevivir. Dependiendo de esa probabilidad, se seleccionan de manera aleatoria los cromosomas que se usaran en la siguiente generación y se van haciendo parejas que se recombinaran. Otro ejemplo de funciones de eficiencia la encontramos en (Hinterding y Kahn, 1995) tanto para la resolución del problema de corte sin contigüidad como para el corte con contigüidad. A continuación se representan las funciones objetivo para el problema de corte sin contigüidad (5.7) y para el problema de corte con contigüidad (5.8). En el caso del corte sin contigüidad, la función objetivo contiene dos términos, el primero busca minimizar el desperdicio y el segundo potenciar las soluciones que contienen genes sin restos. En el caso del corte con contigüidad, el primer término de la función objetivo también minimiza los restos mientras que el segundo minimiza el número de paquetes abiertos en cada corte:

$$\min \frac{1}{n}\left(\sum_{n} \left(\sqrt{\frac{R_j}{L_j}} \right) + \frac{N}{n} \right) \tag{5.7}$$

$$\min \frac{1}{n+10}\left(\sum_{n} \left(\sqrt{\frac{R_j}{L_j}} \right) + \frac{10}{m} \sum_{n} \left(\frac{o_j}{m} \right)^2 \right) \tag{5.8}$$

donde,

L_j = longitud del perfil del gen j

R_j= resto del gen j

o_j= número de paquetes abiertos cuando se ha cortado el gen j

N= cantidad de genes con algo de desperdicio

n= numero de genes

m= numero de vigas diferentes demandados

- **Representación de una solución basada en orden**

  El cromosoma se representa como una lista ordenada con todas las vigas que se tienen que cortar. En este caso, los patrones de corte se asignarían a posteriori mediante un algoritmo tipo *Next Fit* en el que se completa el patrón en el momento en el que la

siguiente viga de la lista no cabe. La implementación del algoritmo *Next Fit* es sencilla cuando sólo hay un único tipo de perfil en stock. En caso de varias longitudes en stock se podría optar por la aplicación de un algoritmo de tipo ramificación y poda. En las figuras siguientes se muestran dos ejemplos de la representación basada en orden.

Vigas y cantidades demandadas
Longitud 3 4 5 6
Cantidades 2 2 1 2

Lista ordenada de vigas (cromosoma)
(5, 4, 6, 3, 3, 4, 6, 6)

Longitud del perfil en stock: 12

Posibles grupos o patrones:

Patrón	Vigas que corta	Perfil	Desperdicio
A	(5 4)	12	3
B	(6 3 3)	12	0
C	(4 6)	12	2
D	(6)	12	6

Figura 5.2- Representación basada en orden para un único perfil en stock (Liang et al. 2002)

Vigas y cantidades demandadas
Longitud 3 4 5 6
Cantidades 2 2 1 2

Lista ordenada de vigas (cromosoma)
(5, 4, 6, 3, 3, 4, 6, 6)

Longitud del perfil en stock: 12 13 15

Posibles grupos o patrones:

Patrón	Vigas que corta	Perfil	Desperdicio
A’	(5 4 6)	15	0
B’	(3 3 4)	12	2
C’	(6 6)	12	0

Figura 5.3- Representación basada en orden para varios perfiles en stock

(Hinterding, Kahn 1995) proponen un algoritmo genético para este tipo de codificación. El principal problema que surge en esta representación es la aplicación del operador de recombinación. A tal efecto, utilizan una plantilla generada aleatoriamente de bits binarios que permite intercambiar las vigas a la vez que mantiene la información relativa

al orden de las soluciones “padres” que denominan *uniform order-based crossover* (UOC).

Motivados por el hecho de que el desempeño de los algoritmos genéticos no es bueno cuando se aplica la recombinación, (Liang et al. 2002) proponen un algoritmo basado en programación evolutiva (EP), donde únicamente utilizan la mutación como generador de nuevas generaciones, sin aplicar ningún tipo de recombinación. Utilizan las mismas funciones objetivo (5.7) y (5.8) que utilizaban en (Hinterding, Kahn 1995) en sus algoritmos genéticos.

La mutación que proponen (Liang et al. 2002) para el caso del problema de corte sin contigüidad consiste en el 3PS (*3 Point Swaps)* que aplica intercambios (*swaps*) entre 3 vigas de la lista. La idea de esta mutación viene del concepto de distancia entre dos listas con los mismos elementos. Dadas dos permutaciones de n elementos $X=(x_1,...,x_i,...,x_j,...,x_n)$ e $Y=(y_1,...,y_i,...,y_j,...,y_n)$, $1<i<j<n$, la distancia mínima entre ellas puede definirse como $D(X,Y)=1$ si $x_i=y_j$, $x_j=y_i$ y $x_k=y_k$ para todos los otros k.

En otras palabras la distancia mínima entre dos listas se determina por el número mínimo de intercambios dos a dos que hay que realizar para pasar de una lista a la otra. En teoría, una solución óptima global puede alcanzarse a través de una secuencia de intercambios (swaps) sencillos a partir de una población inicial de listas. Para evitar un óptimo local, y porque el intercambio dos a dos (2PS) puede resultar muy largo, (Liang et al. 2002) proponen el intercambio de tres puntos o 3PS, de forma que en cada aplicación del 3PS, la nueva lista se ha movido una distancia de 2.

En el caso del problema con contigüidad, (Liang et al. 2002) además del 3PS también aplican el operador SRI hasta cuatro veces por cada mutación. El SRI (*Stock Remove and Insert*) tiene como objetivo reducir el número de paquetes de vigas parcialmente acabados a la vez que se sigue minimizando el desperdicio. El SRI reordena la secuencia de corte sin realizar ningún otro cambio. Su implementación se realiza siguiendo los siguientes pasos:

1- Se selecciona una viga al azar cortada por un patrón
2- Se quita de la lista el patrón que contiene esa viga
3- Se busca dentro de la lista otro patrón que contenga esa misma viga

4- Los dos patrones de corte se ubican en la lista juntos (uno a continuación del otro).

Para evaluar el desempeño del enfoque basado en EP, (Liang et al. 2002), lo comparan con los resultados obtenidos por los GA de (Hinterding, Kahn 1995) sobre 10 problemas de test (cinco con múltiples perfiles en stock y cinco con un único tipo de perfil en stock), con un tamaño de población de 75 individuos, 10 élite y 50 evaluaciones por problema. En todos los casos el algoritmo EP tiene un desempeño al menos tan bueno como los de los GA y en la mayoría de los casos obtuvo mejores soluciones.

5.3 UNA METODOLOGÍA BASADA EN ALGORITMOS GENÉTICOS PARA LA RESOLUCIÓN DEL PROBLEMA DE CORTE UNIDIMESIONAL

En este apartado se describe el método basado en algoritmos genéticos propuesto para la resolución del 1dimensional MSSCSP. La resolución completa del problema de corte que aquí se propone consta de un conjunto de técnicas y algoritmos que se van ejecutando de forma secuencial a lo largo de varias fases. En la siguiente figura se muestra una visión general del método propuesto identificando cada uno de los elementos correspondientes a cada fase.

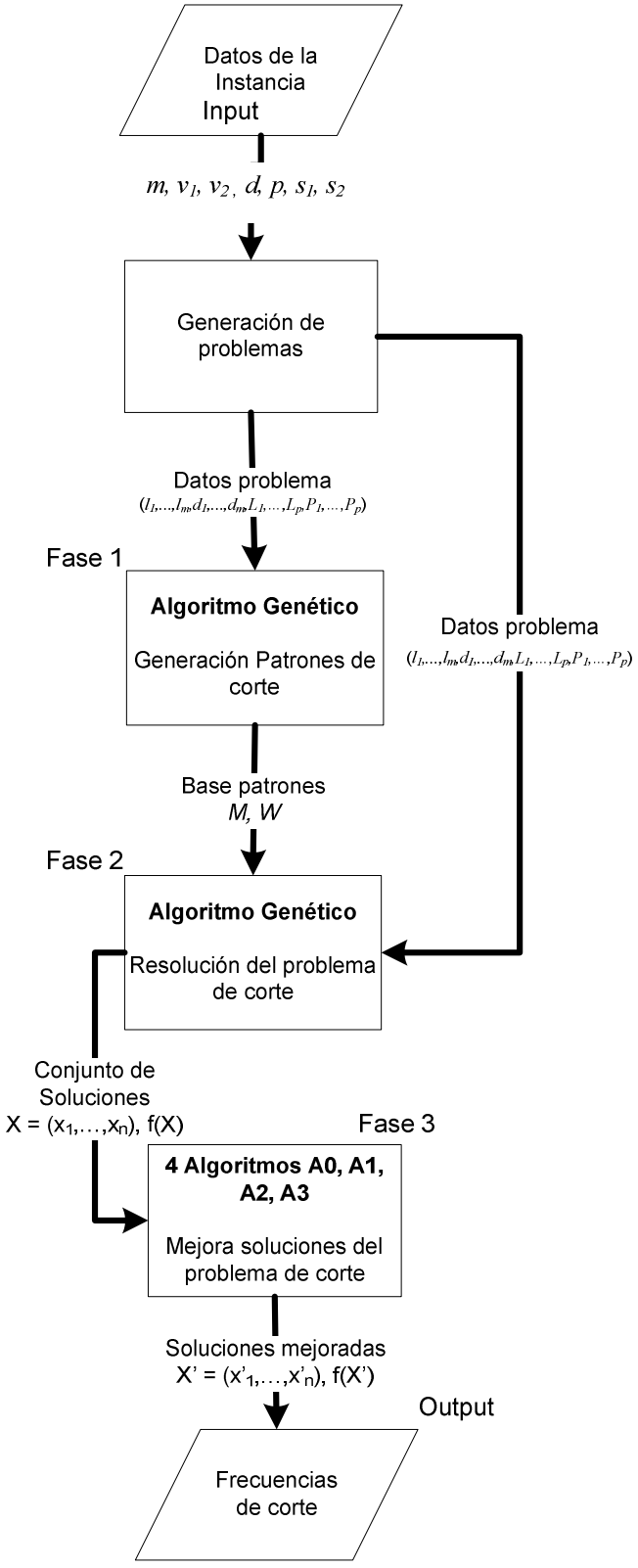

Figura 5.4- Visión general del enfoque de resolución propuesto

A partir de un generador de instancias y problemas de test que se describe en el apartado 5.4, se generará una base de patrones de corte mediante la aplicación de un algoritmo genético (apartado 5.3.1). Los patrones de corte constituirán la información de entrada para la solución del problema de corte propiamente dicho que a su vez se resuelve en dos etapas. En la primera etapa se aplicará otro algoritmo genético (descrito en el apartado 5.3.2) que generará, tras varias iteraciones, una población suficientemente grande de soluciones. Del total de las soluciones obtenidas, se seleccionará un conjunto que contenga las de mejor desempeño. Posteriormente, en una segunda etapa, se aplicarán cuatro algoritmos diferentes, tres de ellos en paralelo, (descritos en el apartado 5.3.3) con el fin de depurar y mejorar el desempeño de las soluciones obtenidas respecto a posibles sobreproducciones o eventuales insatisfacción de demandas.

En nuestro algoritmo se opta por representar las soluciones siguiendo el modelo basado en grupos (g*roup based*). Como se ha indicado en 5.1 mediante este tipo de codificación la implementación de algoritmos genéticos es factible, en el caso de las basadas en orden no es así (Hinterding, Kahn 1995). Sin embargo, la representación basada en grupos obliga a determinar los patrones de corte con antelación.

Como se vio en el apartado anterior, (Wagner 1999) y (Hinterding, Kahn 1995) utilizan técnicas diferentes para determinar la base de patrones que utilizarán sus algoritmos de resolución. Este problema auxiliar de generación de patrones, surge también en enfoques basados en métodos de programación lineal.

5.3.1 Un algoritmo genético para la generación de patrones (FASE 1)

Habitualmente la generación de patrones se ha abordado de dos maneras: previamente a la resolución del problema o en línea. La primera se plantea para problemas de tamaño pequeño y mediano con enumeración completa o para problemas grandes en los que únicamente se genera un subconjunto de patrones representativo. La generación en línea, en cambio, se usa para resolver problemas grandes mediante técnicas de generación de columnas. En el caso de aplicar metaheurísticas para resolver el problema de corte, se hace necesario generar con antelación los patrones de corte.

Así por ejemplo, (Gilmore, Gomory 1996 y 1963) emplean una técnica de generación de columnas basada en el método simplex que posteriormente ha sido usada por multitud de autores. (Haessler 1971 y 1975) desarrolla un procedimiento de secuenciación heurística. (Christofides, Whitlock 1977) diseñan un procedimiento enumerativo basado en las órdenes de corte (simetría) para la generación de las columnas. (Suliman 2001) propone una heurística sencilla basada en un método de solución descrito para el problema de la mochila y que usa un árbol de búsqueda para su desarrollo.

Parece lógico pensar que una condición necesaria para el buen desempeño de las técnicas de resolución del problema de corte sería la generación de patrones eficientes que satisfagan los objetivos del problema (minimización de desperdicio, paquetes abiertos,…) a la vez que la población de patrones sea suficientemente diversa. Con estas consideraciones, se describe a continuación un algoritmo genético para la generación de patrones eficientes (ver apartado 5.4.3) que nos sirvan de base para la resolución del problema de corte.

Un algoritmo genético, a partir de una población inicial de individuos, irá realizando iteraciones basadas en la aplicación de operaciones de selección natural: recombinación, mutación y clonado. A tal efecto, los individuos generados en cada iteración se evalúan mediante una función de eficiencia que los ordena por ranking. Los individuos con mejor evaluación pasa de forma directa a la siguiente generación (élite o clonación), el resto de la población se generará o bien por recombinación o bien por mutación.

En nuestro caso, el algoritmo genético para la generación de patrones se ejecutará tantas veces como perfiles contenga la instancia del problema, de forma que se generen patrones para todos los tipos de perfiles.

Como parámetros de entrada al algoritmo genético se definen los siguientes:

- Tamaño de la población (Pop)
- Número de generaciones máximo o condiciones de finalización del algoritmo
- Tamaño de la población que pasará de una generación a otra ($Elite$)
- Proporción individuos generados por recombinación y por mutación (tasas de recombinación y mutación, T_{rec} y T_{mut}) de forma que se cumpla la siguiente expresión:

$$Pop = Elite + T_{rec} * (Pop - Elite) + T_{mut} * (Pop - Elite) \quad (5.9)$$

$$1 = T_{rec} + T_{mut} \quad (5.10)$$

- **Codificación del patrón**

A la hora de desarrollar un algoritmo genético que permita generar patrones de corte, la principal dificultad radica en crear una codificación efectiva del problema. Una posible representación de un patrón de corte (j) sería un vector en el que se especificaran las diferentes cantidades de cada viga demandada (i) que se obtendrían al cortar un perfil concreto (k). $X_j = (X_{1j}.., X_{ij})$. En la Figura 5.5 se muestra un ejemplo de la codificación de patrones.

Longitudes de vigas (i) demandadas:
(2, 5, 6, 7, 9, 10)= (l_1, l_2, l_3, l_4, l_5, l_6)

Longitudes de los perfiles (k) en stock:
(12, 13, 15) = (L_1, L_2 , L_3)

Posibles grupos o patrones:

Patrón j	vigas que corta	Codificación	Perfil cortado (k)
1	(10)	(0, 0, 0, 0, 0, 1)	(1)
2	(6 5)	(0, 1, 1, 0, 0, 0)	(2)
3	(7 2 2)	(2, 0, 0, 1, 0, 0)	(1)
4	(7 6)	(0, 0, 1, 1, 0, 0)	(2)
5	(10 5)	(0,1, 0, 0, 0, 1)	(3)

Figura 5.5- Codificación de los patrones de corte

Sin embargo, con este tipo de codificación no se consigue una aplicación eficaz de los operadores genéticos (las soluciones obtenidas por mutación y recombinación con frecuencia no son soluciones factibles). Así que se opta por una codificación diferente que es similar a la utilizada por (Anand et al. 1999) para el problema de corte bidimensional que sí permita el uso eficaz de los operadores genéticos y que se describe a continuación. Creamos un espacio auxiliar A cuyas coordenadas sean más manipulables por el algoritmo genético y que puedan ser convertidas en coordenadas del espacio S. De forma que en lugar de establecer las cantidades de cada viga (i), el código auxiliar a utilizar para cada codificar los patrones asociados a cada tipo de perfil (k) es un vector de N_k componentes que indican la secuencia ordenada de las N_k vigas que se superponen sobre la longitud del perfil (k) que utiliza ese patrón. Para determinar

de forma realista (que se ajuste a la realidad de los datos del problema) el valor de N_k para los patrones correspondientes a cada tipo de los perfil (*k*), se tendrá en cuenta la longitud de ese perfil (L_k) y la longitud media de las vigas demandadas ponderadas en función de las cantidades demandadas de cada tipo de viga. Se calcula el número de componentes N_k mediante la siguiente expresión:

$$N_k = \left\lceil \frac{L_k}{\left(\frac{\sum_i d_i * l_i}{\sum_i d_i} \right)} + 0,5 \right\rceil \qquad i = 1,...,m, \tag{5.11}$$

donde d_i es la demanda para cada una de las m vigas, l_i su longitud y L_k la longitud del perfil al que va asociado el patrón. Este tipo de codificación auxiliar resuelve los problemas que se planteaban con la codificación original del problema y es de fácil implementación. En el caso del ejemplo anterior, la codificación propuesta se implementaría de la siguiente manera:

Longitudes de vigas (*i*) y cantidades demandadas:
(2, 5, 6, 7, 9, 10)= (l_1, l_2, l_3, l_4, l_5, l_6); (4,3,2,1,2,2)=(d_1, d_2, d_3, d_4, d_5, d_6)

Longitudes de los perfiles (*k*) en stock:
(12, 15, 17) = (L_1, L_2 , L_3)

Valores calculados de N_k con la expresión (5.6): 2, 3 y 3 para cada perfil

Posibles grupos o patrones:

Patrón j	vigas que corta	Codificación	Perfil cortado (k)
1	(10)	(6, 0)	(1)
2	(6 5)	(3, 2, 0)	(2)
3	(7 2)	(4, 1)	(1)
4	(7 6)	(4, 3, 0)	(2)
5	(10 5)	(6, 2, 0)	(3)

Figura 5.6- Alternativa propuesta para la codificación de los patrones de corte

- **Función de eficiencia**

  Las funciones de eficiencia utilizadas perseguirán la minimización de los restos generados por el corte de perfiles y se recogen en las expresiones (5.12) y (5.13).

  La función que definirá la eficiencia en términos de tasa de restos generados se obtiene de la siguiente expresión:

$$trlossef_j = 1 - \frac{\sum_{h=1}^{N_k} l_{hj}}{L_j} \tag{5.12}$$

  donde L_j es la longitud del perfil utilizado por el patrón j y l_{hj} son las longitudes de las vigas generadas por el patrón j.

  Mientras que la función de eficiencia en términos de resto absoluto, lo cual puede ser útil de cara a un futuro estudio de reusabilidad de restos, se determinará mediante la expresión:

$$trloss_j = L_j - \sum_{h=1}^{N_k} l_{hj} \tag{5.13}$$

- **Generación de patrones**

  El procedimiento que se utilizará para la generación de patrones para cada tipo de perfil es similar al descrito por (Anand et al. 1999) y que ha sido empleado para la generación de patrones bidimensionales. Con este procedimiento se evitará sobrepasar los niveles de demanda fijados para cada viga. Un patrón perteneciente al perfil k, contendrá como máximo N_k vigas ordenadas. El procedimiento para determinar dichas vigas comprende los siguientes pasos:

  1- Seleccionar N_k valores $r_1,...,r_{Nk}$ aleatorios entre 0 y 1

  2- Encontrar el valor de i_0 que hace cumplir la siguiente expresión:

$$\sum_{i<i_0} d_i < r_h * \sum_{i=1}^{m} d_i \leq \sum_{i \leq i_0} d_i \qquad h = 1,...,N_k \tag{5.14}$$

3- Fijar la viga i_0 como el componente h del patrón. Pasar al siguiente h hasta completar los N_k componentes.

Una vez se ha obtenido el patrón, **se confirma que cumpla que la suma de las longitudes de todas las vigas sea menor o igual a la longitud del perfil**, de no ser así se hacen nulos, empezando por el último, los componente necesarios hasta hacer factible el patrón (esto evitará la negatividad de la función de eficiencia).

$$\sum_{h=1}^{N_k} l_{hj} \leq L_j \tag{5.15}$$

donde L_j es la longitud del perfil utilizado por el patrón j y l_{hj} es la longitudes de la viga h de las N_k cortadas por el patrón j.

- **Proceso de clonado o élite**

  En cada generación se ordenarán las soluciones por eficiencia y se establecerá una tasa de clonación o élite que hará pasar a la siguiente generación a la proporción elegida de las de mayor eficiencia. El parámetro de tasa de clonación deberá ser prefijado.

- **Procesos de recombinación y mutación**

  Para llevar a cabo la recombinación y mutación, es necesario seleccionar un conjunto de individuos de la generación anterior que serán los progenitores de los nuevos individuos de la siguiente generación, esta selección se realizará aleatoriamente. En ambos casos se debe fijar una tasa de recombinación y una de mutación, de forma que la suma de los individuos obtenidos por clonación y los obtenidos por recombinación y mutación sea igual al tamaño de la población establecido de acuerdo con (5.9) y (5.10).

  La recombinación consiste en generar nuevas soluciones a partir de la mezcla de otras. Puede ser llevada a cabo de varias maneras. En este caso de entre dos progenitores seleccionados aleatoriamente, se eligen genes de uno y de otro hasta completar un nuevo individuo "hijo". Para generar n individuos harán falta $2n$ progenitores. El procedimiento adoptado es el de "*cruce de Bernoulli*" en el que en cada recombinación se fija un parámetro, **umbral de cruce**, entre 0 y 1. A continuación se realiza la operación de cruce para lo que se seleccionan $r_1,...,r_{Nk}$ números aleatorios entre 0 y 1.

Si el valor de r_k es inferior al del valor del umbral, los componentes *k-ésimos* de los progenitores se intercambian. Se obtienen dos soluciones hijo que se evalúan de acuerdo a la función de eficiencia y de las dos se selecciona aquella de mejor eficiencia.

La mutación consiste en modificar una proporción de las soluciones ya existentes. Para dotar de mayor diversidad a cada iteración, se propone generar patrones totalmente nuevos de acuerdo con la expresión (5.14) y en la cantidad indicada por la tasa de mutación (*1- tasa de recombinación*).

Inherente a las operaciones de creación de nuevos patrones está la comprobación de la factibilidad de esos nuevos patrones de acuerdo con la expresión (5.15).

- **Condiciones especiales en la ejecución del algoritmo**

  La ejecución del algoritmo no parece adecuada en aquellos casos en los que la cantidad de todos los patrones posibles sea reducida. En este caso se optará por realizar la enumeración completa de los posibles patrones.

  Un patrón de corte, de acuerdo a como se ha codificado, no es más una combinación de entre $m+1$ elementos, m tipos de vigas y el valor 0, agrupados en conjuntos de N elementos, valor obtenido en la ecuación (5.11), y con la particularidad de que los elementos se pueden repetir sin que importe el orden en el que se encuentran. Así pues, todos los posibles patrones que se pueden generar para un perfil se obtendrían mediante la aplicación de la siguiente ecuación (combinaciones con repetición de $m+1$ elementos tomados de N en N):

$$CR_{m+1}^{N} = \frac{V_{(m+1+N)-1}^{N}}{P_N} = \frac{(m+N)!}{N! \cdot m!} = \binom{m+N}{N} \qquad (5.16)$$

  En la siguiente tabla se muestran el número de posibles combinaciones obtenidas para diferentes valores de m y *N*.

m / N	*1*	*2*	*3*	*4*	*5*	*6*
1	2	3	4	5	6	7
2	3	6	10	15	21	28
3	4	10	20	35	56	84
4	5	15	35	70	126	210
5	6	21	56	126	252	462
6	7	28	84	210	462	924
7	8	36	120	330	792	1716
8	9	45	165	495	1287	3003
9	10	55	220	715	2002	5005
10	11	66	286	1001	3003	8008
11	12	78	364	1365	4368	12376
12	13	91	455	1820	6188	18564
13	14	105	560	2380	8568	27132
14	15	120	680	3060	11628	38760
15	16	136	816	3876	15504	54264
16	17	153	969	4845	20349	74613
17	18	171	1140	5985	26334	100947

Tabla 5.1- Combinaciones con repetición de m+1 elementos tomados de N en N

Para aquellos casos en los que el número de combinaciones sea bajo, no tendrá sentido la aplicación del algoritmo genético y la selección de los patrones se realizará por enumeración completa. Más adelante, en el apartado de implementación y resultados computacionales, se establecerán como parámetros de trabajo del algoritmo genético una población de 50 individuos y un valor máximo de generaciones de 100 (si el algoritmo no encuentra variaciones superiores al 1% a lo largo de las generaciones en la función de eficiencia, éste detiene la ejecución). Para estos datos, y a partir de un pequeño análisis previo, se ha fijado en 250 el número de combinaciones hasta la cual se realizará la generación de patrones por enumeración completa. En estos casos se fija en 12 el valor máximo de patrones a seleccionar en función de su eficiencia de entre todos los posibles enumerados.

A parte, otra situación excepcional la encontramos en el caso en que el número de vigas demandas es menor o igual que el número de vigas que admite el patrón. Por lo que en principio un único patrón podría contendría todas las demandas.

La generalización del algoritmo para la totalidad de longitudes de perfiles disponibles se realizará de manera sencilla; se generarán los patrones para cada una de las longitudes como se ha indicado en este apartado y posteriormente se unificarán todos en una única matriz M de dimensiones $m \times T$, donde m es el número de vigas diferentes y donde $T=P*G$, P (número de perfiles diferentes) y G (tamaño de la población). Por otro lado se obtendrá también una matriz W que relacionará cada patrón (j) con cada perfil (k). Ambas matrices M y W se tomarán como base para la resolución del problema de corte.

5.3.2 Un algoritmo genético para la resolución del problema de corte (FASE 2)

Como se ha indicado al principio de esta sección, para representar las soluciones al problema de corte se opta por una codificación basada en grupos (patrones).

Al igual que ocurría en el algoritmo para la generación de patrones, en este caso se definen también como parámetros de entrada los siguientes:

- Tamaño de la población (Pop)
- Número de generaciones máximo o condiciones de finalización del algoritmo
- Tamaño de la población que pasará de una generación a otra ($Elite$)
- Proporción individuos generados por recombinación y por mutación (tasas de recombinación y mutación, T_{rec} y T_{mut}) de forma que se cumpla la siguiente expresión.

- **Codificación de una solución al problema de corte**

  Una solución al problema consistirá en una secuencia de patrones que satisfaga la demanda de vigas, sin sobrepasar las existencias de perfiles disponibles en stock y que minimice un determinado objetivo: resto generado; inventarios finales; cambios de partida; espacio ocupado por el trabajo en curso; etc. Por tanto, una solución al

problema codificada en términos genéticos sería un cromosoma con un número G de genes en el que cada gen corresponde a uno de los patrones de corte utilizados.

Con todos los patrones de corte generados por el algoritmo genético del apartado 5.3.1 se formará la base de patrones que en el modelo matemático del apartado 5.1 equivaldría a la matriz $A = (a_{ij}) \in \mathbb{Z}_+^{\eta}$, de la que se seleccionarán los patrones de manera aleatoria.

A la hora de fijar el número de genes que debe tener un cromosoma, unos autores optan por establecer un valor fijo y otros por uno variable. En nuestro algoritmo, con el objetivo de mantener la diversidad de las soluciones, se establecen dos valores enteros máximo y mínimo entre los que oscilará de forma aleatoria el número genes no nulos (patrones) que tendrá cada individuo (solución) de la población. Estos valores se determinan de acuerdo con las siguientes expresiones:

$$G_{\max} = \left\lceil \frac{\sum_{i=1}^{m} d_i}{\min(\sum_{i=1}^{m} a_{ij})} \right\rceil, \; G_{\min} = \left\lceil \frac{\sum_{i=1}^{m} d_i}{\max(\sum_{i=1}^{m} a_{ij})} \right\rceil \qquad (5.17)$$

donde d_i son las demandas de las vigas (i) y $\min\left(\sum_{i=1}^{m} a_{ij}\right)$ y $\max\left(\sum_{i=1}^{m} a_{ij}\right)$ son las cantidades de vigas de los patrones (j) que menor y mayor cantidad de vigas cortan de entre todos los que conforman la base respectivamente.

- **Función de eficiencia**

  A la hora de evaluar las soluciones se deberán tener en cuenta los siguientes objetivos: minimizar desperdicio generado en el corte (suma de los restos generados por cada uno de los patrones que interviene en la solución), satisfacer todas las demandas, minimizar los inventarios finales de vigas (sobreproducción de vigas). La función de eficiencia que se propone es similar a la que Wagner (1999) utiliza en su algoritmo:

$$\min w_{resto} \sum_j X_j * trloss_j + w_{inventario} \sum_i I_i^{1+} + w_{insatisfechas} * \sum_i \left(I_i^{1-} \right)^2 \quad (5.18)$$

donde X_j es la solución representada en forma de vector de dimensión N (cantidad total de patrones en la base), donde sus componentes indican cuántos patrones del tipo j se utilizan en la solución. Donde w_{resto} $w_{inventario}$ y $w_{insatisfechas}$ son pesos aplicados a los resto, al inventario en exceso y a las demandas insatisfechas. I_i^{1+} representa los inventarios finales de vigas (i) (sobreproducción) e I_i^{1-} representa las demandas no satisfechas, ambas obtenidas a partir de las siguientes expresiones.

$$I_i^{1+} = \max\left(I_i^0 + \left[\sum_j X_j * a_{ij} \right] - d_i, 0 \right) \quad (5.19)$$

$$I_i^{1-} = \max\left(d_i - I_i^0 - \left[\sum_j X_j * a_{ij} \right], 0 \right) \quad (5.20)$$

La I_i^0 representa los niveles iniciales de vigas en almacén (si los hubiera).

- **Generación de una población inicial de soluciones**

  La generación de una población inicial de soluciones se realizará de la siguiente manera:

  1- Se asigna a cada individuo un valor entero de forma aleatoria entre G_{min} y G_{max}.

  2- Se asignan de manera aleatoria patrones de entre los de la base hasta el valor obtenido en el paso anterior. A medida que se va efectuando esta asignación, se comprueba que no se viole la disponibilidad en stock de perfiles en almacén. En caso de no ser así se dejan de asignar los patrones correspondientes a ese perfil.

  3- Se hacen cero el resto de genes hasta completar las G_{max} posiciones.

- **Proceso de clonado o élite**

  En cada generación se ordenarán las soluciones por eficiencia y se establecerá una tasa de clonación o élite que hará pasar a la siguiente generación las soluciones de mayor eficiencia. El parámetro de tasa de clonación deberá ser prefijado.

- **Procesos de recombinación y mutación**

  La selección de los progenitores, al igual que en el generador de patrones, se realizará aleatoriamente. En cada recombinación se selecciona aleatoriamente uno de los dos patrones de los progenitores para cada gen. Posteriormente los genes nulos se envían al final del cromosoma y se comprueba la factibilidad de la solución, es decir, que no se sobrepasa el stock de perfiles disponible en almacén. Un ejemplo del proceso se muestra en la figura 5.7.

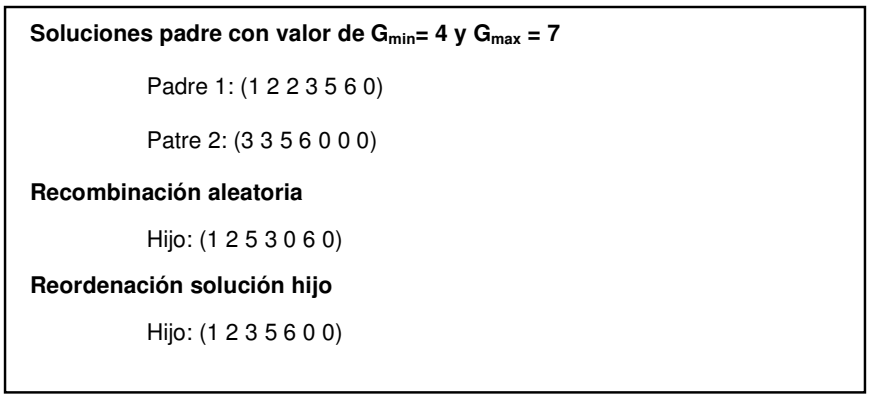

Figura 5.7- Procedimiento de recombinación

En cuanto a la mutación, para dotar de mayor diversidad a cada iteración, se generarán soluciones totalmente nuevas en la cantidad indicada por la tasa de mutación (*1- tasa de recombinación*) y siguiendo el mismo procedimiento utilizado para la población inicial.

- **Condiciones especiales en la ejecución del algoritmo**

  Al igual que ocurría con la generación de los patrones de corte, el conjunto de las soluciones posibles consiste en combinar con repetición y sin importar el orden, un conjunto de patrones agrupados en grupos de $Gmax$ elementos. Por tanto, cuando el número de posibles combinaciones sea menor de 2000, se optará por la enumeración completa y se seleccionarán las mejores soluciones, con un máximo de 100 elegidas. Nótese que el número de patrones posibles habitualmente es elevado (la cantidad obtenida en la Fase 1 para la totalidad de los perfiles) con lo que valores inferiores a 2000 se darán únicamente en los casos en los que: el valor de $Gmax$ sea *1*; cuando $Gmax$ sea *2* y la cantidad de patrones generados menor de *62*; cuando $Gmax$ sea *3* y la cantidad de patrones generados menor de *21* (muy difícil que se dé esta situación). En todos estos casos, la enumeración de las posibles soluciones no resulta complicada.

5.3.3 Algoritmos para la mejora de las soluciones obtenidas (FASE 3)

Posiblemente la principal limitación de un enfoque de resolución del problema de corte basado exclusivamente en el uso de algoritmos genéticos radique en la no satisfacción completa de todas las demandas, así como en la incurrencia en vigas sobreproducidas (inventarios en exceso). Pese a que la función de eficiencia utilizada por el algoritmo genético del apartado anterior (5.18) sí tiene en cuenta consideraciones de este tipo, se observa que bajo determinadas circunstancias de los parámetros del problema (elevado número de vigas, bajo nivel de demandas,…) se obtienen soluciones con niveles de sobreproducción mayores de los deseables e incluso alguna demanda queda parcialmente insatisfecha, lo que supone la violación de las restricciones originales del problema.

Así pues ante la limitación del algoritmo genético a obtener soluciones óptimas, en este apartado se propone la aplicación de una secuencia de algoritmos **novedosa** cuya finalidad es la de convertir en factible la solución obtenida por el algoritmo genético del apartado 5.3.2. A partir de estrategias basadas en el agrupamiento de patrones o selección de perfiles de longitud menor para patrones con mucho resto, se irá buscando la mejora de las soluciones de modo que se satisfaga la totalidad de las demandas y se elimine la sobreproducción.

El método se muestra en la figura 5.8 y consiste en la aplicación de cuatro algoritmos distintos. El primero de los algoritmos que hemos denominado **Algoritmo de patrones incompletos** busca identificar aquellos patrones que contribuyen en mayor medida a la sobreproducción de vigas, estos patrones se eliminarán de la solución. Las demandas insatisfechas (I) junto con las vigas pertenecientes a los patrones incompletos que no eran sobreproducción constituirán lo que hemos denominado **problema residual**. A este problema residual se le aplicarán en paralelo tres algoritmos basados en estrategias distintas: **Algoritmo de agrupamiento, Algoritmo de longitudes menores y Algoritmo genético residual**.

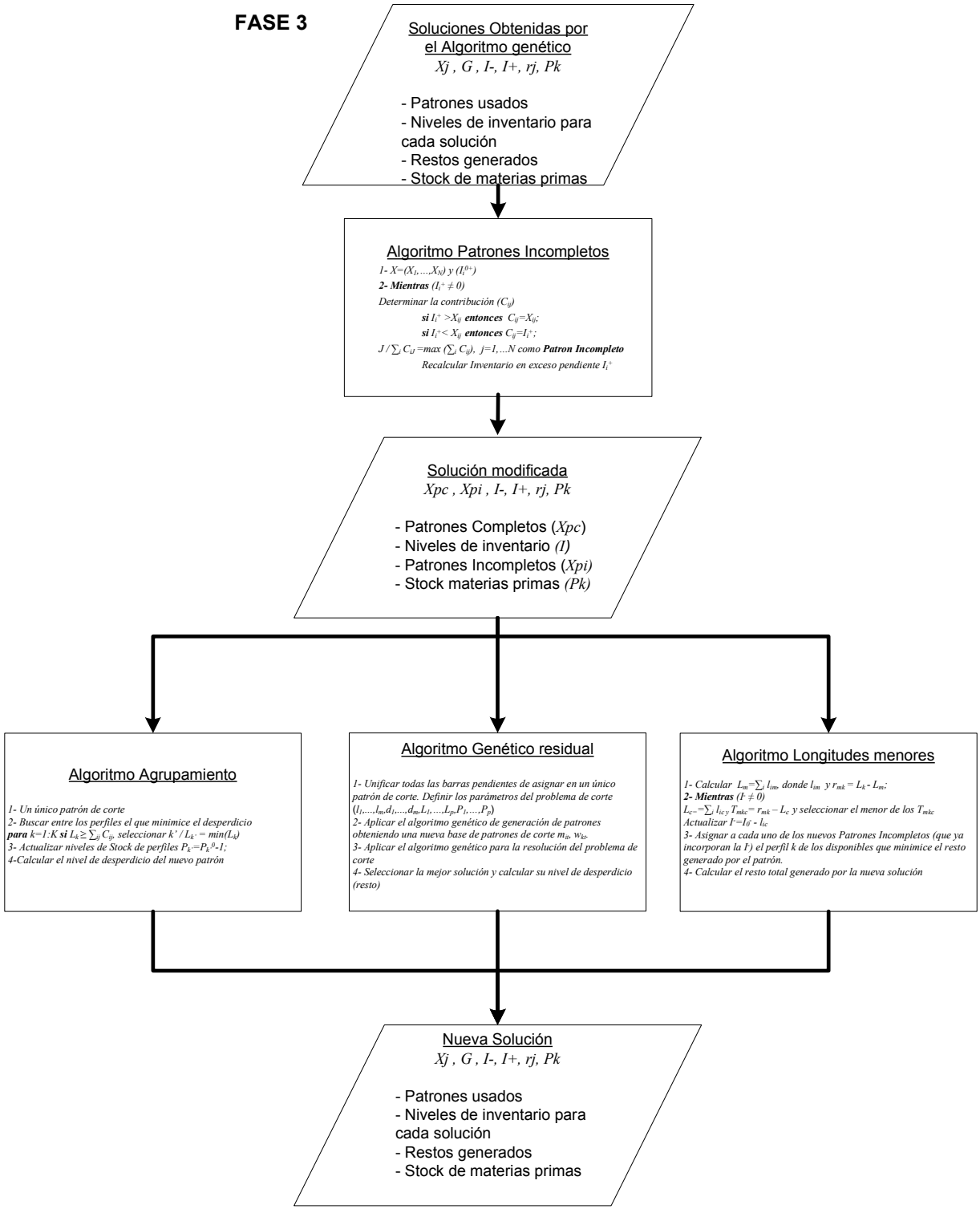

Figura 5.8- Esquema general Fase 3: modificación soluciones

Como se ha visto, el algoritmo genético del apartado 5.3.2 alcanzará en su última iteración una población de soluciones para el problema de corte. Cada una de estas soluciones contendrá un número G de patrones (entre G_{max} y G_{min}), que a su vez están definidos por las vigas que corta, el perfil al que está vinculado y el resto que genera. A consecuencia de la aplicación de estos patrones de corte, se obtienen por un lado las vigas que satisfarán las demandas y por otro restos o puntas de producción. En los casos en los que las vigas cortadas de un tipo superen sus demandas, se tendrá también un inventario en exceso para ese tipo de viga (I^+) y en el caso en el que no se corte la cantidad total demandada para un tipo de viga, se tendrá una no satisfacción de la demanda (I^-). El objetivo de los algoritmos que a continuación se describen será el de hacer nulas las I^+ y I^- vinculadas a cada solución, manteniendo a la vez el nivel de restos bajo.

Para ello y **para cada solución**, el primer problema que se plantea es el de determinar cuáles han sido los patrones y en qué medida han contribuido a la sobreproducción de vigas y así poder "atacar" al problema de la generación de inventario en exceso. Por tanto, dentro de una solución nos encontraremos con que un número de sus patrones (Q) generan vigas que se dedican a satisfacer las demandas íntegramente y que llamaremos ***patrones completos*** (todas sus vigas se suministran a los clientes) y otro (G-Q) que al generar algunas vigas no demandadas, podrían no cortarse completamente y que llamaremos ***patrones incompletos***. No cortar algunos patrones íntegramente generara un resto mucho mayor aunque bien es cierto que tendría más posibilidades de reutilizarse en futuros periodos de producción como nuevos perfiles disponibles en stock.

Es evidente que desde el momento en el que las vigas demandadas se produzcan a partir de más de un patrón, no existirá una única respuesta para la identificación de los *patrones incompletos*. Así pues, nos encontramos con la dificultad añadida de plantear un método que no sólo determine un conjunto de *patrones incompletos*, sino que además éste sea lo más ventajoso posible para la minimización de restos generados por patrones no cortados completamente.

Una de las estrategias que se plantearán más adelante para la eliminación de la sobreproducción es el algoritmo de agrupamiento (descrito más abajo), que consiste en la agrupación de los *patrones incompletos* en un único patrón nuevo. Parece lógico que para poder aplicar esta estrategia con éxito, interesará que la sobreproducción se atribuya a un número pequeño de patrones, que sea lo más pequeño posible. Así pues, el método que se

propone para la identificación de los *patrones incompletos* buscará primero a aquellos patrones que contribuyen a la sobreproducción en mayor medida.

- **Algoritmo para la obtención de los *patrones incompletos* de una solución**

  El algoritmo para la obtención de los *patrones incompletos* irá seleccionando entre los patrones (*j*) de una solución al problema de corte aquellos que más contribuyan a la sobreproducción de vigas siguiendo los siguientes pasos.

Algoritmo

1- Inicialización

Seleccionar una solución al problema de corte $X=(X_1,\ldots,X_N)$

Calcular los Inventarios iniciales en exceso para cada viga (I_i^{0+})

2- ***Mientras*** $(I_i^+ \neq 0)$

Determinar la contribución (C_{ij}) *de cada patrón j al* I^+ *de la viga i*

si $I_i^+ > X_{ij}$ ***entonces*** $C_{ij}=X_{ij}$;

si $I_i^+ \leq X_{ij}$ ***entonces*** $C_{ij}=I_i^+$;

Seleccionar el patrón $J / \sum_i C_{iJ} = max\ (\sum_i C_{ij}),\ j=1,\ldots N$ *como* ***Patron Incompleto***

si *hubiera más de un valor de J, elegir el patrón cuyo resto (rj) sea menor*

Recalcular Inventario en exceso pendiente I_i^+

En el siguiente esquema se representa la manera de determinar los *patrones incompletos* por el algoritmo donde C_{lj}^+ representa la contribución al inventario en exceso del patrón *j* en la etapa *l* e I_0^+ es el inventario en exceso inicial.

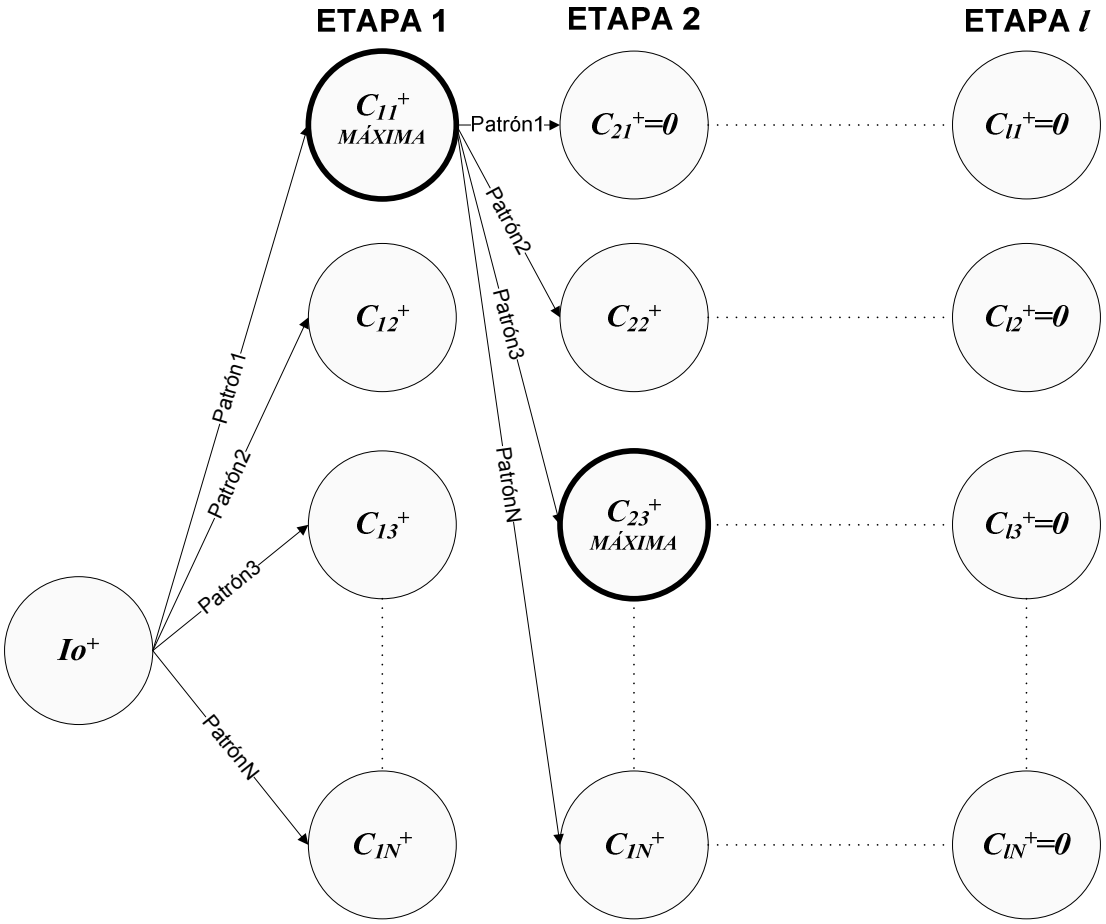

Figura 5.9- Representación del algoritmo de búsqueda de Patrones Incompletos

Imaginemos un problema de corte definido por los datos del cuadro izquierdo de la figura 5.10: se demandan 10 tipos diferentes de vigas con longitudes entre valores 100 y 200 unidades de longitud; y se deben obtener a partir de 5 tipos distintos de perfiles cuyas longitudes se encuentran entre valores de 1200 y 3200.

Tipos de vigas demandadas 10

Longitud vigas
(180 188 191 133 159 164 123 129 106 136)

Demanda vigas (12 16 1 14 14 10 13 4 11 5)

Tipos perfiles disponibles 5

Longitud perfiles
(1310 3081 2772 1222 3181)

Stock perfiles (39 26 45 7 33)

Número de patrones utilizados (G) 8 **Restos generados** 2

Inventarios finales por viga (I^+ e I^-) = (1, 4, 0, 0, 3, 2, -1, 2, 0, 1)

Stocks finales de perfiles = (36, 26, 43, 6, 31)

Descripción de los patrones
*Patron**1**: vigas que corta (2 0 0 1 1 1 1 1 1 1), perfil usado 1, resto=0;*
*Patrón**2**: vigas que corta (1 1 0 2 1 1 2 0 1 0), perfil usado 1, resto=1;*
*Patrón**3**: vigas que corta (2 0 0 2 2 0 1 0 1 1), perfil usado 1, resto=1;*
*Patrón**4**: vigas que corta (1 4 0 2 4 2 2 2 1 0), perfil usado 3, resto=0;*
*Patrón**5**: vigas que corta (1 4 0 2 4 2 2 2 1 0), perfil usado 3, resto=0;*
*Patrón**6**: vigas que corta (1 2 0 0 2 0 0 0 2 1), perfil usado 4, resto=0;*
*Patrón**7**: vigas que corta (2 6 1 1 2 2 2 1 2 1), perfil usado 5, resto=0;*
*Patrón**8**: vigas que corta (3 3 0 4 1 4 2 0 2 2), perfil usado 5, resto=0;*

Figura 5.10- Representación de un problema de corte y su solución

Una vez ejecutados los algoritmos genéticos correspondientes a las Fases 1 y 2, se ha obtenido una población de soluciones de 100 individuos. Supongamos que elegimos una de las soluciones obtenidas por el genético y que se representa en el cuadro de la derecha de la figura anterior. Esta solución consiste en 8 patrones cortados a partir de distintos perfiles (1, 3, 4 y 5) y que generan un desperdicio total de valor 2, pero que deja por cubrir la demanda de la viga 7 en 1 unidad y tiene una sobreproducción de 13 unidades de diferentes tipos de vigas: una de la viga 1, cuatro de la viga 2, tres de la viga 5,…

Analicemos el caso de la viga del tipo 1 cuya longitud y demanda eran de 180 y 12 respectivamente. Se observa que se han cortado un total de 13 unidades, lo que ha supuesto un exceso de inventario de 1 unidad como refleja el I^+. A la hora de buscar qué patrones han contribuido a la sobreproducción de dicha viga, se comprueba que los 8 patrones de la solución cortan esa viga y por tanto todos ellos son responsables de ese inventario en exceso. Todos han podido contribuir de igual forma a la sobreproducción produciendo una viga de exceso. Repitiendo el proceso para el resto de tipos de vigas se va calculando la contribución total de cada uno de los patrones. El algoritmo elige como primer *patrón incompleto* aquel que contribuye en mayor medida a la sobreproducción, en caso de empate se elige el de menor resto. Una vez se ha repetido el cálculo para los diez tipos de vigas, se obtiene el Patrón 4 como el que más contribuye a la sobreproducción. Por tanto el primer *patrón Incompleto* será el patrón 4, de éste habrá una parte que sí formará parte de las demandas y que habrá que seguir

cortando, pero habrá otra que supondrá inventario en exceso y que podría no cortarse, como muestra la siguiente figura.

Patrón4

original = (1 4 0 2 4 2 2 2 1 0)
*parte que **SÍ** hay que cortar (0 0 0 2 1 0 2 0 1 0)*
*parte que podría **NO** cortarse: Patrón 4^+ =(1 4 0 0 3 2 0 2 0 0)*

Nueva situación de las vigas sobreproducidas:

$I_1^+ = I_0^+$ *- Patrón 4^+= (0 0 0 0 0 0 0 0 0 0 1)*

Patrón3

original = (2 0 0 2 2 0 1 0 1 1)
*parte que **SÍ** hay que cortar (2 0 0 2 2 0 1 0 1 0)*
*parte que podría **NO** cortarse: Patrón 3^+ =(0 0 0 0 0 0 0 0 0 1)*

Figura 5.11- Patrones Incompletos obtenidos para una solución del problema de corte

Por tanto de las 13 vigas del I_0^+ original ya se han identificado 12, quedando por identificar 1 viga del tipo 10 que será el nuevo I_1^+ como se muestra en la figura anterior. Ahora se vuelve a aplicar el algoritmo buscando el siguiente patrón que contribuya en mayor medida a la sobreproducción. En este caso concreto al tratarse de una cantidad en exceso de una unidad, cualquier patrón que corte la viga 10 (patrones 1, 3, 6, 7 y 8) podría ser *patrón incompleto*. El algoritmo elige el de mayor resto que en este caso es el patrón 3. Así pues, de los ocho patrones de la solución tenemos que seis de ellos se cortan completamente para seguir satisfaciendo las demandas y que llamaremos Patrones Completos (1, 2, 5, 6, 7 y 8) y dos patrones de los que sólo se necesita una parte de ellos, Patrones Incompletos (4 y 3). Con esto tendríamos una respuesta al problema de identificación de los *patrones incompletos*. La Solución está compuesta por los seis patrones que sí se van a utilizar de forma completa y por otro u otros patrones que aún están por definir y que se obtendrán a partir de los *patrones incompletos*. El estado de los stocks de perfiles también se ha de actualizar desde el momento en el que dejan de cortarse algunos patrones. La siguiente figura muestra la nueva situación después de aplicar el algoritmo.

Número de patrones completos (*G*) 6 **Restos generados** 1

Inventarios finales por viga (I^+ e I^-) = (0, 0, 0, 0, 0, 0, -1, 0, 0, 0)

Stocks finales de perfiles = (37, 26, 44, 6, 31)

Patrones Completos
*Patron**1**: vigas que corta (2 0 0 1 1 1 1 1 1 1), perfil usado 1, resto=0;*
*Patrón**2**: vigas que corta (1 1 0 2 1 1 2 0 1 0), perfil usado 1, resto=1;*
*Patrón**5**: vigas que corta (1 4 0 2 4 2 2 2 1 0), perfil usado 3, resto=0;*
*Patrón**6**: vigas que corta (1 2 0 0 2 0 0 0 2 1), perfil usado 4, resto=0;*
*Patrón**7**: vigas que corta (2 6 1 1 2 2 2 1 2 1), perfil usado 5, resto=0;*
*Patrón**8**: vigas que corta (3 3 0 4 1 4 2 0 2 2), perfil usado 5, resto=0;*

Patrones Incompletos (perfiles y restos por determinar)
vigas que se han de cortar (0 0 0 2 1 0 2 0 1 0)
vigas que se han de cortar (2 0 0 2 2 0 1 0 1 0)

Figura 5.12- *Patrones completos* e *incompletos* para una solución del problema de corte

El siguiente paso en la definición de una nueva solución, buscará nuevos patrones de corte que **contenga tanto las vigas que quedan pendientes por cortar** (las correspondientes a los Patrones Incompletos) **como las que forman parte de los I^-**. A este problema lo llamaremos **problema residual**. A tal efecto se proponen tres algoritmos diferentes basados en tres estrategias distintas, de esta forma se pretende cubrir el mayor número de posibilidades. Se aplicarán los tres algoritmos al problema residual y de ellos se seleccionará aquel que proporcione una mejor solución.

- **Algoritmo de agrupamiento (A1)**

  Este sencillo procedimiento, agrupará todos los *patrones incompletos* en un único patrón. Su aplicación no tendrá solución en los casos en los que el número de *patrones incompletos* sea elevado.

Algoritmo de agrupamiento

1- *Unificar todas las vigas pendientes de asignar en un único patrón de corte*
2- *Buscar entre los perfiles (K) disponibles aquel que pueda contener todas las vigas de forma que se minimice el desperdicio*

   para $k=1:K$

   si $L_k \geq \sum_{ij} C_{ij}$, *seleccionar k' /* $L_{k'} = min(L_k)$

3- *Actualizar niveles de Stock de perfiles* $P_{k'}=P_{k'}^{0}-1;$
4- *Calcular el nivel de desperdicio del nuevo patrón y el de la nueva solución (resto)*

Para el ejemplo anterior, cuando se aplica el algoritmo de agrupamiento sobre los *patrones incompletos*, obtenemos la siguiente solución

Número de patrones completos (G) 7 **Restos Totales** 700

Inventarios finales por viga (I^+ e I^-) = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0)

Stocks finales de perfiles = (37, 26, 43, 6, 31)

Patrones Completos
*Patron**1**: vigas que corta (2 0 0 1 1 1 1 1 1 1), perfil usado 1, resto=0;*
*Patrón**2**: vigas que corta (1 1 0 2 1 1 2 0 1 0), perfil usado 1, resto=1;*
*Patrón**5**: vigas que corta (1 4 0 2 4 2 2 2 1 0), perfil usado 3, resto=0;*
*Patrón**6**: vigas que corta (1 2 0 0 2 0 0 0 2 1), perfil usado 4, resto=0;*
*Patrón**7**: vigas que corta (2 6 1 1 2 2 2 1 2 1), perfil usado 5, resto=0;*
*Patrón**8**: vigas que corta (3 3 0 4 1 4 2 0 2 2), perfil usado 5, resto=0;*

Nuevo Patrón obtenido por A1:
(2 0 0 4 3 0 4 0 2 0), perfil usado 3, resto= 699;

Figura 5.13- Solución obtenida por algoritmo de agrupamiento

- **Algoritmo genético residual (A2)**

Este procedimiento volverá a aplicar el algoritmo genético descrito en el apartado 5.3.2 al problema residual. A tal efecto y como las demandas y los inventarios de perfiles son diferentes a los originales, se tendrán que determinar nuevos patrones de corte y por tanto se repetirá también el algoritmo genético del apartado 5.3.1.

Este algoritmo no tendrá sentido en el caso en el que el número de *patrones incompletos* esté próximo al número original de patrones de la solución, ya que nos encontraríamos con que el problema residual es prácticamente el mismo que el problema original.

Algoritmo genético residual

1- *Unificar todas las vigas pendientes de asignar en un único patrón de corte. Definir los parámetros del problema de corte* ($l_1,...,l_m,d_1,...,d_m,L_1,\ldots,L_p,P_1,\ldots,P_p$)
2- *Aplicar el algoritmo genético de generación de patrones obteniendo una nueva base de patrones de corte* m_{it}, w_{kt}.
3- *Aplicar el algoritmo genético para la resolución del problema de corte*
4- *Seleccionar la mejor solución y calcular su nivel de desperdicio (resto)*

Para el ejemplo anterior, cuando se aplica el algoritmo genético sobre los patrones Incompletos, obtenemos la misma solución que con el algoritmo de agrupamiento

Número de patrones completos (G) 7 **Restos Totales** 700

Inventarios finales por viga (I^+ e I^-) = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0)

Stocks finales de perfiles = (37, 26, 43, 6, 31)

Patrones Completos
*Patron**1**: vigas que corta (2 0 0 1 1 1 1 1 1 1), perfil usado 1, resto=0;*
*Patrón**2**: vigas que corta (1 1 0 2 1 1 2 0 1 0), perfil usado 1, resto=1;*
*Patrón**5**: vigas que corta (1 4 0 2 4 2 2 2 1 0), perfil usado 3, resto=0;*
*Patrón**6**: vigas que corta (1 2 0 0 2 0 0 0 2 1), perfil usado 4, resto=0;*
*Patrón**7**: vigas que corta (2 6 1 1 2 2 2 1 2 1), perfil usado 5, resto=0;*
*Patrón**8**: vigas que corta (3 3 0 4 1 4 2 0 2 2), perfil usado 5, resto=0;*

Nuevo Patrón obtenido por A2:
(2 0 0 4 3 0 4 0 2 0), perfil usado 3, resto= 699;

Figura 5.14- Solución obtenida por algoritmo genético residual

- **Algoritmo de búsqueda de longitudes menores (A3)**

  En el caso en el que el número de *patrones incompletos* sea elevado, la aplicación de los dos algoritmos anteriores no parece acertada. Nos encontraremos con una situación en la que en cada uno de los patrones originales se han eliminado pocas vigas y por tanto parece indicado buscar, para cada Patrón Incompleto, un nuevo perfil cuya longitud sea menor que el utilizado originalmente y que así reduzca el resto generado por ese patrón.

  El algoritmo mantendrá la estructura de los *patrones incompletos* e intentará buscar para cada uno una longitud de entre las disponibles que reduzca el resto.

  Para soluciones del problema de corte en el que se hayan satisfecho todas las demandas y sólo haya inventarios en exceso (I^-=0 y I^+≠0), la aplicación del algoritmo es sencilla. El problema surge cuando además de inventarios en exceso, sí que hay demandas insatisfechas (I^-≠0 y I^+≠0). La dificultad aquí radica en determinar en cuál o cuáles de los *patrones incompletos* se deben incorporar las vigas del I^-. El algoritmo planteará todas las posibles combinaciones de agrupamiento de las vigas del I^- y seleccionará aquella que consiga el menor desperdicio en uno de los patrones incompletos utilizando unos de los perfiles factibles (en stock y en longitud).

  Supongamos que nos encontramos con un problema en el que el número de *patrones incompletos* de la solución es M, la cantidad total de vigas del I^- es de Q unidades y el

número de perfiles disponibles en stock es K. El algoritmo que proponemos tendrá los siguientes pasos:

Algoritmo de búsqueda de longitudes menores

1- Inicialización

Calcular la longitud total L_m de cada Patrón Incompleto (m), $L_m=\sum_i l_{im}$, donde l_{im} son las longitudes de todas las vigas i cortadas en m.

Determinar el resto (r_{mk}) generado si el patrón m lo realizamos a partir de un perfil k de longitud L_k. $r_{mk} = L_k - L_m$;

2- ***Mientras*** $I^- \neq 0$

Determinar las C posibles combinaciones de agrupar las Q vigas de I^- (agrupadas en grupos desde de 1 hasta de Q elementos).

Calcular para cada combinación (c) la suma de las longitudes vigas que contiene $L_{c=}=\sum_i l_{ic}$

Calcular $T_{mkc}= r_{mk} - L_c$

si $T_{mkc} < 0$, *entonces* $T_{mkc} = 10000$

Seleccionar el menor de los T_{mkc} (en caso de empate se coge cualquiera de ellos)

Asignar las vigas de la combinación c al Patrón Incompleto m.

Actualizar $I^-=I_0^- - l_{ic}$

3- Asignar a cada uno de los nuevos Patrones Incompletos (que ya incorporan la I^-) el perfil k de los disponibles que minimice el resto generado por el patrón.

4- Calcular el resto total generado por la nueva solución

Para el ejemplo anterior, cuando se aplica el algoritmo de agrupamiento sobre los *patrones incompletos*, obtenemos la siguiente solución

Número de patrones completos (G) 8 **Restos Totales** 460

Inventarios finales por viga (I^+ e I^-) = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0)

Stocks finales de perfiles = (36, 26, 44, 5, 31)

Patrones Completos
*Patron**1**: vigas que corta (2 0 0 1 1 1 1 1 1 1), perfil usado 1, resto=0;*
*Patrón**2**: vigas que corta (1 1 0 2 1 1 2 0 1 0), perfil usado 1, resto=1;*
*Patrón**5**: vigas que corta (1 4 0 2 4 2 2 2 1 0), perfil usado 3, resto=0;*
*Patrón**6**: vigas que corta (1 2 0 0 2 0 0 0 2 1), perfil usado 4, resto=0;*
*Patrón**7**: vigas que corta (2 6 1 1 2 2 2 1 2 1), perfil usado 5, resto=0;*
*Patrón**8**: vigas que corta (3 3 0 4 1 4 2 0 2 2), perfil usado 5, resto=0;*

Nuevos Patrones obtenidos por A3:
(0 0 0 2 1 0 2 0 1 0), perfil usado 4, resto= 445;
(2 0 0 2 2 0 2 0 1 0), perfil usado 1, resto= 14;

Figura 5.15- Solución obtenida por algoritmo de longitudes menores

Finalmente, de entre los tres algoritmos propuestos se elige aquel cuyo resto sea menor en el caso del ejemplo anterior, el mejor desempeño se obtiene para el algoritmo A3 con un resto generado total de 460 unidades de longitud frente a los 700 que se obtenían mediante los algoritmos A1 y A2.

5.4 EXPERIMENTACÓN Y RESULTADOS COMPUTACIONALES

En este apartado se tratan aspectos relativos a la evaluación del desempeño de la metodología propuesta así como la idoneidad de su uso en el caso real descrito en el capítulo segundo. No obstante, previa a la implementación de los algoritmos genéticos propuestos en el apartado anterior, se hace necesario desarrollar e implementar un generador de problemas. De forma que a partir de una instancia inicial, se puedan obtener datos de diferentes problemas de test para así evaluar el desempeño de los algoritmos propuestos. Es por eso que en el siguiente apartado se describe el generador de problemas desarrollado que se basa en el que se propone en (Gau, Wäscher 1995). También se tratan los aspectos básicos tenidos en cuenta para su implementación con el software comercial ®*MATLAB* R2007b. Una descripción de las funciones generadas para llevar a cabo dicha implementación se encuentra en el documento Anexo.

5.4.1 Generador de problemas de test

En la literatura (Gau, Wäsher 1995) han desarrollado un generador de problemas para el problema de corte unidimensional con un único perfil en stock (1dimensional SSSCSP). Su limitación radica en que sólo considera perfiles iguales mientras que el problema aquí planteado identifica perfiles en stock de diferentes longitudes. Este hecho hace necesario modificar el generador original así como incrementar el número de parámetros del problema necesarios para contemplar diferentes tipos de perfiles en stock.

Utilizando la notación original de (Gau, Wäscher 1995), los parámetros necesarios cuyos valores serán los datos de la instancia, para nuestro generador de problemas serían:

- m: número de vigas o vigas diferentes demandados
- v_1, v_2: límites inferior y superior para las longitudes de las vigas demandadas
- d: demanda media para cada tipo de viga
- p: número de perfiles estándar en stock
- S: cantidades medias para cada perfil en stock
- s_1,s_2: límites inferior y superior para las longitudes de los perfiles en stock

Por lo tanto, es posible describir un problema de testeo como una variable generada de manera aleatoria y 2m+2p-dimensional: $(l_1,...,l_m,d_1,...,d_m,L_1,...,L_p,P_1,...,P_p)$ donde se especifican las longitudes y demandas de m barras y stock de p perfiles y serán los datos del problema. Las variables se generarán a través de un generador aleatorio de números (*pseudo-random number generador)*.

Las longitudes tanto de las vigas como de los perfiles, se determinan a través de los dos siguientes pasos:

1- Generación de una variable Y uniformemente distribuida entre $(b1,b2+1)$ donde $b1$ y $b2$ representan los límites inferior y superior:

$$y_i = b_1 + \left(b_2 + 1 - b_1\right) * rand_i(0,1) \qquad (5.21)$$

2- Redondeo de ésta hasta el siguiente entero inferior

Las demandas de vigas y el stock de perfiles se determinan de manera que los totales ($D = m*d$) se distribuyan uniformemente entre las diferentes longitudes de acuerdo con lo siguiente (se formula para las vigas, para los perfiles sería equivalente):

$$d_i = \max\{1, [d_i' + 0.5]\}, \tag{5.22}$$

$$d_i' = \frac{rand_i(0,1)}{\sum_{k=1}^{n} rand_k(0,1)} * D, \qquad i = 1,\ldots,m-1, \tag{5.23}$$

$$d_m = \max\left\{1, D - \sum_{i=1}^{m-1} d_i\right\}, \tag{5.24}$$

Las demandas d_i' (excepto la de orden m) se modifican añadiéndoles 0.5 puntos con la finalidad de prevenir una subestimación de éstas.

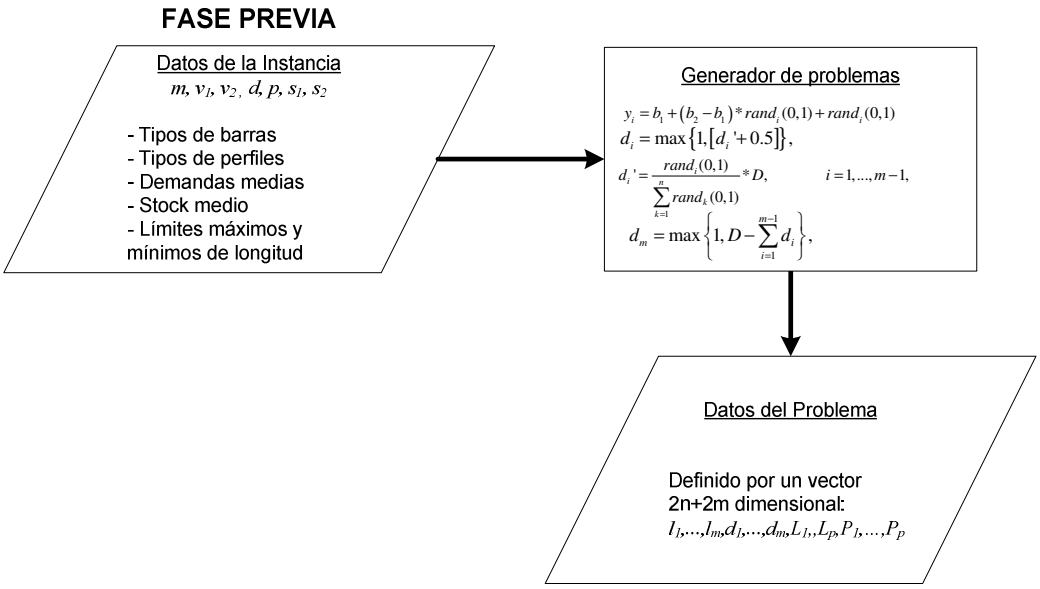

Figura 5.16- Descripción de la Fase previa: Generación de problemas

5.4.2 Generación de las Instancias

Con el fin de evaluar el desempeño de la metodología basada en algoritmos genéticos propuesto en el apartado anterior, se resuelven 15 problemas de test a partir de clases de instancias diferentes. Algunas de estas instancias se han obtenido de la literatura y fueron las utilizadas recientemente en (Poldi, Arenales 2009) en la evaluación de sus métodos

heurísticos para la resolución del problema de corte con múltiples longitudes en stock y bajas demandas. Los parámetros utilizados fueron los siguientes: el número de tipos de perfiles se ha fijado en p ={3 y 5} y los tipos de vigas m = {5, 10 y 20}; las longitudes en stock se generan aleatoriamente entre los valores s_1= 200 y s_2= 1000; el stock medio disponible de cada perfil es de S = 125 (Poldi y Arenales (2009) lo establecen como una distribución uniforme entre 1 y 100*m/2); la longitud para los tipos de vigas estará comprendida entre el intervalo de de v_1, y v_2 que tomará los valores de v_1,=6 y v_2=120, v_1,=6 y v_2=480 y v_1,=60 y v_2=120. La demanda media por viga es de 5 unidades. A partir de estos datos se pueden las generar 21 instancias que se muestran en la siguiente tabla. **De éstas, 18 se usarán** para comparar los resultados obtenidos por el algoritmo propuesto con los que se obtienen por otros métodos de la bibliografía en las tablas 5.4 y 5.5.

Parámetros				
Instancia	*p*	*m*	*v1*	*v2*
1	3	5	6	120
2	3	10	6	120
3	5	10	6	120
4	7	10	6	120
5	3	20	6	120
6	5	20	6	120
7	7	20	6	120
8	3	5	60	480
9	3	10	60	480
10	5	10	60	480
11	7	10	60	480
12	3	20	60	480
13	5	20	60	480
14	7	20	60	480
15	3	5	6	480
16	3	10	6	480
17	5	10	6	480
18	7	10	6	480
19	3	20	6	480
20	5	20	6	480
21	7	20	6	480

Tabla 5.2- Instancias para la experimentación

5.4.3 Evaluación del desempeño del algoritmo genético generador de patrones

De acuerdo con (Haesler y Sweeny 1991), una de las claves del éxito de un procedimiento heurístico aplicado al problema de corte es la eficiencia de los patrones de corte que constituyen la base inicial del problema. En la sección 5.2.1 se ha descrito un generador de problemas basado en algoritmos genéticos. En este apartado se evalua el desempeño de dicho algoritmo para la generación de patrones de corte eficientes (mínimo nivel de restos).

Los parámetros de entrada sobre los que se evaluó el **generador de patrones** fueron: un tamaño de la población de 50 individuos, un número de 100 generaciones (iteraciones) realizadas por el algoritmo, 10 individuos como élite, como en (Anand et al. 1999) la fracción de cruce se estableció en 0.8, la función de eficiencia utilizada para la evaluación de los patrones ha sido la de la expresión (5.13). En el siguiente gráfico (figura 5.17) se muestran los resultados medios obtenidos por el generador de patrones en las instancias de baja demanda de la tabla 5.2 (15 problemas por cada instancia). Cada grupo de porcentaje (0%, 0.1%-2%, 2.1%-6%, 6.1%-12%, 12.1%-19%, 19.1%-25%, 25.1%-31%, 31.1%-100%), muestra cómo se reparten en función del resto los patrones generados por el algoritmo.

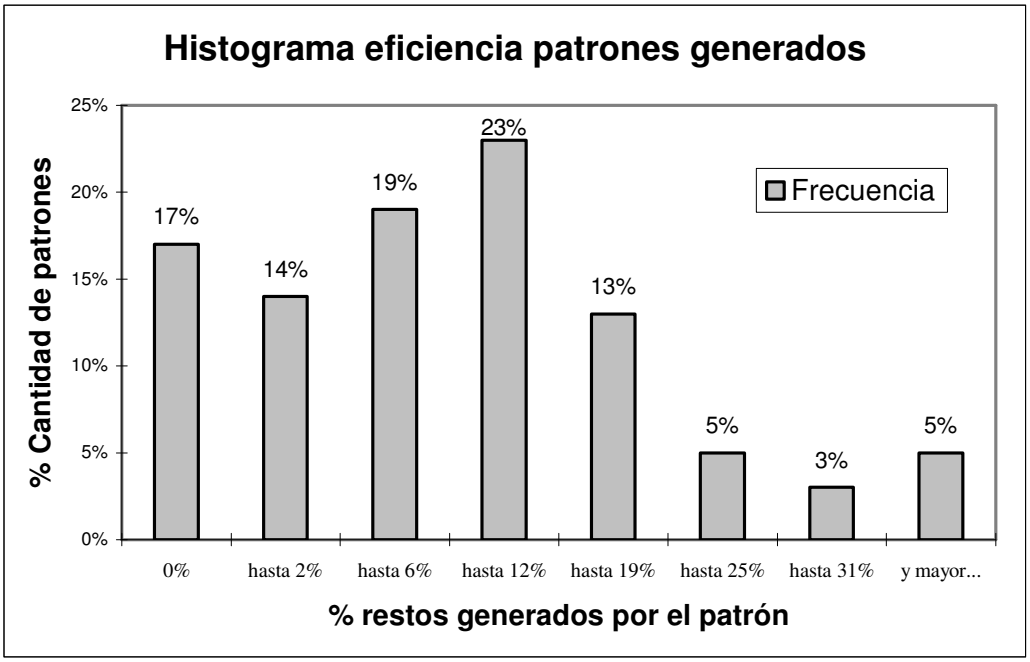

Figura 5.17- Histograma distribución patrones

Como promedio, el 17% de los patrones generados en cada problema es de resto nulo y el 19% con un resto máximo del 2%. Es decir, el 36% de los patrones obtenidos por el generador tienen un desperdicio de material inferior al 2%. Es evidente por tanto, que el generador es capaz de obtener patrones eficientes y además en cantidad suficiente como para garantizar que la solución obtenida en la resolución del problema de corte sea buena.

5.4.4 Análisis de sensibilidad de los parámetros del algoritmo del problema de corte y valores óptimos

Previa a la aplicación de la metodología propuesta, se considera interesante realizar un análisis de la sensibilidad del algoritmo genético del problema de corte (5.3.2) a la fracción de cruce y al $w_{inventario}$ de la expresión (5.18) y comprobar en qué medida influye la variación de estos parámetros sobre el desempeño de éste y así poder determinar el valor apropiado de cada parámetro en la aplicación de la metodología.

A tal efecto, se compararán los resultados obtenidos por los algoritmos de generación de patrones y del problema de corte (5.2.1 y 5.2.2) sobre diferentes instancias. Se tomarán valores de fracción de cruce de 0.4, y 0.8 y valores de $w_{inventario}$ de 1, 3 y 10 y w_{resto} y $w_{insatisfechas}$ fijados en valores de 1 y 10 respectivamente. El objetivo de analizar la sensibilidad a la tasa de cruce es para evaluar el efecto sobre la diversidad en las soluciones obtenidas al incrementar el número de mutaciones en cada iteración del algoritmo. Por otro lado, el aumento de de $w_{inventario}$ penalizará más el exceso de producción que los restos generados pero también puede hacer que el algoritmo incurra en la no satisfacción total de la demanda. Los indicadores del desempeño del algoritmo en cada caso serán: número de soluciones obtenidas que satisfagan todas las demandas, de entre éstas, la cantidad mínima de vigas sobreproducidas por la mejor de las soluciones y los restos generados para ese plan de fabricación (con sobreproducción). En la siguiente tabla 5.3 se muestran los resultados medios obtenidos al modificar los parámetros $w_{inventario}$ y T_{cruce} **en 10 problemas generados a partir de 2 instancias** diferentes ([10,6,120,5,3,200,1000,125] y [5,6,120,5,3,200,1000,125]) .

Instancia $[m, v_1, v_2, d, p, s_1, s_2, S]$	*Tcruce*	W_{inv}	*Numero de soluciones demandas satisfechas*	*Cantidad mínima sobreproducción*	*Restos generados mejor solución*
[10,6,120,5,3,200,1000,125]	0.8	1	25,9	11,2	5,5
[10,6,120,5,3,200,1000,125]	0.4	1	27,2	17,8	26,4
[10,6,120,5,3,200,1000,125]	0.8	3	23,6	10,9	6,6
[10,6,120,5,3,200,1000,125]	0.4	3	28,4	14,6	58,3
[10,6,120,5,3,200,1000,125]	0.8	10	15,9	12,8	37,2
[10,6,120,5,3,200,1000,125]	0.4	10	15,1	18,5	91,7
[5,6,120,5,3,200,1000,125]	0.8	1	46,1	4,4	9,2
[5,6,120,5,3,200,1000,125]	0.4	1	54,1	6,1	23,3
[5,6,120,5,3,200,1000,125]	0.8	3	38,6	3,4	14,9
[5,6,120,5,3,200,1000,125]	0.4	3	37,6	3	9,5
[5,6,120,5,3,200,1000,125]	0.8	10	27,1	2,1	6,7
[5,6,120,5,3,200,1000,125]	0.4	10	36,3	2,9	8,9

Tabla 5.3- Sensibilidad del algoritmo

Como se puede observar en todas ellas se obtiene un elevado número de soluciones con demandas satisfechas. Al reducir la tasa de cruce de 0.8 a 0.4, aunque hay un ligero incremento en el número de soluciones que satisfacen las demandas, se observa un aumento en la cantidad de inventario final así como un incremento significativo en el valor del resto generado. En cuanto a la sensibilidad del parámetro $w_{inventario}$ se comprueba que al aumentar su valor, aunque apenas varía el inventario en exceso, los restos generados sí se incrementan considerablemente. Notar que para todas las instancias se obtienen soluciones con niveles bajos de desperdicio

Mediante el cálculo del coeficiente de correlación de *Pearson*, que mide la relación lineal existente entre dos variables, se puede cuantificar el grado de sensibilidad de las soluciones a la variación de los parámetros (T_{cruce} y $W_{inventario}$). El cálculo del coeficiente de correlación para dos variables cualesquiera x e y, se realiza dividiendo la covarianza (σ_{xy}) entre el producto de las desviaciones estándar de ambas variables (σ_x y σ_y). En primer lugar se obtiene la correlación promedio entre la tasa de cruce y las variables resultado (cantidad minima de sobreproducción, número de soluciones con todas las demandas satisfechas y restos generados por la mejor solución) para un mismo valor de $W_{inventario}$. En la tabla 5.4 se muestran los resultados obtenidos para los datos de la tabla 5.3.

Variables relacionadas	Coeficiente Pearson
Tcruce / Cantidad mínima sobreproducción	-0,26685041
Tcruce / Numero de soluciones demandas satisfechas	-0,1944211
Tcruce / Restos generados mejor solución	-0,6489404

Tabla 5.4- Coeficientes de correlación de Pearson

Se observa que en los tres casos la correlación es negativa (un aumento en la tasa de cruce provocará una disminución en las tres variables) siendo más fuerte en el caso de los restos generados y más débil en el caso del número de soluciones con todas las demandas satisfechas. Un aumento en la tasa de cruce por tanto hace disminuir los restos generados por la mejor solución y hace disminuir también la cantidad mínima de sobreproducción, ambas circunstancias son deseables. Por otro lado, se produce una disminución en el número de soluciones con todas las demandas satisfechas, lo que no es deseable, si bien es cierto que en un grado atenuado (-0,19). Por tanto, entre los dos valores de tasa de cruce considerados (0,4 y 0,8), y a la vista de los resultados, parece razonable **optar por el valor de 0,8 como tasa de cruce**. En la siguiente tabla se indica el coeficiente de correlación obtenido entre la $w_{inventario}$ y las variables de resultado con una tasa de cruce de 0,8.

Variables relacionadas	Coeficiente Pearson
$W_{inventario}$ / Cantidad mínima sobreproducción	-0,01710628
$W_{inventario}$ / Numero de soluciones demandas satisfechas	-0,54228093
$W_{inventario}$ / Restos generados mejor solución	0,45635913

Tabla 5.5- Coeficientes de correlación de Pearson

En este caso se observa que una disminución en la $w_{inventario}$ aumenta el resto generado y un aumento en la $w_{inventario}$ disminuye el número de soluciones con todas las demandas satisfechas. Sin embargo, en contra de lo que podía esperarse a priori, la cantidad mínima de sobreproducción apenas se ve afectada. Así pues, entre los tres valores considerados (1, 3 y 10) **se elige 3 como valor de $w_{inventario}$.**

5.4.5 Resultados

Los parámetros de entrada para el **generador de patrones** (apartado 5.3.1) fueron: un tamaño de la población de 50 individuos, un número de 100 generaciones (iteraciones) realizadas por el algoritmo, 10 individuos como élite y la fracción de cruce se estableció en 0.8, la función de eficiencia utilizada para la evaluación de los patrones ha sido la de la expresión (5.13). Como se indicaba en el apartado 5.3.1, en los casos especiales en los que el número de combinaciones posibles es inferior a 250, se ha realizado la enumeración completa de todas las posibilidades y posteriormente se han seleccionado las 12 mejores que entrarán a formar parte de la base de patrones a introducir en el genético de corte.

En el caso del algoritmo genético para resolver el problema de corte se ha utilizado una población de 100 individuos, un número máximo de 500 iteraciones, una tasa de cruce de 0.8 y la función de eficiencia de la expresión (5.18) con una de $w_{inventario}$ de 3, w_{resto} de 1 y $w_{insatisfechas}$ de 10. Los algoritmos para la mejora de las soluciones (apartado 5.3.3) se han aplicado a las 20 mejores soluciones obtenidas por el algoritmo del apartado 5.3.2. Al igual que en (Poldi, Arenales 2009), los resultados obtenidos se comparan con los siguientes procedimientos heurísticos:

- **Heurísticas Constructivas**

  Consisten en ir construyendo la solución. Primero se selecciona el patrón de corte más eficiente de los generados (el de resto mínimo) y se utiliza tanto como sea posible sin sobrepasar las demandas y las disponibilidades de stock en almacén. Si las demandas no están satisfechas completamente se sigue con el siguiente patrón con mejor valor de eficiencia. Dentro de las heurísticas constructivas están:

  - **Heurística FFD** (*First Fit Decrease*) para generar el patrón de corte (uno para cada perfil disponible): se selecciona primero la viga más grande y se incorpora al patrón tantas veces como sea posible, luego se pasa a la siguiente viga en tamaño y así sucesivamente (de esta manera se garantiza que las vigas más grandes que son las más complicadas de combinar sean las primeras en asignar)
  - **Heurística *Greedy*** para generar el patrón de corte: se resuelve el problema de la mochila asociado al problema dual del problema de corte (uno para cada perfil), se selecciona el patrón que menos restos genere.

- **Heurísticas Residuales**

  Consisten en generar la solución al problema lineal continuo. Redondear a una solución entera (normalmente redondeo inferior), actualizar las demandas y las que aún están pendientes de satisfacer (problema residual) se les aplica alguna heurística. En concreto se cogen:

  - **Heurística FFD**: se aplica el FFD para el problema residual
  - **Heurística Greedy** para el problema residual

En la siguiente tabla 5.6 se muestran los resultados obtenidos al aplicar la metodología propuesta para las instancias de baja demanda, en restos totales medios de la mejor solución de cada uno de los 15 problemas resueltos para cada instancia, así como los resultados obtenidos por las otras heurísticas, (Poldi, Arenales 2009), para esas mismas instancias. En negrita aparece destacado el procedimiento con mejor desempeño para tipo de instancia. Comparando los resultados obtenidos por la aplicación de la metodología propuesta se observa en las instancias 4, 5 y 6 (20 tipos de vigas y longitudes pequeñas de las vigas) las heurísticas constructivas y residuales tienen mejor desempeño con un valor de resto total generado de entre la mitad y la tercera parte. Sin embargo se observa que para el resto de las 15 instancias se obtienen resultados considerablemente mejores mediante la aplicación de los algoritmos genéticos y de los algoritmos A1, A2, A3. En especial en las instancias de la 8 a la 18 en el que las vigas tienen longitudes mayores (algunas incluso mayores que algún perfil) y por tanto es más complicada la asignación de vigas a un patrón.

En la tabla 5.6 se observa cierta correlación entre la relación de longitudes viga/perfil y los restos generados. Esto se pone claramente de manifiesto en el caso de las heurísticas residuales y constructivas del tipo *First Ffit* y *Greedy* en la que el efecto se amplifica considerablemente Para las primeras instancias en las que las vigas, comparadas con los perfiles, son de longitudes pequeñas, los restos generados son reducidos. Sin embargo, en las instancias desde la 8 a la 14 con valores de $v1$ y $v2$ de 60 y 480 respectivamente (ninguna viga de longitud reducida respecto a los perfiles) el nivel de resto se dispara a valores muy elevados. En las instancias de la 15 a la 21, en el que los valores de $v1$ y $v2$ son de 6 y 480 (alguna de las vigas puede ser pequeña en relación con el tamaño del perfil) el nivel de resto disminuye respecto a las anteriores. Por tanto se observa que las heurísticas

constructivas y residuales tipo *First Fit* y *Greedy* son muy sensibles a la relación de longitudes viga/perfil.

Parámetros					*Restos totales*				
					Heurísticas Constructivas		*Heurísticas Residuales*		*Genéticos*
Instancia	*p*	*m*	*v2*	*V2*	*FFD*	*Greedy*	*FFD*	*Greedy*	*A1,A2,A3*
1	3	5	6	120	118,75	113,35	123,25	116,25	**85,40**
3	5	10	6	120	161,05	152,30	146,80	165,20	**27,00**
4	7	10	6	120	179,45	146,05	147,95	142,75	**11,60**
5	3	20	6	120	171,95	170,05	168,50	**161,40**	441,20
6	5	20	6	120	163,80	173,55	202,20	**156,65**	402,60
7	7	20	6	120	199,05	190,30	179,55	**165,20**	310,80
8	3	5	60	480	71319,35	31493,55	36648,90	46662,30	**257,00**
10	5	10	60	480	96475,80	101711,50	57310,90	60145,25	**339,60**
11	7	10	60	480	118647,90	134773,10	52624,25	51184,85	**476,40**
12	3	20	60	480	84775,60	91417,50	49837,50	52420,85	**1083,00**
13	5	20	60	480	98766,15	103892,25	73922,95	72334,80	**647,40**
14	7	20	60	480	136827,20	125865,15	50205,55	42826,80	**683,40**
15	3	5	6	480	42429,35	52808,85	7551,20	8762,75	**594,68**
17	5	10	6	480	74434,35	73920,20	45636,20	47154,75	**540,00**
18	7	10	6	480	87379,30	107851,45	47095,25	35678,10	**303,20**
19	3	20	6	480	53078,50	49565,05	21391,70	24936,65	**694,00**
20	5	20	6	480	81946,20	92632,65	60265,55	57527,50	**828,50**
21	7	20	6	480	92130,75	91638,20	42535,50	39284,45	**242,00**

Tabla 5.6- Resultados obtenidos para Instancias de bajas demandas

En la siguiente tabla (5.7) se comparan los resultados obtenidos, con los procedimientos propuestos recientemente en (Poldi, Arenales 2009) con sus heurísticas residuales RGH1, RGH2 y RGH3. Al igual que en la tabla anterior, se destacan en negrita los mejores resultados para cada instancia. Se observa claramente que en las instancias con un número de tipos de vigas bajo (5) la metodología propuesta alcanza mejores resultados que las heurísticas residuales de Poldi y Arenales. En el caso de 10 tipos de vigas diferentes y con tipos diferentes de perfiles 3 y 5 para los algoritmos genéticos tienen un mejor desempeño mientras. Sin embargo para valores de 20 tipos de vigas diferentes o de 10 con longitudes altas, las heurísticas residuales tienen mejor desempeño.

Parámetros					Restos totales			
					Heurísticas Residuales			Genéticos
Instancia	p	m	*v1*	*v2*	RGH1	RGH2	RGH3	A1,A2,A3
1	3	5	6	120	113,05	114,40	108,7	**85,40**
3	5	10	6	120	137,85	129,20	138,60	**27,00**
4	7	10	6	120	117,65	116,05	115,90	**11,60**
5	3	20	6	120	**155,65**	158,50	161,05	441,20
6	5	20	6	120	150,55	**147,60**	149,80	402,60
7	7	20	6	120	142,55	**126,00**	132,85	310,80
8	3	5	60	480	625,55	646,85	617,20	**257,00**
10	5	10	60	480	785,10	808,40	812,75	**339,60**
11	7	10	60	480	247,75	255,15	**247,65**	476,40
12	3	20	60	480	402,65	375,40	**361,40**	1083,00
13	5	20	60	480	207,35	**192,90**	195,90	647,40
14	7	20	60	480	**153,65**	163,95	177,70	683,40
15	3	5	6	480	709,05	683,05	709,30	**594,68**
17	5	10	6	480	545,00	552,85	576,30	**540,00**
18	7	10	6	480	**241,35**	258,50	247,90	303,20
19	3	20	6	480	423,60	402,15	**392,60**	694,00
20	5	20	6	480	201,35	**172,05**	194,20	828,50
21	7	20	6	480	135,95	140,85	**132,60**	242,00

Tabla 5.7- Resultados obtenidos para Instancias de bajas demandas

En la tabla 5.8 se muestra una comparativa de los ocho métodos en la que cada uno de los valores indicados representa el alejamiento (acercamiento) porcentual respecto del valor mínimo (R_{iB}) para cada instancia y de acuerdo con la expresión (5.25). En la última fila se muestra el porcentaje de veces que cada uno de los métodos ha alcanzado el valor mínimo calculado a partir de la expresión (5.27) y que también se representan en la figura.

$$\widehat{R_{ij}} = \frac{R_{ij} - R_{iB}}{R_{ij}} * 100 \tag{5.25}$$

$$R_{iB} = Min\left\{R_{ij}\right\} \quad i = 1,...,18 \quad \text{y} \quad j = 1,...,8 \tag{5.26}$$

donde R_{ij} es el valor de desperdicio total obtenido para la instancia i por el método j.

$$\widehat{\overline{R_j}} = \frac{\sum_{i=1}^{18} r_{ij}}{18} * 100 \qquad \text{donde } r_{ij} \begin{cases} 1 & \text{si } R_{ij} = 0 \\ 0 & \text{resto de los casos} \end{cases} \tag{5.27}$$

Parámetros					Restos totales							
					Constructivas		Residuales		Heurísticas Residuales			Genéticos
Instancia	p	M	*v1*	*v2*	FFD	Greedy	FFD	Greedy	RGH1	RGH2	RGH3	A1,A2,A3,A4
1	3	5	6	120	28,1%	24,7%	30,7%	26,5%	24,5%	25,3%	21,4%	**0,0%**
3	5	10	6	120	83,2%	82,3%	81,6%	83,7%	80,4%	79,1%	80,5%	**0,0%**
4	7	10	6	120	93,5%	92,1%	92,2%	91,9%	90,1%	90,0%	90,0%	**0,0%**
5	3	20	6	120	9,5%	8,5%	7,6%	3,6%	**0,0%**	1,8%	3,4%	64,7%
6	5	20	6	120	9,9%	15,0%	27,0%	5,8%	2,0%	**0,0%**	1,5%	63,3%
7	7	20	6	120	36,7%	33,8%	29,8%	23,7%	11,6%	**0,0%**	5,2%	59,5%
8	3	5	60	480	99,6%	99,2%	99,3%	99,4%	58,9%	60,3%	58,4%	**0,0%**
10	5	10	60	480	99,6%	99,7%	99,4%	99,4%	56,7%	58,0%	58,2%	**0,0%**
11	7	10	60	480	99,8%	99,8%	99,5%	99,5%	0,0%	2,9%	**0,0%**	48,0%
12	3	20	60	480	99,6%	99,6%	99,3%	99,3%	10,2%	3,7%	**0,0%**	66,6%
13	5	20	60	480	99,8%	99,8%	99,7%	99,7%	7,0%	**0,0%**	1,5%	70,2%
14	7	20	60	480	99,9%	99,9%	99,7%	99,6%	**0,0%**	6,3%	13,5%	77,5%
15	3	5	6	480	98,6%	98,9%	92,1%	93,2%	16,1%	12,9%	16,2%	**0,0%**
17	5	10	6	480	99,3%	99,3%	98,8%	98,9%	0,9%	2,3%	6,3%	**0,0%**
18	7	10	6	480	99,7%	99,8%	99,5%	99,3%	**0,0%**	6,6%	2,6%	20,4%
19	3	20	6	480	99,3%	99,2%	98,2%	98,4%	7,3%	2,4%	**0,0%**	43,4%
20	5	20	6	480	99,8%	99,8%	99,7%	99,7%	14,6%	**0,0%**	11,4%	79,2%
21	7	20	6	480	99,9%	99,9%	99,7%	99,7%	2,5%	5,9%	**0,0%**	45,2%
Veces mejor solución (%)					0,0%	0,0%	0,0%	0,0%	16,7%	22,2%	22,2%	**38,9%**

Tabla 5.8- Resultados porcentuales obtenidos para Instancias de bajas demandas

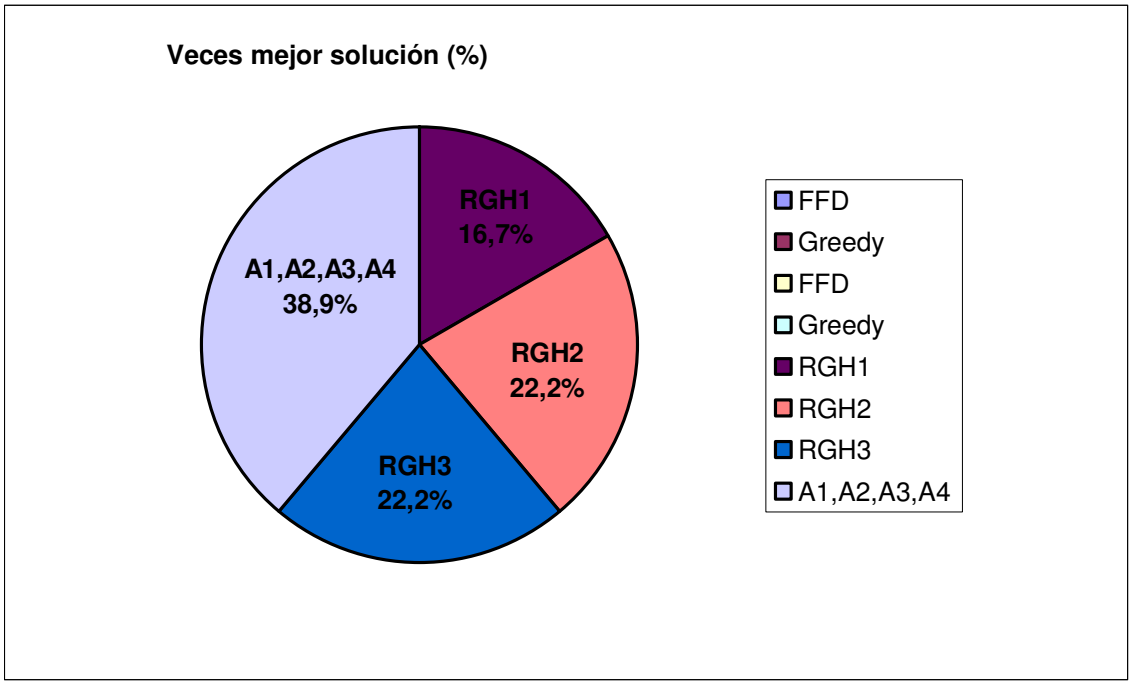

Figura 5.18- Porcentaje de veces que cada método alcanza mejor solución

Como se observa en la Figura 5.18, el uso de la metodología propuesta es adecuado en un 39% de los casos. En concreto y de acuerdo con los resultados obtenidos en las tablas 5.7 y 5.8, para instancias de niveles de demanda bajo, sí parece adecuada la metodología propuesta en el caso en el que el número de vigas no sea muy elevado (5 y 10).

5.4.6 Análisis del impacto de cada una de las fases de la metodología sobre su desempeño

En el apartado anterior se ha constatado cómo el uso de la metodología propuesta resulta adecuado en un porcentaje elevado de las instancias de baja demanda. En concreto, es en las instancias: 1, 3, 4, 8, 10, 15 y 17 en las que se ha obtenido un mejor desempeño. El objetivo ahora, se centrará en evaluar en qué medida cada una de las fases del procedimiento: algoritmo de generación de patrones, algoritmo de corte, y algoritmos de mejora, contribuyen a la consecución de dicho desempeño. En el apartado 5.4.3 se ha comprobado cómo el generador de patrones propuesto, realmente permite disponer de una base de patrones lo suficientemente diversa y eficiente, lo que según (Haesler y Sweeny 1991) es una de las condiciones necesarias para la obtención de buenas soluciones. Pues bien, en este apartado se analizará en qué medida la eficiencia de dichos patrones

contribuye a la obtención de soluciones de bajo nivel de desperdicio, o si por el contrario los algoritmos de mejora del apartado 5.3.3 son capaces de generar buenas soluciones, independientemente de la base de patrones de partida.

A tal efecto, se aplica la metodología a los problemas de test de algunas de las instancias de mejor desempeño (1, 3, 4, 8), utilizando en este caso patrones con peor eficiencia que los obtenidos por el algoritmo genético del apartado 5.3.1. El éxito en la obtención de patrones eficientes por parte del algoritmo genético es atribuible a dos motivos. Por un lado, al procedimiento utilizado para la generación de la población inicial: el valor N_k que de alguna manera garantizará que los perfiles no estén infrautilizados en exceso, y el tamaño de la población (42 individuos por perfil) que permite una dispersión suficiente como para poder encontrar algunos patrones con resto reducido. Y por otro lado, a las operaciones genéticas y a su naturaleza iterativa. Así pues a la hora de analizar el efecto de la calidad de los patrones en la calidad de las soluciones del problema, se analiza cómo responde la metodología si en lugar de aplicar el algoritmo genético, se utilizan los dos grupos siguientes de patrones:

- Patrones generados mediante el mismo procedimiento que la población inicial del algoritmo de 5.3.1: con un número de componentes N_k e igual población de individuos (42 por perfil). Pero sobre los que no se realiza ninguna iteración. A estos patrones se les denomina *patrones de eficiencia media.*

- Patrones también con N_k componentes pero con valores peores de eficiencia que los anteriores y menor población (12 individuos por perfil). A estos patrones se les denomina *patrones de baja eficiencia*

En la figura 5.19 se comparan los histogramas de cada grupo de patrones: patrones del algoritmo genético, patrones de eficiencia media y patrones de baja eficiencia. Se observa claramente el nivel decreciente entre series de los valores de eficiencia.

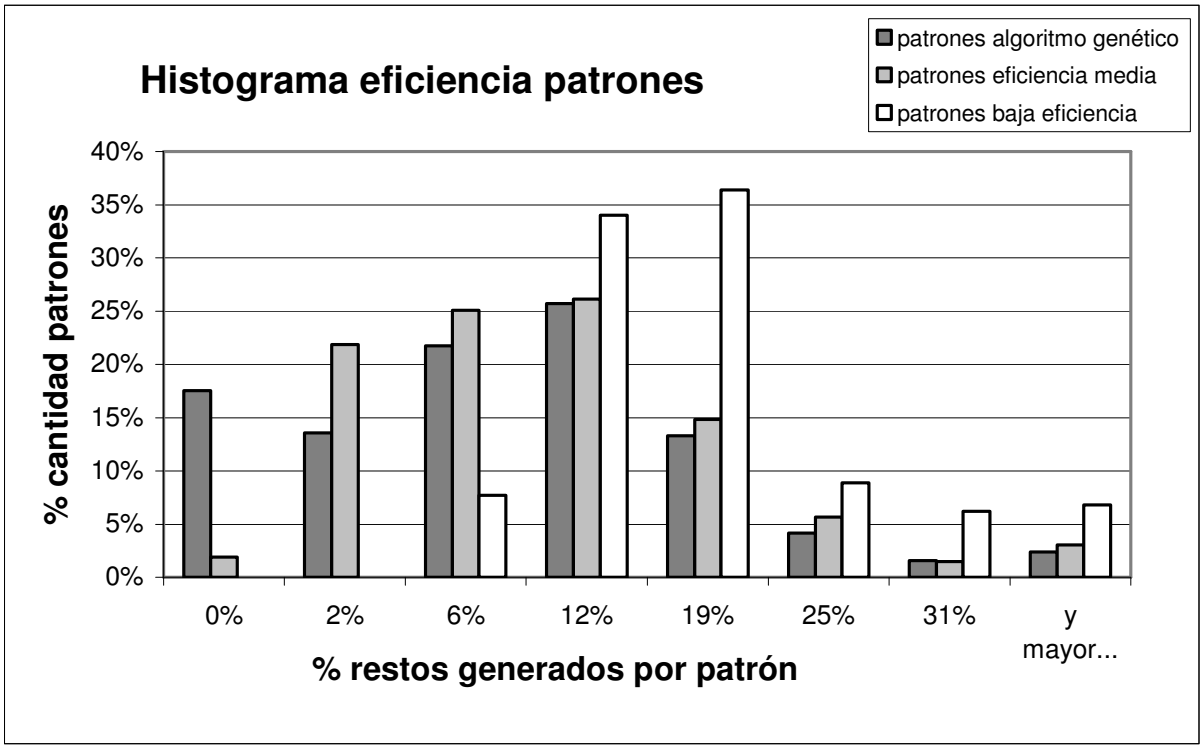

Figura 5.19- Histograma de eficiencia patrones

En la tabla 5.9 se comparan los resultados medios (sobre 15 problemas de test) obtenidos para cada una de las instancias en cada caso: aplicando la metodología íntegramente; y prescindiendo del algoritmo de generación de patrones con patrones de eficiencia media y eficiencia baja. Se indican los restos totales medios para la mejor solución de cada uno de los problemas así como el momento de primer orden respecto al valor mínimo de las otras 19 soluciones generadas por problema.

	Patrones algoritmo genético		**Patrones eficiencia media**		**Patrones baja eficiencia**	
	Resto medio	Momento de primer orden respecto al mínimo	Resto medio	Momento de primer orden respecto al mínimo	Resto medio	Momento de primer orden respecto al mínimo
Instancia 1	**85,40**	**114,58**	**85,40**	**164,91**	**116,45**	**100,44**
Instancia 3	**27,00**	**212,75**	**50,20**	**150,75**	**103,78**	**159,06**
Instancia 4	**11,60**	**78,06**	**33,60**	**128,05**	**43,05**	**75,93**
Instancia 8	**257,00**	**1654,00**	**462,60**	**1736,70**	**1378,05**	**2999,87**

Tabla 5.9- Resultados obtenidos para patrones de distintos niveles de eficiencia

En el caso de los patrones de baja eficiencia, se observa un incremento respecto a los valores obtenidos con el algoritmo genético. Este incremento no se produce en la misma proporción en todas las instancias. En el caso de los patrones de eficiencia media también se observa un incremento respecto al los valores mínimos obtenidos por el algoritmo genético, a excepción de en la Instancia 1, pero siempre menor que en el caso de los patrones de baja eficiencia. Así pues, podemos afirmar que sí que hay **cierta relación directa entre la calidad de los patrones y la calidad de las soluciones**. Cuanto peor es la calidad de los patrones peor es la solución obtenida y por tanto parece apropiado el uso del algoritmo genético del apartado 5.3.1, si bien **esta relación no se manifiesta con la misma intensidad en todas las instancias**.

Por otro lado, hay identificar la causa de las diferencias entre instancias (no todas reaccionan igual ante patrones de peor calidad). Desde el punto de vista morfológico, la diferencia principal entre las soluciones de unas y otras instancias radica en la cantidad de elementos (patrones) que poseen, cantidad que depende de los valores obtenidos en la expresión (5.17). En la tabla 5.10 se indican los valores medios del número de patrones (no nulos) de las mejores soluciones para cada tipo de instancia y utilizando cada uno de los tipos de patrones. Se observa claramente cómo el número de patrones medio por solución varía entre instancias, por lo que es probable que una de las causas de las diferencias a las que antes se apuntaba, sea debida a la variable "cantidad de patrones de la solución". Por otro lado, otro de los aspectos a tener en cuenta y que puede verse afectado en función de la calidad de los patrones, es el grado de contribución a la solución por parte de los algoritmos de mejora (Fase 3). En la tabla 5.10 se indica también qué proporción de los patrones que contiene cada solución ha sido generada en la última fase del procedimiento (algoritmos de mejora).

	Patrones algoritmo genético		**Patrones eficiencia media**		**Patrones baja eficiencia**	
	Cantidad media patrones en solución	**% Patrones generados en la Fase 3**	**Cantidad media patrones en solución**	**% Patrones generados en la Fase 3**	**Cantidad media patrones en solución**	**% Patrones generados en la Fase 3**
Instancia 1	2,00	50%	2,00	50%	2,00	50%
Instancia 3	4,75	27%	4,75	27%	4,75	39%
Instancia 4	6,40	18%	5,60	27%	5,00	34%
Instancia 8	6,50	27%	6,50	34%	6,00	41%

Tabla 5.10- Cantidad de patrones contenidos en solución y contribución fase 3

Así pues nos encontramos con dos variables que caracterizan una solución: “cantidad media de patrones de la solución” y “contribución de los algoritmos de mejora”, que pueden estar relacionadas con el nivel de resto obtenido en función del tipo de patrón.

En la tabla 5.11 se indican los coeficientes de correlación de Pearson entre las variables obtenidos en cada caso (patrones del algoritmo, patrones de eficiencia media y patrones de baja eficiencia)

	Patrones algoritmo genético	Patrones eficiencia media	Patrones baja eficiencia
Coeficiente correlación entre: Cantidad de patrones en la solución /Restos Obtenidos	**0,24**	**0,53**	**0,58**
Coeficiente correlación entre: Cantidad de patrones en la solución /Contribución Fase 3	**-0,93**	**-0,79**	**-0,77**

Tabla 5.11- Coeficientes de correlación por grupos de patrones

Atendiendo únicamente a los patrones del algoritmo genético, se obtiene una relación directa de 0,24 entre los restos obtenidos y la cantidad de patrones de la solución (a más cantidad de patrones, mayores restos). Sin embargo, en el caso de usar patrones menos eficientes se observa que esta correlación se incrementa entorno a valores de 0,53 y 0,58, lo que viene a corroborar lo que se afirmaba más arriba sobre las diferencias entre instancias: **cuantos más patrones contiene la solución, peores desempeños obtienen los patrones de baja eficiencia**. Estos coeficientes de 0,53 y 0,58 deberían ser incluso mayores ya que, como se verá a continuación, los algoritmos de la fase 3 tienen un efecto atenuador sobre la mala calidad de las soluciones obtenidas por patrones de eficiencia media y baja. A medida que aumenta la cantidad de patrones en la solución, se observa que disminuye el grado de contribución de los algoritmos de mejora (coeficientes de correlación negativos). En el caso de los generados por el algoritmo genético, se observa un coeficiente de -0,93. Sin embargo, en los patrones de menor eficiencia este coeficiente cae a valores de -0,77 y -0,79, lo que se traduce en que a medida que aumenta la cantidad de patrones en la solución, la contribución de los algoritmos de mejora es mayor para patrones con peor eficiencia. Por tanto, los algoritmos de la fase 3 mejoran un poco la mala calidad de las soluciones para patrones de baja y media eficiencia.

En definitiva, **una base eficiente de patrones se hace más necesaria a medida que aumenta el número de patrones de la solución**, aunque la contribución de los algoritmos de mejora sea mayor para patrones de baja y media eficiencia, ésta no es suficiente como para neutralizar los efectos de utilizar dichos patrones.

5.4.7 Coste computacional

Si bien el problema del corte de perfiles es del tipo *off-line* y por tanto el coste computacional no supone un aspecto crítico para el algoritmo, en la tabla 5.12 se muestran los tiempos medios de computación (sobre el total de los problemas test generados) para cada una de las fases de la metodología, el tiempo medio total y el tiempo medio por iteración para los algoritmos: genético de generación de patrones, genético de corte y genético residual. Los algoritmos se han implementado con el software comercial *®MATLAB* en su versión *2007b* sobre un procesador *®Intel Core 2 Duo @2.00GHz*.

	tiempo (s)	**tiempo por iteración (ms)**
Fase Previa- Generación Problemas	**0,1218**	
Fase 1- Algoritmo generación de patrones	**3,3560**	33,5598
Fase 2- Algoritmo genético problema corte	**7,6435**	15,2871
Fase 3.1- Patrones Incompletos	0,6384	
Fase 3.2- Algoritmo A1	0,0037	
Fase 3.2- Algoritmo A2	171,4699	342,9399
Fase 3.2- Algoritmo A3	0,2859	
Fase 3.3- Comparación Algoritmos	0,0026	
Total Fase 3- Algoritmos mejora solución	**135,8562**	
Total	**146,9775**	

Tabla 5.12- Costes computacionales

Como se puede observar, la ejecución completa de la metodología es muy rápida no superando por término medio los 150 segundos, esto supone un valor añadido al uso de algoritmos genéticos para la resolución del problema de corte de vigas. Se observa que todos los algoritmos tienen un coste computacional bajo, excepto el algoritmo genético residual que tiene un tiempo mayor de ejecución justificado por el hecho de que aplica,

sobre veinte soluciones, el algoritmo genético de corte (Fase 2). Además, en aquellas soluciones con pocos patrones, el número de combinaciones puede ser menor que el establecido para ejecutar el algoritmo general y por tanto se evalúan todas las combinaciones

5.4.8 Aplicabilidad de la metodología al caso real

En el segundo capítulo, en el apartado 2.3 se especificaban las características principales del problema real objeto de estudio. En concreto se identificaban como datos relativos a los perfiles y a las vigas los siguientes:

- la cantidad de perfiles diferentes de una misma sección que hay disponible en stock estará entorno a cinco y ocasionalmente mayor, llegando en raras ocasiones a diez.

- La cantidad de vigas diferentes demandadas para un el horizonte del programa de producción (un día) para una misma sección es muy baja, al rededor de cinco y pocas veces llegando a diez

- La diversidad en las longitudes demandas de las vigas es, en general, muy elevada y demandándose por igual vigas pequeñas como vigas grandes

Así pues nos encontramos con instancias como la 3, 10 y 17. En estas instancias se ha comprobado que el uso de la metodología es adecuado.

Por otro lado, se ha calculado el porcentaje total de resto que supone la aplicación de la metodología para cada una de estas tres instancias. El porcentaje indicado en la tabla 5.8 incluye excesos totales (tanto chatarras como posibles puntas de producción generadas). Como se indicó en el capítulo segundo, el indicador de chatarra del procedimiento del caso estudiado estaba fijado en un 2,6% de media anual (descontando el volumen de exceso generado como puntas de producción que alcanzaba el 12%). Claramente se observa que el método planteado es considerablemente mejor ya que incluyendo posibles puntas de producción se llega a desperdicios inferiores al 5% en el peor de los casos y por debajo del 1% en el mejor de los casos.

Parámetros					Restos generados	
Instancia	p	m	*v1*	*v2*	Totales	% chatarra
3	5	10	6	120	**27,00**	**0,7**
10	5	10	60	480	**339,60**	**2,4**
17	5	10	6	480	**540,00**	**4,7**

Tabla 5.13- Resultados porcentuales de restos totales generados

5.5 EJEMPLO

Con el fin de poder ilustrar con detalle la metodología propuesta en el apartado 5.3, en esta sección se resolverá uno de los problemas de test generados a partir de la instancia 1.

Los parámetros de la instancia 1 son los siguientes: el número de tipos de perfiles se ha fijado en p ={3} y los tipos de vigas m = {5}; las longitudes en stock se generan aleatoriamente entre los valores s_1= 200 y s_2= 1000; el stock medio disponible de cada perfil es de S = 125; la longitud para los tipos de vigas estará comprendida entre v_1,=6 y v_2=120; la demanda media por viga es de 5 unidades. Estos parámetros forman el siguiente vector de instancia: [5,6,120,5,3,200,1000,125]. En la tabla 5.14 se indican los datos del problema, en la se muestra el valor de *N* para cado uno de los perfiles calculado con la expresión (5.11) y que es el número máximo de componentes que puede contener cada uno de los patrones generados para cada tipo de perfil.

Tipo de viga (m)	***Longitud viga (l)***	***Demanda viga (d)***
1	63	7
2	7	7
3	91	3
4	83	4
5	45	4

Tipo de Perfil (p)	***Longitud perfil (L)***	***Stock perfil (s)***	***Valor N expresión (5.6)***
1	508	121	10
2	655	113	13
3	650	141	13

Tabla 5.14- Datos del problema ejemplo

A partir de estos datos, se generan los patrones de corte para cada tipo de perfil aplicando el algoritmo genético del apartado 5.3.1 que realizará un número máximo de 100 iteraciones para una población de 50 individuos. Una vez se ha ejecutado el algoritmo se seleccionan 42 patrones para cada perfil (a excepción de los casos indicados en las condiciones especiales del apartado 5.3.1), lo que permite disponer en este caso de una base inicial para la resolución del problema de **126 patrones**. En la tabla 5.15 se muestran los 10 primeros patrones generados para el perfil 1, así como los restos absolutos generados por cada uno de ellos. El valor de *N* obtenido para el primer perfil es de 10, como se puede observar la suma del total de vigas cortadas por cada uno de los patrones nunca supera esa cantidad.

		Cantidad de vigas que corta						
Patrón Nº	**Perfil**	**Tipo 1**	**Tipo 2**	**Tipo 3**	**Tipo 4**	**Tipo 5**	**Total**	**Resto**
1	1	1	0	2	1	4	8	0
2	1	1	1	2	2	2	8	0
3	1	1	2	2	3	0	8	0
4	1	4	0	0	2	2	8	0
5	1	4	1	0	3	0	8	0
6	1	1	1	1	2	4	9	1
7	1	1	2	1	3	2	9	1
8	1	1	3	1	4	0	9	1
9	1	3	0	3	0	1	7	1
10	1	6	0	0	1	1	8	2

Tabla 5.15- Patrones generados para el ejemplo

En el siguiente gráfico (figura 5.19) se muestra por grupos de porcentaje (0%, 0.1%-2%, 2.1%-6%, 6.1%-12%, 12.1%-19%, 19.1%-25%, 25.1%-31%, 31.1%-100%), cómo se reparten en función del resto generado los 126 patrones. Se observa que una gran cantidad de éstos, tienen restos inferiores al 6%, lo que sin duda permite disponer de una buena base inicial sobre la que generar una primera solución al problema de corte.

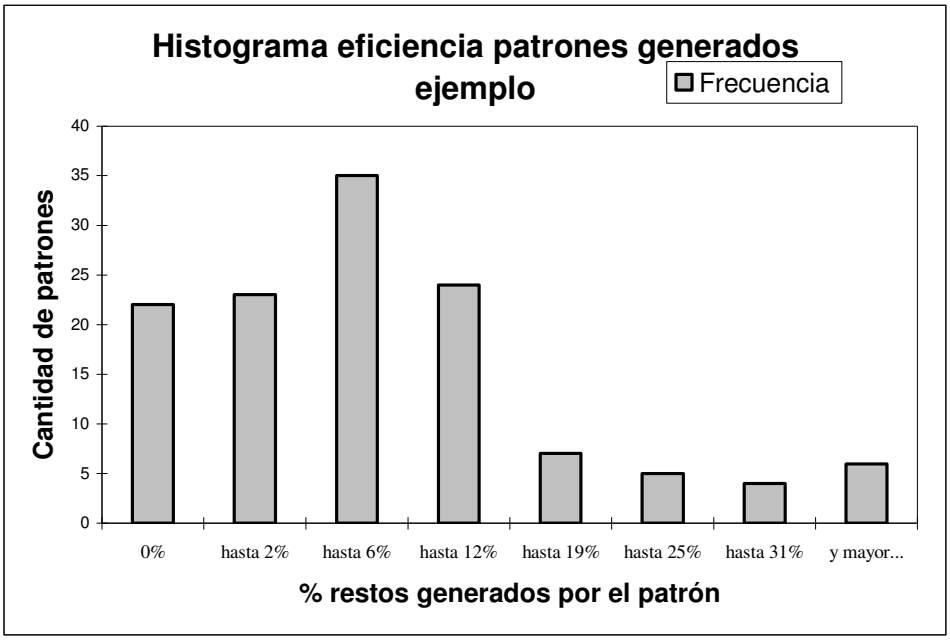

Figura 5.20- Histograma distribución patrones

Sobre la base de 126 patrones asociados a los 3 perfiles, se ejecuta el algoritmo genético de corte con un número máximo de 500 iteraciones y sobre una población de 100 individuos. Previamente a aplicar el algoritmo genético, se calculan las cotas superior e inferior que establecen los límites en el número de elementos de la solución de acuerdo con la expresión (5.17). Para el ejemplo se obtienen valores de G_{min} = 2 y G_{max} =5.

Al ejecutar el algoritmo genético se obtienen 100 soluciones ordenadas de acuerdo a su eficiencia, expresión (5.18), de las que se seleccionan las 20 primeras para posteriormente aplicar los algoritmos de mejora (apartado 5.3.3). En la siguiente tabla se muestran las 20 mejores soluciones obtenidas por el algoritmo genético de corte (apartado 5.3.2). Todas ellas con tres patrones (menor que los cinco máximos posibles), donde el número indicado es la posición del patrón en la base inicial. Se muestran también, para cada solución, los tres parámetros sobre los que se evalúan: restos absolutos generados, sobreproducción y demandas insatisfechas.

Solución Nº	Patrón 1	Patrón 2	Patrón 3	Restos	Cantidad de vigas sobreproducidas	Demandas Insatisfechas
1	3	47	87	0	6	0
2	2	48	87	0	6	0
3	2	47	88	0	6	0
4	7	48	87	1	7	0
5	7	47	88	1	7	0
6	7	47	87	1	7	0
7	6	48	88	1	7	0
8	6	48	87	1	7	0
9	6	47	88	1	7	0
10	2	47	94	1	7	0
11	7	85	94	2	7	0
12	2	47	89	0	8	0
13	85	87	93	1	8	0
14	2	47	48	0	9	0
15	7	47	89	1	9	0
16	6	47	89	1	9	0
17	2	47	87	0	7	-1
18	47	48	87	0	11	0
19	6	47	87	1	8	-1
20	1	47	94	1	8	-1

Tabla 5.16- Soluciones obtenidas por el algoritmo genético de corte

A estas 20 soluciones se les aplican los algoritmos de mejora (apartado 5.3.3). Éstos incorporan nuevos patrones a la base inicial de forma que su uso permite mejorar la solución (eliminar sobreproducción y demandas insatisfechas). La nueva base de patrones cuenta con **148 patrones** obtenidos mediante la aplicación de los algoritmos A1, A2 y A3. En la tabla 5.17 aparecen las soluciones modificadas (incorporan los nuevos patrones) así como los valores de restos que genera cada una. Se seleccionaría una de las soluciones con resto mínimo, que en este caso son: la número11 (patrones 85 y 139); la 13 (patrones 85 y 141); y la 16 (patrones 89 y 144).

Solución Nº	Patrón 1	Patrón 2	Patrón 3	Restos	% Restos	Cantidad de vigas sobreproducidas	Demandas Insatisfechas
1	47	127	0	30	2,30	0	0
2	48	128	0	30	2,30	0	0
3	47	129	0	30	2,30	0	0
4	7	130	131	249	16,34	0	0
5	47	132	0	30	1,65	0	0
6	47	133	0	30	1,65	0	0
7	48	134	0	30	1,65	0	0
8	48	135	0	30	1,65	0	0
9	6	136	137	249	16,34	0	0
10	2	94	138	391	23,47	0	0
11	**85**	**139**	**0**	**25**	**1,38**	**0**	**0**
12	47	140	0	30	1,65	0	0
13	**85**	**141**	**0**	**25**	**1,38**	**0**	**0**
14	47	142	0	30	1,65	0	0
15	47	143	0	30	1,65	0	0
16	**89**	**144**	**0**	**25**	**1,38**	**0**	**0**
17	47	145	0	30	1,65	0	0
18	48	146	0	30	1,65	0	0
19	47	147	0	30	1,65	0	0
20	1	94	148	391	23,47	0	0

Tabla 5.17- Soluciones obtenidas para el problema de corte

Se observa la coincidencia de varias soluciones en un porcentaje determinado de resto (1.38%, 1.65%,…) lo cual se debe a que al no incurrir en sobreproducción ni demandas insatisfechas, todas las soluciones obtienen el mismo número y tipo de vigas. De modo que para el bajo número de patrones de varias soluciones, si éstos actúan sobre el mismo tipo de perfil, hay una alta probabilidad de que el resultado en el resto generado sea equivalente.

CAPÍTULO 6

ALGORITMOS GENÉTICOS PARA LA RESOLUCIÓN DEL PROBLEMA DE SECUENCIACIÓN DE PATRONES

Como se ha estudiado en el capítulo anterior, en un primer estadio, la resolución de los problemas de corte requiere de la obtención de un conjunto de patrones y sus frecuencias de corte de forma que se vean satisfechas las cantidades demandas para un periodo de tiempo concreto. Sin embargo, en algunas ocasiones y en el contexto de la planificación de las operaciones industriales de corte, y más allá de la generación de una solución que satisfaga las demandas, se deben considerar otros aspectos adicionales inherentes al proceso productivo que condicionarán la manera de llevar a cabo el plan de fabricación obtenido. Dichos aspectos incluyen decisiones que tienen que ver con la determinación de **la secuencia en la que se procesan los patrones**. Diferentes objetivos deberán considerarse en las decisiones de este tipo.

En industrias como la papelera en la que hay una fuerte necesidad de evitar paros o interrupciones en la producción, aquella secuencia de corte que minimice los cambios de utillaje será la deseable (Johnston 1984). En el caso de la industria cristalera, los pedidos se entregan directamente al cliente una vez se ha completado su fabricación, evitando así operaciones adicionales de manutención y almacenaje que puedan poner en riesgo la integridad de las hojas de cristal. Así, en cuanto se ha completado la fabricación de un

pedido, éste se envía al cliente directamente desde la zona de corte. A menudo, el espacio disponible al final de las máquinas de corte es limitado lo que obligará a establecer una secuencia de corte que minimice la cantidad máxima de pedidos en curso o paquetes abiertos (*open stacks*) al final de la línea de corte y así evitar problemas de capacidad a causa de almacenajes intermedios (Yuen 1995) y (Yuen, Richardson 1995).

Como apuntan (Foerster, Wäscher 1998), algo similar ocurre también en la producción de marcos metálicos de ventanas. En este caso, se persigue el mínimo coste por manutención de materiales. Tras la operación de corte, las piezas metálicas se llevan a la sección de montaje. Como los pedidos no se almacenan parcialmente, hasta que no estén cortadas todas las piezas de una misma longitud no se pasa a la fase de montaje. El objetivo de la secuencia de fabricación será el de reducir al máximo la cantidad de tiempo que permanecen los pedidos inacabados al final de la línea. Se obtiene así un flujo continuo de materiales entre las zonas de corte y montaje que se logra manteniendo al mínimo la extensión de un pedido (*order spread*), entendida ésta como el número de patrones diferentes que tienen que procesarse entre el primer y el último patrón que contenga ese pedido.

En el **caso concreto del problema identificado en el corte de perfiles metálicos**, ya se apuntaba en el capítulo 2, la necesidad de obtener una secuencia de ejecución de los patrones que permita la optimización del espacio disponible en la línea de corte y evite que los productos en curso sean elevados. Es decir, el número de paquetes abiertos para las distintas órdenes de fabricación tendrá que ser lo más bajo posible y el tiempo que debe permanecer abierto un paquete (un paquete irá asociado a un tipo de viga) debe ser lo menor posible para permitir así una rápida expedición del pedido y una fácil identificación del material.

Así pues, **el objetivo del presente capítulo** se centra en desarrollar e implementar una metodología basada en el uso de algoritmos genéticos que permita resolver **el problema de secuenciación de patrones multiobjetivo** de forma óptima a partir de la solución al problema de corte obtenida mediante la aplicación de la metodología propuesta en el capítulo anterior.

A tal efecto, en el apartado 6.1 se revisarán las principales aportaciones bibliográficas en el contexto de la secuenciación de patrones. En el **apartado 6.2** se describen los dos problemas de secuenciación que se plantean en el corte de vigas, la minimización de la cantidad media de la extensión de pedidos (*Minimization Order Spread Problem*, MORP) y la minimización de la cantidad máxima de paquetes abiertos (*Minimization Open Stacks Problem*, MOSP), y se propone una metodología basada en algoritmos genéticos para la resolución eficaz de éstos. Posteriormente en el **apartado 6.3** se aborda en problema de optimización multiobjetivo en el que se persigue de manera conjunta la consecución tanto del MOS como del MOR. Para ello, se revisan conceptos como el de óptimo y el de frente de Pareto, así como se enumeran las principales técnicas desarrolladas en optimización multiobjetivo para posteriormente, y en ese contexto, desarrollar un algoritmo genético que permita obtener el conjunto de óptimos de Pareto. En el **apartado 6.4** se implementan los algoritmos genéticos y se muestran los resultados computacionales obtenidos para las soluciones obtenidas en el capítulo 5. Por último, en el **apartado 6.5** y a partir de los resultados obtenidos para el problema multiobjetivo de secuenciación, se propone una metodología para el Problema Global de Corte y Secuenciación. .

6.1 LOS PROBLEMAS DE SECUENCIACIÓN DE PATRONES

Los problemas de secuenciación han aparecido bajo diferentes denominaciones dependiendo de la función objetivo que se pretenda optimizar. Si se considera que cada pedido (tipo de viga) lleva asociado un paquete (*stack*) en el que se van acumulando las piezas cortadas hasta que se completa la totalidad del pedido, un objetivo podría ser la minimización de la extensión media del tiempo que un pedido permanece abierto, este problema se ha denominado como el **MORP** (*Minimization of Order Spread Problem)*. Si por el contrario intentamos minimizar el número de veces que se interrumpe el corte de una viga, el problema ha aparecido en la literatura bajo la denominación **MDP** (*Minimization of Discontinuities Problem)*. En el caso en el que se persiga minimizar la cantidad máxima de paquetes al final de la línea a lo largo de todo el proceso de corte, entonces se habla del problema **MOSP** (*Minimization of Open Stacks Problem)*. El paquete permanece abierto mientras el pedido no se ha completado. En procesos en los que la disponibilidad de espacio para paquetes al final de la máquina de corte es limitado, interesa minimizar el número de paquetes máximo (MOS).

(Andreatta et al. 1988) demuestran que el problema de secuenciación MOSP es de tipo NP completo. En (Linhares, Yanasse 2002) se presenta un listado de otros problemas NP completos que son equivalentes al MOSP. (Foerster, Wäscher 1998) consideran que el los problemas de secuenciación de patrones son del tipo NP completo y que en definitiva son una generalización del Problema del Viajante (TSP).

Muchos han sido los métodos propuestos en la literatura para resolver el MOSP. (Yuen 1991 y 1995) presenta seis heurísticas simples que resuelven el MOSP en dos etapas. En la primera fase se elige una viga cuyo paquete esté abierto y en la segunda se elige un patrón que contenga esa viga. Dependiendo de los criterios que se utilizan para elegir uno u otra viga y uno u otro patrón se define la heurística. (Yanasse 1997) propone algoritmos de ramificación y poda para resolver el MOSP a partir de una enumeración de las permutaciones de vigas posibles de forma que se obtiene la secuencia óptima. Este tipo de enfoque difiere del habitualmente utilizado, en el que la enumeración se realiza sobre las permutaciones de patrones (Faggioli, Bentivoglio 1998) o (Yuen, Richardson 1995). Se han propuesto también métodos basados en técnicas metaheurísticas para resolver tanto el MOSP como el MORP: (Fink, Voβ 1999) utilizan el recocido simulado; (Linhares et al. 1999) usan optimización basada en recocido simulado; (Oliveira, Lorena 2002) utilizan una variante de los algoritmos genéticos (algoritmo genético constructivo).

En una primera época, (Dyson, Gregory 1974) estudiaron el problema de corte con secuenciación de patrones para la minimización de las discontinuidades (MDP) en el contexto de la industria del cristal. Aplicaron una metodología de resolución estructurada en dos fases: primero resolvían el problema de corte para la minimización de desperdicios por técnicas de generación de columnas (Gilmore, Gomory 1965) y posteriormente, en una segunda etapa, utilizaban un método heurístico para resolver el problema de secuenciación. En el mismo contexto (industria del cristal), en (Madsen 1988) se aplica una metodología organizada en dos fases que resuelve en un primer estadio el problema de corte y en una segunda etapa el problema de secuenciación.

Varias han sido las publicaciones que han resuelto el problema de secuenciación de manera integrada en la misma resolución del problema de corte. Encontramos las publicaciones (Armbuster, 2002), (Pileggi 2002) y (Yanasse, Pinto 2007) que utilizan

metodologías de resolución conjunta de los problemas de corte y secuenciación. Así por ejemplo, en (Plieggi 2002) se proponen hasta tres enfoques: en el primero resuelve en dos etapas el problema integrado aplicando generación de columnas y heurísticas tipo las de (Yuen 1995) y si el resultado no cumple la restricción de cantidad máxima de paquetes abiertos, se vuelve a resolver el problema de corte pero prohibiendo los patrones que utilizan mayor cantidad de vigas; en el segundo utiliza un algoritmo tipo *greedy*; el tercer enfoque es una variación del método simplex con generación de columnas en el que obtiene una solución secuenciada de forma que un nuevo patrón sólo se incorpora a la secuencia de corte si cumple la restricción de cantidad máxima de paquetes abiertos. En (Yanasse, Pinto 2007) se utiliza la relajación lagrangiana para descomponer el problema integrado en dos subproblemas: el problema dual asociado al problema de corte se resuelve mediante el método de subgradiente modificado; y el problema de secuenciación se resuelve mediante una técnica de enumeración recursiva.

Ya que podemos considerar los problemas de secuenciación de patrones como generalizaciones del Problema del Viajante, y por tanto se pueden aplicar los métodos de resolución que se han ido proponiendo para éste, veamos en qué consiste el TSP, también conocido como *Travelling Salesman Problem* (TSP), que puede enunciarse del siguiente modo:

> *Un viajante de comercio tiene que visitar n ciudades, comenzando y terminando en su propia ciudad. Si se conoce el coste de ir de una ciudad a otra, hay que determinar el recorrido que pase por todas las ciudades y cuyo coste sea mínimo.*

Este ha sido uno de los problemas más estudiados en Investigación operativa y se ha convertido en una prueba de validación para cualquier técnica de resolución de problemas combinatorios. Formalmente podemos decir:

Sea un grafo $G = (V, A, C)$ donde V es el conjunto de vértices (ciudades), A el conjunto de aristas (caminos entre vértices) y $C=(c_{ij})$ la matriz de costes (o distancias de una arista) entre dos vértices (i,j). Definimos un camino como una sucesión de aristas en la que el vértice final de cada arista es el inicial de la siguiente y decimos que un ciclo es un camino en el que el vértice inicial del camino coincide con el final. El Problema del Viajante consiste en determinar un *tour* de coste mínimo, donde un *tour* es un ciclo que pasa por todos los

vértices del grafo G. En la siguiente figura se muestra un ejemplo de un grafo con 8 vértices en los que se muestran las aristas que indican los posibles caminos así como sus distancias a recorrer para pasar de un vértice a otro. Se marca con trazo más grueso un posible *tour* que contiene todos los vértices.

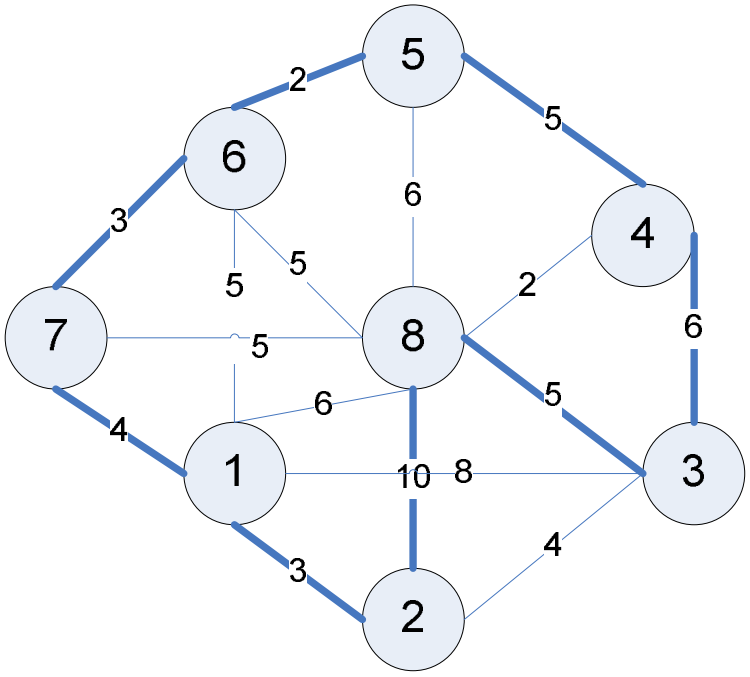

Figura 6.1- Ejemplo de *tour* en un grafo de 8 vértices

El TSP puede ser formulado mediante un modelo de programación lineal entera con variables binarias X. Si el viajante va de la ciudad i a j x_{ij} tendrá un valor de 1 y en otro caso tendrá un valor de 0. Por otro lado el coste de ir de la ciudad i a j c_{ij}.

$$z = \min \sum_{i<j} c_{ij} x_{ij} \tag{6.1}$$

s. a:

$$\sum_{i<j} x_{ij} + \sum_{j<i} x_{ij} = 2, \quad i = 1,\ldots,n \tag{6.2}$$

$$\sum_{(i,j)\in\partial(S)} x_{ij} \geq 2, \ \forall S \subseteq \{1,\ldots,n\} \ 3 \leq |S| \leq \left[\frac{n}{2}\right] \tag{6.3}$$

$$x_{ij} = 0,1 \quad \forall i < j \tag{6.4}$$

Donde $\delta(S)$ representa el conjunto de aristas incidentes con exactamente un vértice de S. La restricciones que aparecen en (6.3) reciben el nombre de *restricciones de eliminación de subtours* y garantizan que la solución sea un único *tour*. El problema de esta formulación es que aparecen del orden de $2^{n/2}$ restricciones con lo que hace que el problema no sea manejable. Mediante la incorporación de una cantidad polinómica de restricciones se evita la formación de *subtours* (Miller et al. 1960). Aún así la resolución óptima del problema resulta poco eficiente, excepto para casos pequeños. El TSP es del tipo NP- completo por lo que para resolver problemas de tamaño realista se deberá recurrir a diferentes tipos de heurísticas.

En (Johnson y McGeoch 1997) se describen las principales técnicas de resolución que se han adaptado al TSP, desde las clásicas técnicas de optimización local hasta técnicas metaheurísticas, como son el recocido simulado, las redes neuronales o los algoritmos genéticos. Se evalúa su éxito tanto desde un punto de vista experimental como teórico. Se destacan las siguientes:

- **Heurísticas constructivas**

  Se trata de procedimientos iterativos que van añadiendo un elemento hasta completar la solución. Se trata de métodos deterministas que se basan en seleccionar en cada iteración el mejor elemento de los que se evalúan. Son métodos fuertemente dependientes del problema. (Johnston, McGeoch 1997) destacan los siguientes algoritmos: el vecino más próximo (Jünger et al. 1977), Greedy, basados en ahorros (Clark, Wright 1964) y basados en árboles generadores (Christofides 1976).

- **Búsqueda local basada en 2 y 3-intercambio y sus variantes**

  Se trata de algoritmos dependientes del problema y consisten en aplicar movimientos 2-intercambio y 3-intercambio (*2-opt y 3-opt*) que eliminan dos o tres aristas del ciclo y reconectan los vértices obteniendo así nuevos caminos. Este procedimiento se basa en la observación de que si un ciclo se cruza a sí mismo, éste puede acortarse únicamente sustituyendo las aristas que se cruzan, de forma que el ciclo final es más corto que el inicial. La figura siguiente ilustra este movimiento en el que las aristas (i,d) y (c,h) son sustituidas por (c,d) y (i,h). En este tipo de procedimiento sólo existe una forma de reconectar los dos caminos.

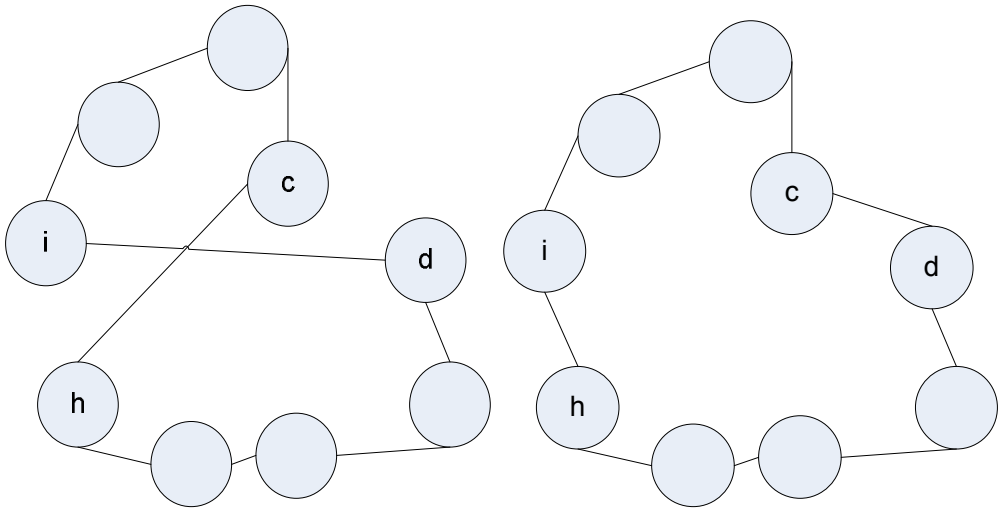

Figura 6.2- Solución mejorada a partir de un procedimiento 2 óptimo

- **Búsqueda Tabú y algoritmo de Lin-Kernighan**

  A partir de la idea de que para alcanzar una solución mejorada partiendo de un óptimo local se debe pasar por soluciones peores, se desarrolla el algoritmo de Lin-Kernigham. Se proponen movimientos compuestos en donde cada una de las partes de las que constan éstos no necesariamente mejoran la solución si bien el movimiento compuesto sí debe hacerlo.

- **Metaheurísticas: Recocido simulado, algoritmos genéticos, redes neuronales y algoritmo Lin-Kernigham iterado**

  La aplicación particular de los algoritmos genéticos a la resolución del TSP consistirá en establecer cómo realizarán las operaciones de recombinación y mutación. En la recombinación se suelen seleccionar los progenitores según su función de eficiencia, siendo el procedimiento de la ruleta uno de los más empleados. Los operadores de cruce más utilizados establecen puntos de ruptura de forma que se intercambian las cadenas. La operación de mutación más empleada consiste en realizar el reemplazo de elementos en función de una cierta probabilidad, introduciendo así un componente de diversificación para evitar una convergencia prematura del procedimiento.

(Johnson, McGoech 1997) concluyen que las mejores técnicas de resolución son aquellas basadas en búsqueda local y que si se cuenta con el tiempo suficiente como para

ejecutar algo más sofisticado que una heurística simple constructiva, la primera elección debería centrarse en la implementación de algoritmos basados en procedimientos 2-*opt* y 3-*opt* o el algoritmo Lin-Kernigham, los cuales se acercan bastante al óptimo para instancias aleatorias de hasta un millón de ciudades (vértices). Sin embargo, **en el caso de problemas con menos vértices y suponiendo que se disponga de más tiempo, tanto el recocido simulado como los algoritmos genéticos, pueden en la mayoría de ocasiones encontrar mejores soluciones** que las obtenidas por el algoritmo Lin-Kernigham en el mismo tiempo. Así pues parece adecuado el uso de algoritmos genéticos en la resolución de problemas de secuenciación de patrones de corte, tanto por el tiempo del que se dispone como por el tamaño de las secuencias que se manejan (menos de 100 patrones de corte diferentes). En la siguiente sección se propone un algoritmo genético que permita resolver de forma eficiente los problemas de secuenciación MOSP y MORP.

6.2 ALGORITMOS GENÉTICOS PARA LA RESOLUCIÓN DE PROBLEMAS DE SECUENCUACIÓN DE PATRONES EN EL CORTE DE VIGAS

Dos son los problemas de secuenciación que aparecen en el corte de vigas: minimización de la cantidad media de la extensión de pedidos (*Minimization Order Spread Problem*, MORP) y minimización de la cantidad máxima de paquetes abiertos (*Minimization Open Stacks Problem*, MOSP). En primer lugar se abordarán las consideraciones generales a tener en cuenta de cara a plantear una metodología de resolución de éstos en el contexto del corte de vigas.

En general, a la hora de tratar los problemas de secuenciación partiremos de las siguientes consideraciones previas que se indican en (Foerster, Wäscher 1998):

- Un pedido consiste en una cantidad demandada de vigas de un tamaño concreto que se han de cortar a partir de material disponible en stock. Cada pedido se relaciona de manera unívoca con un determinado tipo de viga.

- Se han determinado un conjunto de patrones y sus frecuencias correspondientes, de forma que mediante su ejecución todas las demandas son satisfechas.

- Cada patrón de corte requiere de una preparación o montaje específico de las herramientas de corte (*set-ups*). Con el fin de evitar montajes de herramientas innecesarios, patrones idénticos se cortan uno a continuación del otro, por lo que el número de *set-ups* coincide con el número de patrones diferentes.

- Sea m el número de pedidos demandados y n el número de patrones de corte distintos que satisfacen dichas demandas. Una secuencia de patrones se representará como una permutación concreta π de los n elementos $\pi=\{\pi(1), \pi(2),...,\pi(n)\}$ donde $\pi(j)$ indica el patrón que se corta en la posición j

.

6.2.1 Problema de minimización de la cantidad de paquetes abiertos (MOSP)

Definimos la cantidad máxima de paquetes abiertos (*MOS*) para una determinada secuencia de corte como el máximo de los pedidos en curso (no completados) que se encuentran simultáneamente mientras se realiza el proceso de corte de los patrones.

Para cada secuencia se puede definir una matriz $P(\pi)=p_{ij}$ en la que las filas representan los pedidos y las columnas los patrones de corte. De forma que se cumpla que $p_{ij} = \begin{cases} 1 & \text{si el pedido } i \text{ se corta con el patrón } j \\ 0 & \text{en el resto de los casos} \end{cases}$ para $i = 1, ..., m,\ j = 1, ..., n.$ Se define también otra matriz $Q^1(\pi) = q_{ij}$, llamada matriz de paquetes abiertos, que permitirá calcular el número de paquetes abiertos y que se obtiene a partir de la matriz $P(\pi)$ de la siguiente forma:

$$q_{ij} = \begin{cases} 1 & \text{si hay un } x \text{ y un } y \,/\, pos(x) \le j \le pos(y) \text{ y } p_{ix}=p_{iy}=1 \\ 0 & \text{en el resto de los casos} \end{cases} \text{ para } i = 1, ..., m,\ j = 1, ..., n,$$

en donde $pos(a)$ indica la posición del patrón a en la secuencia.

La matriz Q muestra los paquetes abiertos a lo largo de la secuencia mediante los unos consecutivos que aparecen en cada fila (para cada pedido). La cantidad máxima de paquetes (*MOS*) vendrá determinada por la siguiente expresión:

$$MOS = \max(\sum_{j} q_{ij}) \tag{6.5}$$

A continuación se muestra un ejemplo extraído de (Oliveira, Lorena 2005) de una matriz P con su correspondiente matriz Q^l para un caso de 5 patrones de corte y 8 pedidos. En cada una de las filas se observa el momento en el que se abre el paquete (cuando se procesa el primer patrón j que contiene piezas de ese pedido i) y el momento en el que el paquete se cierra (cuando se procesa el último de los patrones que contiene piezas del pedido i). La suma de cada columna j define el número de paquetes abiertos cuando se procesa el patrón al que corresponde dicha columna j.

$$P = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}; \; Q^1 = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}; \; MOS = \max\left(\sum_{j} q_{ij}\right) = max\{2,5,3,4,2\} = 5$$

Cuando se corta el primer patrón se abren 2 paquetes (correspondientes a los pedidos 3 y 5), en el segundo patrón (que corta las vigas de los pedidos 1, 2, 5, 6) se abren 3 paquetes más (vigas 1, 2 y 6) por lo que la cantidad de paquetes totales abiertos es de 5 y así sucesivamente. La cantidad máxima de paquetes abiertos (*MOS*) a lo largo de la secuencia será de 5**. En el MOSP, el objetivo será buscar la permutación que minimice la cantidad máxima de paquetes abiertos**. Si la secuencia de patrones de P corresponde a la permutación {1,2,3,4,5}, donde cada patrón se representa en una columna de P, entonces para la permutación {5,3,1,2,4} obtendríamos la matriz Q^l siguiente.

$$Q^1 = \begin{pmatrix} patrón5 & patrón3 & patrón1 & patrón2 & patrón4 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}; \; MOS = \max\{2,2,3,4,3\} = 4$$

Que es el valor óptimo, el cual se alcanza también con otras permutaciones como por ejemplo {2,3,1,5,4}.

6.2.2 Problema de minimización de la cantidad media de la extensión de pedidos (MORP)

Un pedido se completa en el momento en el que se ha alcanzado su nivel de demanda. La diferencia entre la posición del primer patrón que corta el pedido y la del último es lo que se define como la extensión del pedido (*order spread*). El objetivo que debe perseguir la secuencia de patrones es aquel que minimice la extensión media de los pedidos (*Average Order Spreads* o *AOS*).

Para cada secuencia se puede definir una matriz $P(\pi)=p_{ij}$ en la que las filas representan los pedidos y las columnas los patrones de corte. De forma que se cumpla que $p_{ij} = \begin{cases} 1 & \text{si el pedido } i \text{ se corta con el patrón } j \\ 0 & \text{en el resto de los casos} \end{cases}$ para $i = 1,...,m, \; j = 1,...,n.$ Se define también otra matriz $Q^2(\pi) = q_{ij}$ llamada matriz de extensión de pedidos que permitirá calcular la extensión media de pedidos para una secuencia concreta, de forma que se cumpla que $q_{ij} = \begin{cases} j & \text{si el pedido } i \text{ se corta con el patrón } j \\ 0 & \text{en el resto de los casos} \end{cases}$ para $i = 1,...,m, \; j = 1,...,n.$

Las entradas en la fila i de $Q^2(\pi)$ indican las posiciones en la secuencia de los patrones que contienen el pedido i. Así pues, se puede definir la extensión de un pedido i en la secuencia π como $s_{i\pi} = \max\left\{q_{ij}, j=1,...,n;\ q_{ij}>0\right\} - \min\left\{q_{ij}, j=1,...,n;\ q_{ij}>0\right\}$.

De acuerdo con esta definición de extensión de un pedido, aquél que únicamente esté incluido en un único patrón tendrá una extensión nula y aquél que se encuentre en dos patrones consecutivos tendrá un valor de extensión de uno. Podríamos decir que la extensión de cada pedido es lo que se demora su finalización una vez se ha terminado de cortar el patrón con el que se inicia su fabricación.

Para la evaluación de una secuencia completa se obtiene la extensión media de los pedidos (AOS) a partir de la siguiente ecuación:

$$AOS = \bar{s}(\pi) = \frac{1}{m}\sum_{i=1}^{m} s_{i\pi} \tag{6.6}$$

Nótese que para el cálculo del *AOS* también se podría haber utilizado la matriz Q^1 definida en el apartado anterior 6.2.1, no obstante se ha comprobado que el tiempo computacional para el cálculo de la matriz Q^2 es sensiblemente menor que la del Q^1 y esto permitirá la ejecución de algoritmos de resolución de manera más rápida a la hora de abordar independiente los problemas MORP y MOSP.

Por tanto, el problema de seleccionar una secuencia de patrones que minimice la extensión media de los pedidos (*AOS*) es una generalización del problema de viajante de comercio en el que la "matriz de distancias" debe calcularse para cada secuencia. A continuación se calcula la extensión media de pedido para la secuencia del el ejemplo anterior con 5 patrones de corte (j) y 8 pedidos de vigas (i)). A parir de la matriz P se obtiene la correspondiente matriz Q^2 con la que se obtendrá el valor de la extensión media de pedido para esa secuencia de corte. En este caso, la matriz Q^2 es diferente a la que se obtenía en el ejemplo de 5.2.1. La extensión de cada pedido se determina a partir de los elementos de cada fila (por pedido), mientras que en el caso del MOSP la cantidad de paquetes abiertos se obtenía a partir de los elementos de cada columna (por patrón). Se observa que en el caso del primer pedido, éste se corta en los patrones 2 y 3 por lo que el

valor de su extensión de pedido será de 1 ya que se corta con dos patrones que son consecutivos.

$$P = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}; \; \mathrm{Q}^2 = \begin{pmatrix} 0 & 2 & 3 & 0 & 0 & : & 1 \\ 0 & 2 & 0 & 0 & 0 & : & 0 \\ 1 & 0 & 3 & 0 & 5 & : & 4 \\ 0 & 0 & 0 & 4 & 0 & : & 0 \\ 1 & 2 & 0 & 4 & 0 & : & 3 \\ 0 & 2 & 0 & 0 & 0 & : & 0 \\ 0 & 0 & 0 & 0 & 5 & : & 0 \\ 0 & 0 & 0 & 4 & 0 & : & 0 \end{pmatrix}; \; AOS_{\pi} = \bar{S}\pi = \frac{1}{8}(1+0+4+0+3+0+0+0) = 1$$

6.2.3 Un algoritmo genético para la resolución independiente del MOSP y el MORP en el corte de vigas.

Como se apuntaba en el apartado 6.1 y de acuerdo con las conclusiones de (Johnson, McGoech 1997), en el corte de vigas es adecuado el uso de algoritmos genéticos para la resolución de problemas de secuenciación de patrones, tanto por el tiempo del que se dispone como por el tamaño de las secuencias que se manejan (menos de 100 patrones de corte diferentes). A continuación se describirá el algoritmo genético propuesto para la obtención de la secuencia de corte, que se aplicará tanto si se trata de obtener la solución que minimice la cantidad máxima de paquetes abiertos (*MOS*) como si el objetivo consiste en reducir la extensión media de pedido (*AOS*). En el apartado 6.2.4 se describirá un algoritmo que abordará la resolución en un mismo problema de ambos objetivos (minimización del *maximum open stacks* y minimización del *average order spread*) simultáneamente.

Al igual que ocurría con los algoritmos descritos en el capítulo 5, se definen también como parámetros de entrada al algoritmo los siguientes:

- Tamaño de la población (pop_size)
- Número de generaciones máximo o condiciones de finalización del algoritmo
- Tamaño de la población que pasará de una generación a otra ($Elite$)
- Procedimientos de mutación y recombinación.

Además, se debe proporcionar la matriz P_{ij} que relaciona cada uno de los patrones contenidos en la secuencia con los pedidos o vigas que hay que cortar.

- **Codificación de una solución al problema de corte**

  Una solución al problema consistirá en una secuencia ordenada de los distintos patrones que constituyen la solución al problema de corte. Por tanto, cualquier permutación de los J patrones que componen la solución será una solución al problema de secuenciación. Al igual que ocurría en el capítulo 5 cuando se abordaba la idoneidad de la aplicación de un algoritmo genético para la generación de patrones de corte y de soluciones (apartados 5.2.1 y 5.2.2), la aplicación del algoritmo sólo tendrá sentido en aquellos casos en los que la enumeración completa no sea manejable.

- **Función de eficiencia**

  Las funciones que se implementarán para la evaluación de soluciones son las de las expresiones (6.6) en el caso del MORP y la (6.5) en el caso del MOSP. Todos los individuos de cada una de las poblaciones de las iteraciones se evaluarán de acuerdo con la función de eficiencia y se almacenará el mejor de cada iteración en una matriz de históricos. Para poder evaluar cada secuencia es necesario determinar su matriz Q^1 o Q^2 a partir de su matriz P, cada secuencia tendrá, según el caso MOSP o MORP, una matriz Q^1 o Q^2 específica y diferente a las de las demás secuencias.

- **Generación de una población inicial de soluciones**

  La generación de la población inicial de soluciones se realizará de manera aleatoria. Se realizará un número de permutaciones igual al tamaño fijado de la población.

- **Procesos de clonado y mutación**

  En el caso de una codificación en la que la única diferencia entre un individuo y otro es el orden en el que se permutan los elementos de la secuencia, no tiene sentido hablar de procesos de recombinación, ya que la asociación de diferentes progenitores (padre y madre) generaría soluciones *hijo* no factibles (duplicaría algunos patrones y carecería e otros). Por lo tanto, el único proceso factible, a parte de la propia clonación del individuo, consistirá en realizar operaciones de mutación que mantengan parcialmente el orden de la secuencia de corte .

  Con la finalidad de disponer de una mayor dispersión en los nuevos individuos generados por mutación en cada iteración, se proponen tres procedimientos de clonado

que generarán a su vez tres nuevos individuos. La tasa de clonado de una generación a otra se establece en el 25% y se va a realizar en base al siguiente esquema.

1- Se asignan posiciones aleatorias a todos los elementos de la población
2- Se agrupan éstos en conjuntos de cuatro individuos
3- De cada grupo se selecciona el mejor, que automáticamente pasará a la siguiente generación y que por tanto es el individuo clonado..
4- Del individuo que se ha clonado se seleccionan dos elementos al azar y realizan las tres operaciones siguientes de mutación obteniendo tres nuevos individuos que formarán parte de la nueva población:
    a. Se intercambian sus posiciones.
    b. Se realiza un procedimiento del tipo 2 *opt* como el que se muestra en la figura 6.1
    c. Se desliza la secuencia entre los puntos seleccionados una posición hacia la izquierda
5- Los nuevos elementos resultantes de la mutación junto con los clonados constituyen la nueva población que se evaluará en base a la función de eficiencia definida.

Este tipo de procedimientos de mutación en algoritmos genéticos aplicados a la resolución del problema del viajante de comercio (TSP) lo podemos encontrar en Brady (1985) o Martin, Otto y Felten (1991).

En la siguiente figura se muestra un ejemplo de los procedimientos de mutación que se proponen a partir de una secuencia {1,2,3,4,5,6,7} y seleccionados 2 elementos al azar {2} y {6}.

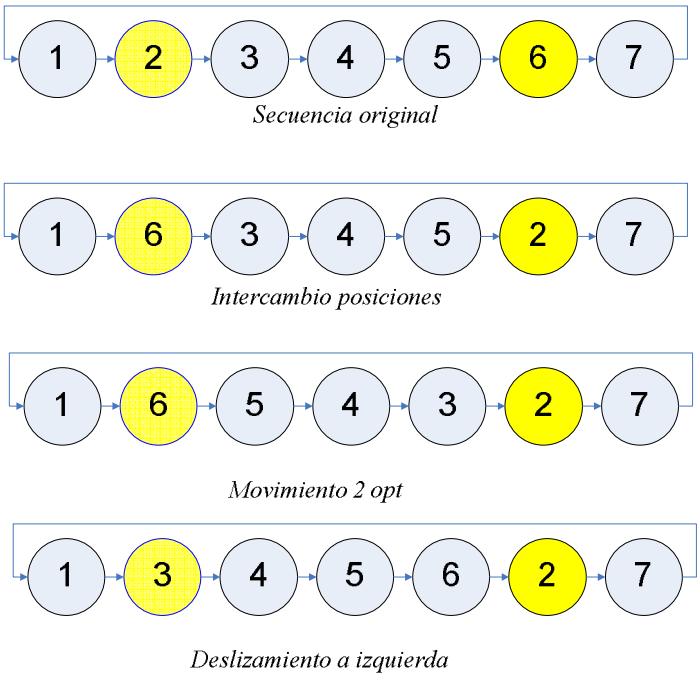

Figura 6.3- Procedimientos de mutación de individuos

6.3 RESOLUCIÓN CONJUNTA DEL MOSP Y EL MORP EN EL CORTE DE VIGAS (PROBLEMA MULTIOBJETIVO)

Ante la necesidad de llegar a una secuencia que minimice tanto la cantidad máxima de paquetes abiertos (*MOS*) por restricciones de espacio como la extensión media de los pedidos (*AOS*) por restricciones en tiempos de entrega nos encontramos con que el problema de optimización se convierte en problema de naturaleza multiobjetivo. A continuación se abordan algunos conceptos básicos en el contexto de la optimización multiobjetivo así como las principales técnicas que se utilizan para la resolución de problemas de esta naturaleza.

6.3.1 Optimización Multiobjetivo

El planteamiento del problema de optimización multiobjetivo es similar al de optimización con un único objetivo. El problema consiste en encontrar el vector solución $\vec{x}$ que optimice:

$$\vec{f}(\vec{x})=\left[f_1(\vec{x}), f_2(\vec{x}),...,f_k(\vec{x})\right]^T, \tag{6.7}$$

donde $\vec{f}$ es el vector de funciones objetivo, compuesto por las funciones f_i, con $i=1,2,...,k$. El problema puede o no tener restricciones, aunque en nuestro caso no las tiene. El concepto de optimizar varias funciones simultáneamente es más complejo que encontrar un óptimo para cada función ya que en la mayoría de los casos éstas suelen estar en conflicto entre sí. Así pues, el óptimo puede interpretarse como encontrar un buen compromiso entre las funciones objetivo del problema.

Vilfredo Pareto dio, en el siglo XIX, una definición formal de óptimo para el caso de problemas multiobjetivo, que actualmente se conoce como óptimo de Pareto y que se basa en el concepto de dominancia. Si como en nuestro caso se trata de un problema de minimización de objetivos, entonces se dice que un punto $\vec{x}$ domina a otro y si y sólo si:

$$f_i(\vec{x})\leq f_i(\vec{y}) \quad i=1,...,k, \tag{6.8}$$

y que, para al menos un i se cumple que:

$$f_i(\vec{x})< f_i(\vec{y}). \tag{6.9}$$

Por otro lado, se dice que un punto $\vec{x}*$ es un óptimo de Pareto, si no existe ningún otro punto $\vec{x}$ que lo domine.

Es evidente que, un problema no necesariamente tendrá sólo un único punto que sea óptimo de Pareto. En general, se tiene un conjunto de puntos que cumplen con la definición de óptimo de Pareto. Así pues, el proceso de optimización se encargará de encontrar los puntos óptimos de Pareto de forma que en una fase posterior el tomador de decisiones elija la solución final que habrá de utilizarse en la práctica.

Otro de los conceptos habituales en la optimización multiobjetivo es el de frente de Pareto. Se define el frente de Pareto como el contradominio del conjunto de puntos que son el óptimo de Pareto. En problemas multiobjetivo con espacios continuos, el frente de Pareto suele ser una línea casi continua. Con frecuencia se representan gráficamente los frentes de Pareto para mostrar el resultado de los problemas multiobjetivo.

En (Coello 1996) se puede encontrar una buena revisión sobre las diferentes técnicas utilizadas para la resolución de los problemas multiobjetivo. A la hora de obtener los puntos óptimos de Pareto, aquellas que aplican métodos de computación evolutiva están en ventaja frente a otros métodos de optimización ya que operan sobre un conjunto poblacional de individuos y por tanto en una sola ejecución pueden arrojar un conjunto de puntos óptimos.

Se pueden clasificar las técnicas evolutivas propuestas para optimización multiobjetivo en función del momento en el que se establecen las preferencias sobre las distintas funciones objetivo.

Así pues, hablamos de **Técnicas a Priori** cuando las preferencias sobre las funciones objetivo son dadas antes de que la técnica comience la búsqueda. Las más significativas son:

- **Orden lexicográfico:** las funciones objetivo se ordenan según su importancia, y se optimizan en orden comenzando por la más importante, y terminando por la menos importante

- **Funciones de agregación lineales:** combinan los resultados de las distintas funciones objetivo en un solo valor de aptitud. Este único valor casi siempre se obtiene mediante una combinación lineal de las funciones objetivo, aunque no es la única manera de hacerlo.

- **Funciones de agregación no lineales:** las más utilizadas han sido las técnicas basadas en vectores de metas que minimizan la diferencia entre los valores generados por el proceso de búsqueda, y un vector de objetivos deseados.

Las **Técnicas Progresivas** son aquellas en las que las preferencias se expresan mientras el proceso está generando soluciones. El encargado de tomar las decisiones expresará cómo de buena es una solución, y el proceso va actualizando las preferencias a fin de considerar la opinión del tomador de decisiones, antes de continuar con el proceso de búsqueda.

Por último en las **Técnicas a Posteriori** las preferencias se expresan al final del proceso de generación de soluciones, de forma que no se interfiere en la búsqueda. Se generan soluciones que representan todos los compromisos entre las funciones objetivo (es decir, se intenta generar un frente de Pareto completo), y posteriormente el tomador de decisiones expresará las preferencias al elegir la solución final. Algunos ejemplos de este tipo son:

- **Técnicas de muestreo independiente:** realizan varias ejecuciones de la técnica para encontrar distintos puntos del frente de Pareto. Las técnicas de muestreo independiente que se han propuesto en computación evolutiva consisten en funciones agregativas, pero en este caso los pesos no representan las preferencias, sino que son variados para realizar ejecuciones independientes, y así encontrar distintas porciones del frente de Pareto.

- **Técnicas de selección por criterio:** Se divide a la población total, y en cada subpoblación se realiza la selección basada únicamente en uno de los objetivos. El ejemplo más claro de estas técnicas es el algoritmo de (Schaffer 1985) llamado VEGA (*Vector Evaluated Genetic Algorithm*). Una vez realizada la selección, la población se mezcla para aplicar el resto de los operadores evolutivos. Todo este proceso se repite en cada generación. Una limitación de VEGA es que "prefiere" las soluciones mejores en sólo uno de los objetivos.

- **Técnicas con muestreo de Pareto:** en este grupo de técnicas se identifican a los individuos no dominados, óptimo de Pareto, y se utiliza esa información para hacer la asignación de la aptitud. Existen varias formas de llevar a cabo la selección utilizando muestreo de Pareto, pero la idea original es de (Goldberg 1989) que propuso una jerarquización en la que primero se identifican los individuos no dominados de la población, asignándoles la jerarquía más alta. Se continúa identificando los no dominados de la población restante, a los que se les asigna el siguiente nivel de jerarquía y así hasta que se ha asignado una jerarquía a todos los individuos de la población. Para evitar una convergencia en un único punto no dominado, se propone una técnica basada en nichos. En este sentido, el uso de técnicas para mantener diversidad se ha convertido en una característica general de los algoritmos evolutivos

para optimización multiobjetivo. Un ejemplo del muestreo de Pareto con jerarquización lo encontramos en un algoritmo llamado MOGA (*Multi-Objective Genetic Algorithm*) desarrollado en (Fonseca, Fleming 1995), donde la jerarquía del individuo depende del número de individuos de la población actual que lo dominan. En (Srinivas, Deb 1994) se propone el algoritmo NSGA (*Nondominated Sorting Genetic Algorithm*) que utiliza casi sin modificar la idea de la jerarquización de Goldberg, la cual se hace por capas.

En optimización evolutiva de tipo mono-objetivo, retener al mejor individuo (élite) intacto entre generaciones es necesario para asegurar la convergencia al óptimo. Sin embargo, en el caso de la optimización evolutiva del tipo multiobjetivo no está claro el papel que puede jugar el elitismo, si bien la mayoría de las técnicas modernas lo incluyen. El elitismo en optimización multiobjetivo es más complicado que en el caso mono-objetivo, ya que todos los individuos no dominados pueden ser igualmente buenos. Así pues, la forma más frecuente de mantener intacta a la élite en optimización evolutiva multiobjetivo se lleva a cabo mediante un archivo externo, que almacena los individuos no dominados.

6.3.2 Un algoritmo genético para la resolución conjunta del MOSP y el MORP

En cuanto a la relación entre los objetivos, intuitivamente se puede prever que ambos estén muy relacionados ya que secuencias con una *AOS* pequeña supondrá que las vigas permanezcan poco tiempo al pie de la línea y por tanto la cantidad media de paquetes abiertos sea también pequeña, lo que redundará en un nivel bajo de *MOS*. Esto se observa de forma empírica en los estudios realizados sobre los algoritmos del apartado 5.2.3. En una misma instancia, y a partir de un número “suficiente” de iteraciones, se consigue que la secuencia resultado obtenida por el algoritmo que resuelve el MOSP obtenga valores de *AOS* muy próximos a los obtenidos por el algoritmo del MORP y viceversa. Es decir, las secuencias cuyos valores de *AOS* son mínimos obtienen valores cuasi mínimos de *MOS*, y al revés.

A la hora de plantear un algoritmo que aborde la resolución del problema multiobjetivo debemos considerar las siguientes características que condicionarán la definición de un método de evaluar ambos objetivos de forma conjunta:

- La cantidad de paquetes abiertos es un número entero. Lo que tiene dos consecuencias importantes: por un lado hace que el frente de Pareto sea discontinuo; y por otro hace que en un problema MOSP se suelan encontrar varias secuencias solución que alcanzan el valor mínimo.

- Al ser el frente de Pareto discontinuo y las secuencias con valores mínimos de *AOS* tener valores de *MOS* cercanos al mínimo (ej. siguiente entero) y viceversa. El frente de Pareto estará constituido por una cantidad discreta de puntos (muchas veces por sólo dos puntos).

- Desde el punto de vista computacional, la evaluación de cada secuencia de corte y, sobre todo, la obtención de su correspondiente matriz Q_{ij} es en lo que invierte la mayor parte del tiempo de ejecución del algoritmo. La generación de las matrices Q_{ij} para el MOSP es más costosa que la de las matrices Q_{ij} del MORP ya que requiere de dos bucles que recorren elemento a elemento la matriz $P_{ij.}$ A modo de ilustración, se muestran los códigos necesarios en cada caso:

  - **Código para la obtención de Q_{ij} en el MOSP**

```
for j=1:n
    auxiliar(:,j)=xy(:,pop(p,j));
end
[r c]=size(auxiliar);
for i=1:r
    for j=1:c
      if auxiliar(i,j)==0;
        if logical(sum(auxiliar(i,1:j-1))) && logical(sum(auxiliar(i,j+1:c)))
            auxiliar(i,j)=1;
        end
     end
   end
end
for i=1:size(xy,2)
    stacks(i)=sum(auxiliar(:,i));
end
total_dist(p) =max(stacks);
```

  - **Código para la obtención de Q_{ij} en el MORP**

```
for j=1:n
    auxiliar(:,j)=xy(:,pop(p,j))*j;
end
for i=1:size(xy,1)
 morp(i)=max(auxiliar(i,find(auxiliar(i,:))))-
       min(auxiliar(i,find(auxiliar(i,:))));
```

```
end
total_dist(p) =mean(morp);
```

El número de instrucciones empleadas en el caso del MOSP es considerablemente mayor que en el MORP y por tanto computacionalmente su evaluación es más costosa. Así pues la técnica propuesta para la optimización multiobjetivo, deberá tener en cuenta el coste computacional y evitar evaluar a todos los individuos de la población de acuerdo a las dos funciones objetivo.

De entre las técnicas arriba descritas y teniendo en cuenta que el conjunto de óptimos de Pareto va a estar constituido generalmente por dos puntos, se opta por **una técnica de muestreo independiente** en la que se obtendrán los óptimos de Pareto en dos ejecuciones independientes. El problema de establecer una jerarquización eficiente de individuos desaparece al tratarse de un frente de Pareto discontinuo con dos objetivos con relación proporcional.

En las dos ejecuciones del algoritmo se buscará la secuencia, de entre todas las que hacen mínima una de las funciones objetivo, la que tenga el menor valor de la otra función objetivo.

- En la **primera ejecución** se procede de la siguiente manera: Se inicializa el algoritmo genético y se realiza una primera evaluación de todas las soluciones de acuerdo con su *AOS* y aquella o aquellas que tengan el menor *AOS* se evalúan de acuerdo con su *MOS*. A tal efecto, definiremos una nueva función de eficiencia $(dist\_total)^1$ que incorpora dos términos ($dist1$ y $dist2$). En cada uno de ellos se almacena información sobre cada objetivo: MORP y MOSP. En concreto para una secuencia π, tenemos la siguiente expresión:

$$dist\_total_{\pi}^{1} = (dist1_{\pi} + dist2_{\pi}) \tag{6.10}$$

$$dist1_{\pi} = AOS_{\pi}$$

$$dist2_{\pi} = \begin{cases} MOS_{\pi} & \text{si } AOS_{\pi} = Min\{AOS_{pop}\} \\ \infty & \text{en el resto de los casos} \end{cases} \qquad \text{donde} \quad pop = 1, ..., pop\_size.$$

El procedimiento queda como sigue:

1- Se evalúan todos los elementos de la población obteniendo sus AOS de acuerdo con la expresión (6.6) que será el valor del parámetro $dist1 = AOS$
2- Para aquellos individuos cuyo AOS sea Mínimo se obtiene su valor máximo de paquetes abiertos de acuerdo con (6.5) se hace $dist2 = MOS$
3- Para el resto de individuos se hace $dist2 = \infty$
4- Se selecciona como mejor individuo de la población aquel cuyo *dist_total*[1] de (6.10) sea menor, por ej. el que de entre los de menor *AOS* tenga la mínima *MOS*.

Siempre se clonará el mejor de cada iteración, por lo que será posible mejorar el *MOS* manteniendo como mínimo el valor obtenido para el *AOS*.

- En la **segunda ejecución** se procederá de manera análoga: Se inicializa el algoritmo genético y se realiza una primera evaluación de todas las soluciones de acuerdo con su *MOS* y aquella o aquellas que tengan el menor *MOS* se les evaluará de acuerdo con su *AOS*. A tal efecto, redefinimos la nueva función de eficiencia (*dist_total*[2]) que incorpora dos términos (*dist1* y *dist2*). En cada uno de ellos se almacena información sobre cada objetivo: MORP y MOSP. En concreto, para una secuencia π tenemos la siguiente expresión:

$$dist\_total_{\pi}^{2} = (dist1_{\pi} + dist2_{\pi}) \tag{6.11}$$

Donde

$$dist1_{\pi} = MOS_{\pi}$$

$$dist2_{\pi} = \begin{cases} AOS_{\pi} & \text{si } \overline{MOS}_{\pi} = Min\{\overline{MOS}\, pop\} \\ \infty & \text{en el resto de los casos} \end{cases} \qquad \text{donde} \quad pop = 1, ..., pop\_size.$$

Primero se determinará para toda la población de secuencias (*pop_size*) su cantidad máxima de paquetes abiertos (*MOS*) y para aquel o aquellos individuos cantidad máxima de paquetes abiertos sea mínima se obtendrá también su *AOS*. Finalmente como mejor individuo de la iteración se seleccionará aquel que minimice la expresión (6.11), el que de entre los de menor *AOS* tenga la mínima *MOS*.

6.4 IMPLEMENTACIÓN Y RESULTADOS COMPUTACIONALES

La evaluación del desempeño de los algoritmos genéticos propuestos para la resolución de los problemas de secuenciación de patrones se realizará sobre las soluciones al problema de corte obtenidas en el capítulo 5 para las instancias definidas en el apartado 5.3.2. El análisis del desempeño de los algoritmos se centrará en la evaluación de la velocidad de éste al valor óptimo en función del número de patrones que contiene la secuencia. La implementación se realiza utilizando el software matemático *MATLAB*.

6.4.1 Instancias

Las instancias que utilizaremos para el análisis del desempeño de los algoritmos propuestos serán las obtenidas como solución al problema de corte. Para cada una de las 21 instancias de baja demanda se generaron distintos problemas de test y de cada problema de test se obtuvieron hasta 20 posibles soluciones sin sobreproducción. En función del tipo de instancia las soluciones obtenidas contienen más o menos patrones de corte. La evaluación de los algoritmos se organizará en base al número de patrones que contiene la solución, ya que cuantos más patrones tiene una solución, más complejo es obtener su secuenciación óptima. En la siguiente tabla se muestra el número patrones medio que se obtuvo para cada tipo de instancias y su desviación típica.

Parámetros						
					Cantidad de patrones diferentes	
Instancia	*P*	*m*	*v1*	*v2*	*Media*	*Desviación típica*
1	3	5	6	120	**2,17**	**0,40**
2	3	10	6	120	**7,14**	**1,26**
3	5	10	6	120	**5,4**	**1,13**
4	7	10	6	120	**5,26**	**1,53**
5	3	20	6	120	**12,45**	**3,30**
6	5	20	6	120	**13,1**	**2,53**
7	7	20	6	120	**14,81**	**2,45**
8	3	5	60	480	**6,26**	**2,15**
9	3	10	60	480	**14,92**	**2,62**
10	5	10	60	480	**14,43**	**2,05**
11	7	10	60	480	**15,24**	**2,88**
12	3	20	60	480	**25,3**	**4,24**
13	5	20	60	480	**27,33**	**3,56**
14	7	20	60	480	**27,9**	**4,79**
15	3	5	6	480	**5,71**	**1,75**
16	3	10	6	480	**12,87**	**6,51**
17	5	10	6	480	**15**	**2,44**
18	7	10	6	480	**13,77**	**2,11**
19	3	20	6	480	**26**	**3,32**
20	5	20	6	480	**25,1**	**3,59**
21	7	20	6	480	**12,59**	**3,14**

Tabla 6.1- Cantidades de patrones para instancias de bajas demandas

Se observa que la cantidad de patrones en la secuencia oscila desde el valor 2,17 obtenido para la instancia 1 hasta los 27,9 patrones obtenidos para la instancia 14. Analizaremos por tanto el desempeño de los algoritmos para tres tipos de cantidades: para secuencias de corte con entorno a 10 patrones diferentes; para secuencias con 20 patrones diferentes y para secuencias de entre 25 a 30 patrones diferentes.

Para la evaluación de los algoritmos seleccionaremos 20 soluciones distintas, que serán los problemas de test para el algoritmo, a partir de las que se obtuvieron en el capítulo anterior en la resolución del problema de corte de problemas generados a partir de las instancias

número 6, 16 y 13. Cada una con los siguientes parámetros: Instancia 6 [1,20,6,120,5,5,200,1000,125] con secuencias que contienen aproximadamente 10 patrones diferentes; Instancia 16 [1,10,6,480,5,3,200,1000,125] con secuencias que contienen aproximadamente 20 patrones diferentes; Instancia 13 [1,20,60,480,5,5,200,1000,125] con secuencias que contienen aproximadamente 30.

6.4.2 Resultados

En todos los casos se establecieron como parámetros de entrada de los algoritmos genéticos un tamaño de la población de 50 individuos y un número de 500 y 1000 generaciones (iteraciones).

A la hora de analizar el desempeño de los algoritmos, nos interesarán aspectos como son la rapidez en la convergencia a los valores óptimos, la interrelación que hay entre los objetivos o la capacidad para conseguir secuencias óptimas.

- **Análisis de la convergencia del algoritmo genético en cada uno de los problemas (MORP, MOSP y Multiobjetivo)**

  En la siguiente tabla se indican los valores medios de las iteraciones en las que se alcanzan los óptimos para cada problema. Los datos se agrupan en función de la longitud de las secuencias (10, 20 o 30 patrones por secuencia). En el caso del problema multiobjetivo, como se realizan dos ejecuciones independientes el valor que se considera es el máximo de éstas.

Secuencia 10 patrones			*Secuencia 20 patrones*		*Secuencia 30 patrones*	
TIPO PROBLEMA	*Iteración alcanza el óptimo*	*Desviación típica*	*Iteración alcanza el óptimo*	*Desviación típica*	*Iteración alcanza el óptimo*	*Desviación típica*
MORP	72,5	144,97	86,6	43,38	210	127,18
MOSP	24,5	31,04	28,7	40,43	296,7	126,07
MULTI OBJETIVO	117,9 *	98,06	96,9 *	30,68	348 *	75,36

(*) el valor que se considera es el máximo del las dos ejecuciones que se realizan en el algoritmo del problema multiobjetivo

Tabla 6.2- Convergencia del algoritmo en cada tipo de problema

En la siguiente gráfica se representan los valores de número de iteración en la que se alcanza el óptimo obtenidos para cada rango de patrones por secuencia (10, 20 y 30) en cada uno de los problemas.

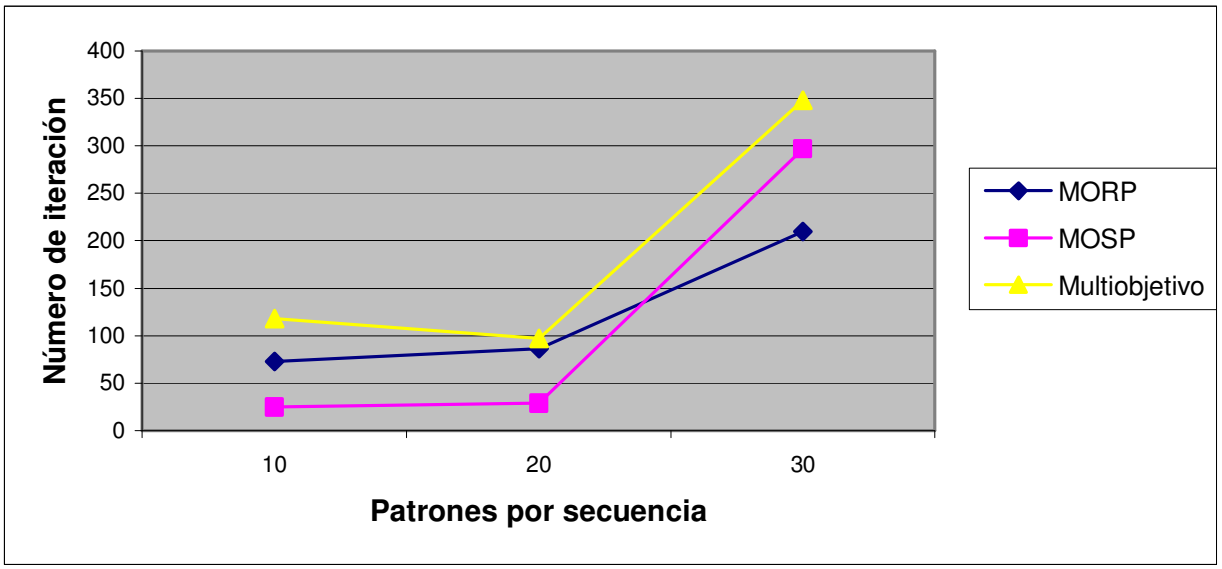

Figura 6.4- Convergencia del algoritmo en cada tipo de problema

Observamos que la convergencia es más rápida en el problema de minimización de paquetes (MOSP) que en el de extensión media de pedido (MORP), esto es una consecuencia de que la variable MOS es entera y para secuencias con un número de patrones pequeño, se alcanza el óptimo rápidamente, lo que no ocurre para cadenas de 40 patrones. En el caso del problema multiobjetivo, como se realizan dos ejecuciones independientes y para un valor mínimo de un objetivo se busca el menordel otro, la convergencia es más lenta. A la vista de los resultados podemos concluir que **la convergencia es rápida en cualquier caso y por tanto el uso de algoritmos genéticos en es adecuado**.

- **Análisis de la relación entre los objetivos cantidad de paquetes abiertos (*MOS*) y extensión media de pedido (*AOS*).**

  Para comprobar la relación existente entre los objetivos *MOS* y *AOS* se calculará, para cada una de las soluciones óptimas obtenidas en la resolución de uno de los problemas, su valor para la función objetivo del otro problema y se comparará con el valor óptimo

obtenido. Así por ejemplo, en el caso de las soluciones óptimas del MORP se calculará su valor de *AOS* y se comparará por el obtenido al resolver el MOSP. En las siguientes tablas (6.3 y 6.4), se agrupan los datos por bloques de secuencias (10, 20 y 30 patrones), en el primer caso se han analizado veinte problemas de test y en el segundo, al comprobar cierta relación directa, se han analizado únicamente diez. Se observa que en ambos casos con las soluciones óptimas de uno de los problemas de minimización se obtienen valores en el otro objetivo muy cercanos a los óptimos. Por tanto es **evidente la relación directa existente entre ambos objetivos.**

Secuencia 10 patrones				*Secuencia 20 patrones*				*Secuencia 30 patrones*			
Problema test	*Valor MOS MORP (I)*	*Óptimo MOS MOSP (II)*	*I-II*	*Problema test*	*Valor MOS MORP (I)*	*Óptimo MOS MOSP (II)*	*I-II*	*Problema test*	*Valor MOS MORP (I)*	*Óptimo MOS MOSP (II)*	*I-II*
1	12	11	**1**	1	3	3	**0**	1	5	5	**0**
2	12	12	**0**	2	3	3	**0**	2	7	7	**0**
3	12	12	**0**	3	3	3	**0**	3	6	6	**0**
4	13	13	**0**	4	3	3	**0**	4	7	5	**2**
5	11	11	**0**	5	3	3	**0**	5	6	6	**0**
6	13	12	**1**	6	3	3	**0**	6	7	5	**2**
7	13	12	**1**	7	3	3	**0**	7	8	7	**1**
8	12	11	**1**	8	3	3	**0**	8	6	6	**0**
9	12	12	**0**	9	3	3	**0**	9	6	6	**0**
10	11	11	**0**	10	3	3	**0**	10	6	6	**0**
11	12	11	**1**	11	3	3	**0**	11	5	5	**0**
12	12	12	**0**	12	3	3	**0**	12	7	7	**0**
13	12	12	**0**	13	3	3	**0**	13	6	6	**0**
14	13	13	**0**	14	3	3	**0**	14	7	5	**2**
15	11	11	**0**	15	3	3	**0**	15	6	6	**0**
16	13	12	**1**	16	3	3	**0**	16	7	5	**2**
17	13	12	**1**	17	3	3	**0**	17	8	7	**1**
18	12	11	**1**	18	3	3	**0**	18	6	6	**0**
19	12	12	**0**	19	3	3	**0**	19	6	6	**0**
20	11	11	**0**	20	3	3	**0**	20	6	6	**0**

Tabla 6.3- Relación entre cantidad máxima de paquetes abiertos y extensión media de pedido

Secuencia 10 patrones				*Secuencia 20 patrones*				*Secuencia 30 patrones*			
Problema test	*Óptimo AOS MORP (I)*	*Valor AOS MOSP (II)*	*II-I*	*Problema test*	*Óptimo AOS MORP (I)*	*Valor AOS MOSP (II)*	*II-I*	*Problema test*	*Óptimo AOS MORP (I)*	*Valor AOS MOSP (II)*	*II-I*
1	4,25	4,45	**0,20**	1	1,7	1,7	**0,00**	1	4,15	4,15	**0,00**
2	4,35	4,35	**0,00**	2	1,7	1,7	**0,00**	2	4,9	5,2	**0,30**
3	4,7	4,75	**0,05**	3	1,6	1,6	**0,00**	3	4,8	4,8	**0,00**
4	4,05	4,05	**0,00**	4	1,6	1,6	**0,00**	4	4,25	4,35	**0,10**
5	3,15	3,25	**0,10**	5	1,6	1,6	**0,00**	5	4,9	4,9	**0,00**
6	4,15	4,15	**0,00**	6	1,6	1,6	**0,00**	6	4,35	4,4	**0,05**
7	5,35	5,4	**0,05**	7	1,6	1,6	**0,00**	7	5,35	5,35	**0,00**
8	4,75	4,9	**0,15**	8	2,1	2,2	**0,10**	8	4,25	4,55	**0,30**
9	4	4,2	**0,20**	9	1,9	1,9	**0,00**	9	4,75	4,85	**0,10**
10	2,85	2,85	**0,00**	10	1,6	1,6	**0,00**	10	4,75	5,25	**0,50**

Tabla 6.4- Relación entre cantidad máxima de paquetes abiertos y extensión media de pedido

En el caso de secuencias con patrones que contienen una cantidad pequeña de vigas diferentes como ocurre en las secuencias de 20 patrones, en la que el patrón que más vigas distintas contiene es de tres, la convergencia del óptimo de un problema de minimización al óptimo del otro problema de minimización es completa. Para secuencias con patrones que contienen un valor más alto de vigas como es el caso de las secuencias de 10 patrones, cuyo patrón máximo contiene entorno a nueve vigas distintas, o las secuencias de 30 patrones, con patrones entorno a 5 vigas diferentes, se observa que en algunos casos el valor de *MOS* del óptimo del MORP se queda un entero por encima del óptimo del MOSP. Únicamente en algunas de los problemas de las secuencias con 30 patrones se observa que se aleja del óptimo hasta dos enteros, esto es debido a que la convergencia del algoritmo para secuencias más largas se sitúa en valores próximos a 500 que es el número de iteraciones del algoritmo.

Por tanto queda probada la relación existente entre ambos objetivos lo que conllevará que el conjunto de puntos del frente de Pareto sea reducido y justifica el uso de técnicas de muestreo independiente.

- **Análisis del desempeño del algoritmo multiobjetivo (obtención del frente de Pareto)**

  Para cada tipo de secuencia (10, 20 y 30 patrones) se analizan los resultados obtenidos por el algoritmo aplicado al problema multiobjetivo. Al tener dos ejecuciones independientes, el algoritmo obtendrá dos puntos óptimos (uno en cada ejecución), uno de ellos hará mínima la *MOS* y el otro el *AOS*, constituyendo el frente de Pareto. A continuación, en la tabla 6.5, se muestran los resultados obtenidos para cada tipo de secuencia. En negrita se han destacado los frentes de Pareto constituidos por un único punto (se llega al mismo punto en ambas ejecuciones). Se observa que en el caso de las secuencias de 20 patrones que tienen un valor de MOS de 3, todos los problemas obtienen un único óptimo de Pareto en los que simultáneamente se alcanzan los valores óptimos de AOS y MOS obtenidos al resolver los problemas MORP y MOSP. Sin embargo, en el caso de las secuencias de 10 y 30 patrones que tienen valores más elevados de MOS en bastantes casos se obtienen dos valores óptimos de Pareto.

  En cualquier caso y debido a la relación directa entre ambos objetivos **mediante la aplicación del algoritmo se obtienen los puntos que constituyen el frente de Pareto**.

Secuencia 10 patrones						
Problema test	*AOS Óptimo*	*MOS Óptimo*	*punto 1*		*Punto 2*	
1	4,25	11	4,25	12	4,45	11
2	4,35	12	**4,35**	**12**	**4,35**	**12**
3	4,7	12	4,70	12	4,75	12
4	4,05	13	**4,05**	**13**	**4,05**	**13**
5	3,15	11	**3,15**	**11**	3,25	11
6	4,15	13	**4,15**	**13**	**4,15**	**13**
7	5,35	12	5,30	13	5,40	12
8	4,75	11	4,75	12	4,90	11
9	4	12	**4,00**	**12**	4,20	12
10	2,85	11	**2,85**	**11**	**2,85**	**11**

Secuencia 20 patrones						
Problema test	*AOS Óptimo*	*MOS Óptimo*	*punto 1*		*Punto 2*	
1	1,7	3	**1,70**	**3**	**1,70**	**3**
2	1,7	3	**1,70**	**3**	**1,70**	**3**
3	1,6	3	**1,60**	**3**	**1,60**	**3**
4	1,6	3	**1,60**	**3**	**1,60**	**3**
5	1,6	3	**1,60**	**3**	**1,60**	**3**
6	1,6	3	**1,60**	**3**	**1,60**	**3**
7	1,6	3	**1,60**	**3**	**1,60**	**3**
8	2,1	3	**2,10**	**3**	**2,10**	**3**
9	1,9	3	**1,90**	**3**	**1,90**	**3**
10	1,6	3	**1,60**	**3**	**1,60**	**3**
Secuencia 30 patrones						
Problema test	*AOS Óptimo*	*MOS Óptimo*	*punto 1*		*Punto 2*	
1	4,15	5	**4,15**	**5**	4,25	5
2	5,15	7	**5,15**	**7**	5,70	7
3	4,05	6	**4,05**	**6**	4,35	6
4	4,35	5	**4,35**	**7**	**4,35**	**5**
5	4,9	6	**4,90**	**6**	**4,90**	**6**
6	4,35	5	4,35	7	4,40	6
7	5,25	7	5,25	8	5,35	7
8	4,25	6	**4,25**	**6**	4,55	6
9	4,8	6	**4,80**	**6**	4,85	6
10	4,85	6	**4,85**	**6**	5,25	6

Tabla 6.5- Desempeño del algoritmo para el problema Multiobjetivo

- **Análisis del desempeño del algoritmo para la resolución del MOSP**

  Otro de los aspectos a analizar del desempeño de los algoritmos será la capacidad de obtener soluciones (secuencias) óptimas. En el caso del problema de minimización de cantidad máxima de paquetes abiertos (MOSP), **el parámetro que se pueden utilizar a la hora de valorar la eficiencia del algoritmo es la cantidad de vigas diferentes que corta el patrón con mayor número de vigas diferentes**. Notar que el problema tendrá un valor de *MOS* con una cota inferior limitada por este dato ya. El dato vendrá determinado por la matriz que relaciona las vigas con los patrones de la secuencia.

Se han estudiado **tres tipos secuencias**: secuencias cuyo patrón con mayor cantidad de vigas diferentes cortadas es de 3 vigas; secuencias con patrones máximos que cortan entre 3 y 5 vigas diferentes; patrones que cortan desde 8 a 10 vigas diferentes.

En la siguiente tabla se muestran los resultados obtenidos para 10 instancias diferentes para cada uno de los tres tipos de secuencias. Observamos que **los valores óptimos de *MOS* alcanzados por el algoritmo son bastante cercanos al límite inferior impuesto por el problema** (límite definido por el patrón de corte) y en especial cuando el número de vigas diferentes que cortan los patrones es bajo (tres vigas). En cualquier caso, se comprueba cómo el procedimiento permite obtener secuencias cuyos valores de *MOS* son cercanos a la cota inferior delimitada por el propio patrón de corte. Lo que sin duda es un dato muy útil de cara a plantear en un futuro la **resolución integrada del Problema Conjunto (Corte & Secuenciación)**.

*Secuencias de patrones con pocos vigas cortados (**hasta 3**)*			*Secuencias de patrones con vigas cortados entre **3 y 5***			*Secuencias de patrones con vigas cortados entre **8 y 10***		
Patrón que más vigas corta	*Valor de MOS conseguido por el algoritmo*	*Iteración en la que se alcanza el óptimo*	***Patrón que más vigas corta***	*Valor de MOS conseguido por el algoritmo*	*Iteración en la que se alcanza el óptimo*	***Patrón que más vigas corta***	*Valor de MOS conseguido por el algoritmo*	*Iteración en la que se alcanza el óptimo*
3	3	10	3	5	350	8	11	103
3	3	8	4	7	405	8	12	8
3	3	7	5	6	450	8	12	8
3	3	18	3	5	194	9	13	6
3	3	14	3	6	378	10	11	1
3	3	137	3	5	169	9	12	42
3	3	54	4	7	442	8	12	37
3	3	16	3	6	126	8	11	18
3	3	9	3	6	151	8	12	1
3	3	14	4	7	302	10	11	21

Tabla 6.6- Desempeño del algoritmo para el MOSP

6.4.3 Coste computacional

Si bien el problema de secuenciación de patrones, al igual que ocurría con el de corte de perfiles, es del tipo *off-line* y por tanto el coste computacional tampoco supone un aspecto crítico para el algoritmo, en a tabla 6.7 se muestran los tiempos medios y el tiempo medio por iteración del algoritmo genético para la resolución de los problemas de secuenciación MOSP, MORP y el multiobjetivo. Al igual que en el capítulo anterior, los algoritmos se han implementado con el software comercial *®MATLAB* en su versión *2007b* sobre un procesador *®Intel Core 2 Duo @2.00GHz*.

	Secuencias 10 patrones		Secuencias 20 patrones		Secuencias 30 patrones	
	Tiempo (s)	**Tiempo por iteración (ms)**	**tiempo (s)**	**Tiempo por iteración (ms)**	**tiempo (s)**	**Tiempo por iteración (ms)**
MORP	5,2923	10,5845	6,9354	13,8708	8,2732	16,5464
MOSP	22,1290	44,2581	30,5124	61,0249	78,3290	156,6580
Multiobjetivo	27,2237	54,4474	38,3877	76,7754	83,8343	167,6685

Tabla 6.7- Costes computacionales

Como se puede observar en todos los casos se obtienen tiempos razonablemente bajos. Se comprueba que a medida que se incrementa el número de patrones de la secuencia, también aumenta el coste computacional. Por otro lado, y como se apuntaba en el apartado 6.3.2, el cálculo del *MOS* supone un mayor coste computacional que el cálculo del *AOS* (ver códigos asociados) lo que se traduce en que la resolución del problema MORP es más rápida que la del MOSP. De igual forma, al combinar ambos códigos para la resolución del problema multiobjetivo, los tiempos de ejecución son aproximadamente la suma de los anteriores (MORP y MOSP). Como ocurría con el problema de corte, esto supone un valor añadido al uso de algoritmos genéticos en la secuenciación de patrones.

6.5 UNA METODOLOGÍA PARA LA RESOLUCIÓN DEL PROBLEMA GLOBAL DE CORTE Y SECUENCIACIÓN

A la luz de los resultados obtenidos en la sección anterior sobre lo próximos que se encuentran los valores obtenidos de *MOS* al valor máximo de cantidad de vigas diferentes cortadas de entre todos los patrones de la secuencia, podemos pensar en proponer una metodología que **integre la resolución del problema de corte con la del problema de secuenciación** similar a la del primer enfoque de los propuestos en (Pileggi 2002), en la que va prohibiendo el uso de los patrones de corte con mayor cantidad de vigas diferentes.

Así pues la metodología propuesta para la resolución del problema conjunto o **Problema global de Corte y Secuenciación c**ontendrá todos los algoritmos propuestos hasta ahora (capítulos 5 y 6) como se muestra en el siguiente diagrama, pero añadiendo una restricción de cantidad máxima de vigas diferentes cortadas por patrón que deben cumplir todos los patrones generados en la fase 1 de generación de patrones, como se indica en la siguiente figura.

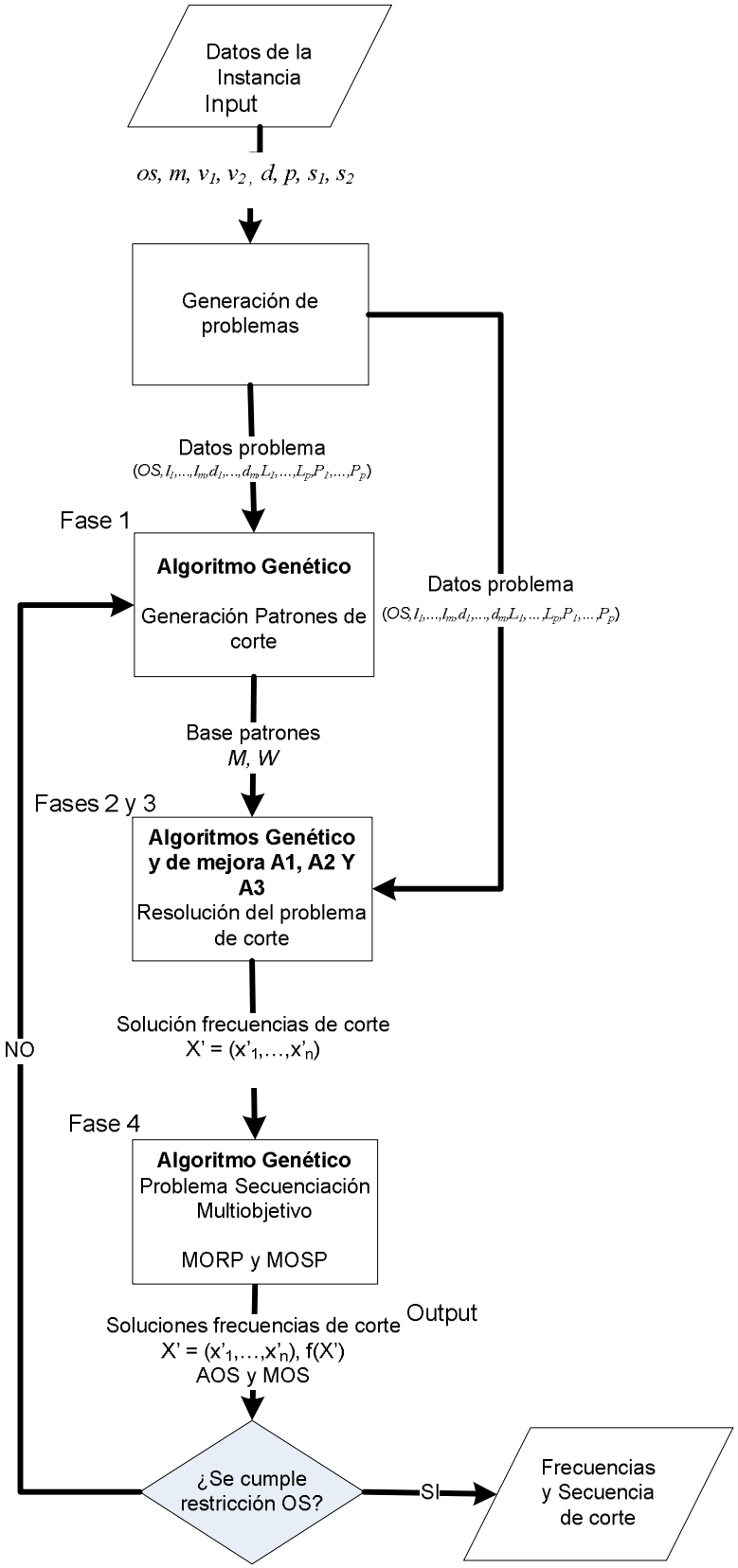

Figura 6.5- Metodología para la resolución del Problema Global (Corte & Secuenciación)

- Conocido un **valor máximo de paquetes abiertos** (*OS*) admisible para la secuencia de corte y que vendrá definido por las restricciones de espacio al final de la línea que puede tener la secuencia de corte. Por tanto a la información de entrada de la metodología habrá que incorporar el dato del máximo *OS*.

  Se aplica la metodología descrita en el **capítulo 5** pero en este caso se limitará la fase de generación de patrones únicamente a aquellos que corten una cantidad de vigas diferentes de valor inferior o igual al valor máximo de *OS*, para lo que se penalizará aquellos patrones con valores superiores al *OS* máximo. Así pues se debe redefinir la función de eficiencia del algoritmo de generación de patrones (fase 1) descrito en 5.2.1 para que a parte de tener en cuenta los restos generados, también se evalúen los patrones de acuerdo con la cantidad de vigas diferentes que corta. En aquellos casos en los que el patrón tenga un valor superior al *OS* máximo, se penaliza para que no se incorpore a la base de patrones que utiliza el algoritmo genético para resolver el problema de corte (fase 2).

- Se aplican el resto de fases de la metodología y se obtiene una solución al Problema de Corte que únicamente contendrá patrones factibles desde el punto de vista de los paquetes diferentes que contienen.

- Por último se aplica el procedimiento descrito en el presente capítulo para el problema de secuenciación.

- Si la solución obtenida no cumple la restricción de OS, se vuelve a la fase 1 endureciendo la restricción de generación de patrones un entero por debajo del valor del *OS* y se repite el proceso hasta encontrar una solución factible.

CAPÍTULO 7

CONCLUSIONES Y TRABAJOS FUTUROS

Cuando se inició el trabajo de investigación descrito en esta memoria, los objetivos eran básicamente dos:

- Desarrollar una metodología que permitiera resolver el problema del corte de perfiles estructurales identificado en la empresa objeto de estudio, empleando para ello técnicas basadas en algoritmos genéticos.

- Proponer un algoritmo, también basado en métodos evolutivos, que permitiera resolver el problema de secuenciación de patrones de corte considerando simultáneamente más de un objetivo.

A la vista de los resultados obtenidos a lo largo del presente trabajo, podemos decir que los objetivos propuestos se han logrado y además se puede concluir con lo siguiente:

1. La identificación de problemas vinculados a la planificación de la producción de vigas, ha evidenciado, como demuestra el análisis del caso real de una empresa del Sector, la necesidad de métodos operativos que resuelvan satisfactoriamente el corte de perfiles estructurales

2. La metodología basada en algoritmos genéticos desarrollada en la presente tesis, se ha mostrada efectiva (elimina sobreproducciones, demandas insatisfechas, y niveles altos de desperdicio) y consistente, tras aplicarla con éxito a la simulación de casos generados de modo aleatorio.

3. La economía en el tiempo de computación, así como la menor complejidad en la estructura, que se ha logrado con los algoritmos propuestos, hacen viable y recomendable su uso en la planificación de corte de perfiles incluso en pequeñas empresas del Sector.

4. Así mismo, se ha probado que las restricciones de espacio de almacenamiento de perfiles, pueden incorporarse a la planificación del proceso de secuenciación de cortes con los algoritmos desarrollados en el capítulo sexto de esta tesis.

Los resultados obtenidos nos animan a trabajar en un futuro inmediato en nuevas líneas de investigación que permitan extender el trabajo desarrollado en esta memoria con el fin de ampliar su ámbito de aplicación. Se tiene previsto trabajar en:

- Estudio y validación de métodos de resolución que contemplen los problemas de Corte y Secuenciación de manera global, como la metodología que se esboza en el último apartado del capítulo 6.

- Planteamiento y resolución del problema de minimización del número de patrones de corte diferentes con el fin de poder realizar cortes simultáneos sobre varios perfiles (apilabilidad de perfiles en la máquina de corte) y poder potenciar la reutilización de puntas de producción.

BIBLIOGRAFÍA

ABOUDI, R. and BARCIA, P., 1998. Determining cutting stock patterns when defects arepresent. *Annals of Operations Research,* **82**, 343-354.

ADAMOWICZ, M. and ALBANO, A., 1976. Nesting two-dimensional shapes in rectangular modules. *Computer-Aided Design,* **8**(1), 27-33.

ÁLVAREZ-VALDÉS, R., PARAJÓN, A. and TAMARIT, J.M., 2002. A tabu search algorithm for large-scale guillotine (un)constrained two-dimensional cutting problems. *Computers & Operations Research,* **29**(7), 925-947.

ÁLVAREZ-VALDÉS, R., PARREÑO, F. and TAMARIT, J., 2009. A branch and bound algorithm for the strip packing problem. *OR Spectrum,* **31**(2), 431-459.

ÁLVAREZ-VALDÉS, R., PARREÑO, F. and TAMARIT, J.M., 2008. Reactive GRASP for the strip-packing problem. *Computers & Operations Research,* **35**(4), 1065-1083.

ÁLVAREZ-VALDÉS, R., PARREÑO, F. and TAMARIT, J.M., 2007. A tabu search algorithm for a two-dimensional non-guillotine cutting problem. *European Journal of Operational Research,* **183**(3), 1167-1182.

ALVES, C. and VALÉRIO DE CARVALHO, J.M., 2008. A stabilized branch-and-price-and-cut algorithm for the multiple length cutting stock problem. *Computers & Operations Research,* **35**(4), 1315-1328.

AMARAL, A. and WRIGHT, M.B., Experiments with a strategic oscillation algorithm for the pallet loading problem. *International Journal of Production Research,* **39(11)**, 2341-2351.

ANAND, S., MCCORD, C., SHARMA, R. and BALACHANDER, T., 1999. An integrated machine vision based system for solving the nonconvex cutting stock problem using genetic algorithms. *Journal of Manufacturing Systems,* **18**(6), 396-415.

ANDONOV, R., POIRRIEZ, V. and RAJOPADHYE, S., 2000. Unbounded knapsack problem: Dynamic programming revisited. *European Journal of Operational Research,* **123**, 394-407.

ANDREATTA, G., BASSO, A., CAUMO, A. and DESERTI, L., 1989. Un problema min cutwidth generaliz- zato e sue applicazionni ad un FMS. Atti delle giornate di lavoro. *AIRO,* , 1-17(en italiano).

ANTONIO, J., CHAUVET, F., CHU, C. and PROTH, J., 1999. The cutting stock problem with mixed objectives: Two heuristics based on dynamic programming. *European Journal of Operational Research,* **114**(2), 395-402.

ARMBRUSTER, M., 2002. A solution procedure for a pattern sequencing problem as part of a one-dimensional cutting stock problem in the steel industry. *European Journal of Operational Research,* **141**(2), 328-340.

BARKER, B.S., COFFMAN, E.G. and RIVEST, R.L., 1980. Orthogonal packings in two dimensions. *SIAM Journal on Computing,* **9**, 846-855.

BEASLEY, J.E., 1985. An algorithm for the two-dimensional assortment problem. *European Journal of Operational Research,* **19**(2), 253-261.

BECCENERI, J.C., YANASSE, H.H. and SOMA, N.Y., 2004. A method for solving the minimization of the maximum number of open stacks problem within a cutting process. *Computers & Operations Research,* **31**(14), 2315-2332.

BELOV, G. and SCHEITHAUER, G., 2006. A branch-and-cut-and-price algorithm for one-dimensional stock cutting and two-dimensional two-stage cutting. *European Journal of Operational Research,* **171**(1), 85-106.

BELOV, G. and SCHEITHAUER, G., 2002. A cutting plane algorithm for the one-dimensional cutting stock problem with multiple stock lengths. *European Journal of Operational Research,* **141**(2), 274-294.

BENNELL, J.A. and DOWSLAND, K.A., 2001. Hybridising Tabu Search with Optimisation Techniques for Irregular Stock Cutting. *Management Science,* **47**, 1160-1172.

BISCHOFF, E.E., JANETZ, F. and RATCLIFF, M.S.W., 1995. Loading pallets with non-identical items. *European Journal of Operational Research,* **584**, 681-692.

BISCHOFF, E.E. and DOWSLAND, W.B., 1982. An application of the micro to product design and distribution. *Journal of the Operational Research Society,* **3**, 271-280.

BORTFELDT, A. and MACK, D., 2007. A heuristic for the three-dimensional strip packing problem. *European Journal of Operational Research,* **183**(3), 1267-1279.

BOUNSAYTHIP, C., MAOUCHE, S. and NEUS, M., 1995. Evolutionary Search Techniques Application to Automated Lay-Planning Optimization Problem, 22-25 Octubre 1995, pp4497-4503.

BURKE, E.K. and KENDALL, G., 1999. Comparison of meta-heuristic algorithms for clustering rectangles. *Computers and Industrial Engineering,* **37 (1-2)**, 383-386.

CABALLÉ PINÓS, C., 1975. El sector siderúrgico español. 1975. Pamplona: Eunsa.

CHRISTOFIDES, N., 1976. *Worst-case analysis of a new heuristic for the travelling salesman problem.* 388. Pittsburgh, PA: Carnegie-Mellon University.

CHRISTOFIDES, N. and HADJICONSTANTINOU, E., 1995. An exact algorithm for orthogonal 2-D cutting problems using guillotine cuts. *European Journal of Operational Research,* **83**(1), 21-38.

CHRISTOFIDES, N., MINGOZZI, A. and TOTH, P., 1979. Combinatorial Optimization. In: N. CHRISTOFIDES, A. MINGOZZI, P. TOTH and C. SANDI, eds, Chichester: Wiley, pp. 339-369.

CHRISTOFIDES, N. and WHITLOCK, C., 1977. An algorithm for two- dimensional cutting problems. *Operations research,* **25**, 31-44.

CLARKE, G. and WRIGHT, J.W., 1964. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research,* **12**, 568-581.

COELLO COELLO, C.A., 1998. A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. *Knowledge and Information Systems. An International Journal,* **1(3)**, 269-308.

COFFMAN, E.G., GAREY, D.S. and JOHNSON, D.S., 1984. Approximation Algorithms for Bin-Packing,: An updated survey, in Algorithm Design for Computer Systems Design, In: G. AUSIELLO, M. LUCERTINI and P. SERAFINI, eds, New York: Springer-Verlag, pp. 49-106.

COFFMAN, E.G., GAREY, D.S. and TARJAN, R.E., 1980. Performance bounds for level-oriented two-dimensional Packing Algorithms. *SIAM Journal on Computing,* **9 (4)**, 808-826.

COFFMAN, E.G., JOHNSON, D.S., SHOR, P.W. and WEBER, R.R., 1997. Bin packing with discrete item sizes, part II: Tight bounds on First Fit. *Random Structures and Algorithms,* **10**(1-2), 69-101.

CONFEMETAL, confemetal, confederación española de organizaciones empresariales del metal. Available: http://www.confemetal.es/.

CORREIA, M.H., OLIVEIRA, J.F. and FERREIRA, J.S., 2001. A New Upper Bound for the Cylinder Packing Problem. *International Transactions in Operational Research,* **8**(5), 571-583.

CRISPIN, A.J., CLAY, P., TAYLOR, G.E., BAYES, T. and REEDMAN, D., 2003. Genetic algorithms applied to leather lay plan material utilisation. *IMechE PartB: Journal of Engineering,* **217**, 1753-1756.

CUI, Y., 2006. Generating optimal multi-segment cutting patterns for circular blanks in the manufacturing of electric motors. *European Journal of Operational Research,* **169**(1), 30-40.

CUI, Y., 2006. Simplest optimal cutting patterns for equal rectangles. *Operations Research Letters,* **34**(6), 630-638.

CUI, Y., GU, T. and HU, W., 2009. A cutting-and-inventory control problem in the manufacturing industry of stainless steel wares. *Omega,* **37**(4), 864-875.

CUI, Y. and LU, Y., 2009. Heuristic algorithm for a cutting stock problem in the steel bridge construction. *Computers & Operations Research,* **36**(2), 612-622.

CUNG, V., HIFI, M. and LE CUN, B., 2000. Constrained two-dimensional cutting stock problems a best-first branch-and-bound algorithm. *International Transactions in Operational Research,* **7**(3), 185-210.

DALMAU, J.I., Análisis estratégico de los sectores industriales y del turismo en la Comunidad Valenciana. **2**.

DAVIES, A.P. and BISCHOFF, E.E., 1999. Weight distribution considerations in container loading. *European Journal of Operational Research,* **114**, 509-527.

DAZA, V.P., MUÑOZ, R. and GÓMES DE ALVARENGA, A., 1995. A Hybrid Genetic Algorithm for the Two-Dimensional Guillotine Cutting Problem. *Evolutionary Algorithms in Management Applications,* , 183-196.

(De Cani. 1979) DE CANI, P., 1979. Packing problems in theory and practice. Phd Thesis. Department of Engineering Production, University of Birmingham.

DORI, D. and BEN-BASSAT, M., 1984. Efficient nesting of congruent convex figures. *Image Process. Comput. Vision,* **27**, 228–235.

DORI, D. and BEN-BASSAT, M., 1983. Circumscribing a convex polygon by a polygon of fewer sides with minimal area addition. *Computer Vision, Graphics, and Image Processing,* **24**(2), 131-159.

DOWSLAND, K.A., 1996. Simple tabu tresholding and the pallet loading problem. In: I.H. OSHMAN and J.P. KELLY, eds, *Metaheuristics: theory and applications.* Kluwer Academic Publishers, pp. 379-406.

DOWSLAND, K.A., 1993. Simulated Annealing in modern heuristic techniques for combinatorial problems. Oxford: Blackwell.

DOWSLAND, K.A., 1987. An exact algorithm for the pallet loading problema. *European Journal of Operational Research,* **31**, 78-84.

DOWSLAND, K.A., 1985. Determining an upper bound for a class of rectangular packing problems. *Computers & Operations Research,* **12**(2), 201-205.

DOWSLAND, K.A. and DOWSLAND, W.B., 1995. Solution approaches to irregular nesting problems. *European Journal of Operational Research,* **84**, 506-521.

DOWSLAND, K.A. and DOWSLAND, W.B., 1992. Packing problems. *European Journal of Operational Research,* **56**(1), 2-14.

DREXL, A., 1988. A simulated annealing approach to the multi-constraint zero-one knapsack problem. *Computing,* **40**, 1-8.

DYCKHOFF, H., 1990. A typology of cutting and packing problems. *European Journal of Operational Research,* **44**(2), 145-159.

DYCKHOFF, H., 1981. A new linear programming approach to the cutting stock problem. *Operations research,* **29**, 1092-1104.

DYCKHOFF, H., KRUSE, H., ABEL, D. and GAL, T., 1985. Trim loss and related problems. *Omega,* **13**(1), 59-72.

DYSON, R.G. and GREGORY, A.S., The cutting stock problem in the flat glass industry. *Operational Research Quarterly,* **25**, 41-53.

EISEMANN, K., 1957. The trim problem. *Management Science,* **3**, 279-284.

FAGGIOLI, E. and BENTIVOGLIO, C.A., 1998. Heuristic and exact methods for the cutting sequencing problem. *European Journal of Operational Research,* **110**(3), 564-575.

FAYARD, D., HIFI, M. and ZISSIMOPOULOS, V., 1998. An efficient approach for large-scale two-dimensional guillotine cutting stock problems. *Journal of the Operational Research Society,* **49**, 1270-1277.

FEO, T. and RESENDE, M.G.C., 1995. Greedy Randomized Adaptive Search Procedures. *Journal of Global Optimization,* **2**, 1-27.

FEO, T. and RESENDE, M.G.C., 1989. A probabilistic heuristic for a computational difficult set covering problems. *Operations Research Letters,* **8**, 67-71.

FINK, A. and VOB, S., 1999. *Applications of modern heuristic search methods to continuous flow-shop scheduling problems. Technical report,* Germany: Technische Universitat Braunschweig.

FOERSTER, H. and WÄSCHER, G., 2000. Pattern reduction in one dimensional cutting stock problems. *International Journal of Production Research,* **7**, 1657-1676.

FOERSTER, H. and WÄSCHER, G., 1998. Simulated annealing for order spread minimization in sequencing cutting patterns. *European Journal of Operational Research,* **110**(2), 272-281.

FRENK, J. and GALAMBOS, G., 1987. Hybrid next-fit algorithm for the two-dimensional rectangle bin-packing problem. *Computing,* **39**(3), 201-217.

G, Y., SEONG, Y. and KANG, M., 2003. A best-first branch and bound algorithm for unconstrained two-dimensional cutting problems. *Operations Research Letters,* **31**(4), 301-307.

GAREY, M.R. and JOHNSON, D.S., 1979. Computers and intractability: A Guide to the Theory of NP-Completeness. Nueva York: W. H. Freeman & Company.

GAU, T. and WÄSCHER, G., 1995. CUTGEN1: A problem generator for the standard one-dimensional cutting stock problem. *European Journal of Operational Research,* **84**(3), 572-579.

GIANNELOS, N.E. and GEORGIADIS, M.C., 2001. Scheduling of Cutting-Stock Processes on Multiple Parallel Machines. *Chemical Engineering Research and Design,* **79, 7**, 747-753.

GILMORE, P.C. and GOMORY, R.E., 1963. A linear programming approach to the cutting stock problem, PartII. *Operations research,* **11**, 863-888.

GILMORE, P.C. and GOMORY, R.E., 1961. A linear programming approach to the cutting stock problem. *Operations research,* **9**, 849-859.

GLOVER, F., 1999. A Template for Scatter Search and Path Relinking, in Artificial Evolution, Lecture Notes in Computer Science, J. HAO, E. LUTTON, E. RONALD, M. SCHOENAUER and D. SNYERS, eds. In: 1999, Springer-Verlag pp13-54.

GLOVER, F., 1990. Tabu Search: Part II. *ORSA Journal on Computing,* , 223-254.

GLOVER, F., 1989. Tabu Search: Part I. *ORSA Journal on Computing,* **1**, 190-206.

GLOVER, F. and KOCHENBERGER, G., 2003. Handbook of Metaheuristics. Dordrecht: Kluwer Academic Publishers.

GLOVER, F., LAGUNA, M. and MARTI, R., 2000. Fundamentals of Scatter Search and Path Relinking. *Control and cybernetics,* **29(3)**, 653-684.

GOLDBERG, D.E., 1989. Genetic algorithms in search, optimization and machine learning.

GOMES, A.M. and OLIVEIRA, J.F., 2006. Solving Irregular Strip Packing problems by hybridising simulated annealing and linear programming. *European Journal of Operational Research,* **171**(3), 811-829.

GOMES, A.M. and OLIVEIRA, J.F., 2002. A 2-exchange heuristic for nesting problems. *European Journal of Operational Research,* **141**(2), 359-370.

GÓMEZ, A. and de la Fuente, D., 2000. Resolution of strip-packing problems with genetic algorithms. *Journal of the Operational Research Society*, **51**, 1289-1295.

GOMORY, R.E., 1963. *An Algorithm for Integer Solutions to Linear Programs.* Nueva York: Mc GrawHill.

GOULIMIS, C., 1990. Optimal solutions for the cutting stock problem. *European Journal of Operational Research,* **44**, 197-208.

GRADISAR, M., KLJAJIC, M., RESINOVIC, G. and JESENKO, J., 1999. A sequential heuristic procedure for one-dimensional cutting. *European Journal of Operational Research,* **114**(3), 557-568.

GRADISAR, M., RESINOVIC, G. and KLJAJIC, M., 2002. Evaluation of algorithms for one-dimensional cutting. *Computers & Operations Research,* **29**(9), 1207-1220.

GRADISAR, M. and TRKMAN, P., 2005. A combined approach to the solution to the general one-dimensional cutting stock problem. *Computers & Operations Research,* **32**(7), 1793-1807.

HAESLER, R.W., Operations Research. Controlling cutting pattern changes in one dimensional trim problems. *1975,* **23 (3)**, 483-493.

HAESLER, R.W., 1992. One-dimensional cutting stock problems and solution procedures. *Mathematical and Computer Modelling,* **16**(1), 1-8.

HAESLER, R.W., 1979. Solving the two-stage cutting stock problem. *Omega,* **7**(2), 145-151.

HAESLER, R.W., 1971. A heuristic programming solution to a non linear cutting stock problem. *Management Science,* **17**, 793-802.

HAESLER, R.W. and SWEENEY, P.E., 1991. Cutting stock problems and solution procedures. *European Journal of Operational Research,* **54**(2), 141-150.

HAHN, S.G., 1968. On the Optimal Cutting of Defective Sheets. *Operations research,* **16**, 1100-114.

HALL SMITH, A. and DE CANI, P., 1980. An algorithm to optimize the lay out of boxes in pallets. *Journal of Operation Research Society,* **31**, 573-578.

HARJUNKOSKI, I., WESTERLUND, T., PÖRN, R. and SKRIFVARS, H., 1998. Different transformations for solving non-convex trim-loss problems by MINLP. *European Journal of Operational Research,* **105**(3), 594-603.

HEISTERMANN, J. and LENGAUER, T., 1995. The nesting problem in the leather manufacturing industry. *Annals of Operations Research,* **57**, 103-133.

HIFI, M., 1998. Exact Algorithms for the Guillotine Strip Cutting/Packing Problem. *Computers & Operations Research,* **2**, 925-940.

HIFI, M., 1997. The DH/KD algorithm: a hybrid approach for unconstrained two-dimensional cutting problems. *European Journal of Operational Research,* **97**(1), 41-52.

HIFI, M. and M'HALLAH, R., 2004. Approximate algorithms for constrained circular cutting problems. *Computers & Operations Research,* **31**(5), 675-694.

HIFI, M. and OUAFI, R., 1998. A Best-first Branch-and-bound Algorithm for Orthogonal Rectangular Packing Problems. *International Transactions in Operational Research,* **5**(4), 345-356.

HIFI, M. and OUAFI, R., 1997. Best-first search and dynamic programming methods for cutting problems: The cases of one or more stock plates. *Computers & Industrial Engineering,* **32**(1), 187-205.

HINTERDING, R. and KHAN, L., 1995. Genetic Algorithms for Cutting Stock Problems: with and without Contiguity. *Lecture Notes in Computer Science,* **956**, 166-186.

HIRANO, H., 1998. 5 pilares de la fábrica visual: la fuente para la implantación de las 5S. 1ª, 1ª Reimpresión edn.

HOLTHAUS, O., 2002. Decomposition approaches for solving the integer one-dimensional cutting stock problem with different types of standard lengths. *European Journal of Operational Research,* **141**(2), 295-312.

HUNG, M.S. and FISK, J.C., 1979. An algorithm for 0-1 multiple knapsack problems. *Naval Research Logistics Quarterly,* **25 (3)**, 571-579.

JAKOBS, S., 1996. On genetic algorithms for the packing of polygons. *European Journal of Operational Research,* **88**, 165-181.

JOHNSON, D.S. and MCGEOCH, L.A., 1997. The Travelling Salesman Problem: A case study in Local optimization. In: E.H.L. AARTS and J.K. LENSTRA, eds, *Local Search in Combinatorial Optimization.* John Wiley and Sons, pp. 215-310.

JOHNSON, M.P., RENNICK, C. and ZAK, E.J., 1997. Skiving addition to the cutting stock problem in the paper industry. *SIAM Review,* **39,3**, 472-483.

JOHNSTON, R.E., 1986. Rounding algorithm for cutting stock problems. *Journal of Asian-Pacific Operations Research Societies,* **3**, 166-171.

JOHNSTON, R.E., 1984. Cutting patterns and cutting schedulers. *ASOR Bulletin,* **4**, 3-13.

JOHNSTON, R.E. and SADINLIJA, E., 2004. A new model for complete solutions to one-dimensional cutting stock problems. *European Journal of Operational Research,* **153**, 176-183.

JÜNGER, M., REINELT, G. and RINALDI, G., 1994. *The Traveling Salesman Problem* 92.113. Cologne, Germany: Angewandte Mathematik und Informatik, Universitat zu Köln,.

KANG, J. and PARK, S., 2003. Algorithms for the variable sized bin packing problem. *European Journal of Operational Research,* **147**(2), 365-372.

KANTOROVIC, L.V., 1960. Mathematical methods of organizing and plannig production (1939), Traducción del ruso al inglés aparecida en. *Management Science,* **6**, 366-422.

KIRKPATRICK, S., GELATT, C.D. and VECCHI, M.P., 1983. Optimization by **Simulated Annealing**. *Science, New Series,* **220**(4598), 671-680.

KOPARDEKAR, P. and MITAL, A., 1999. Cutting stock problem: a heuristic solution based on operators' intuitive strategies. *International Journal of Computer Integrated Manufacturing,* **12**, 371-372.

KOS, L. and DUHOVNIK, J., 2002. Cutting optimization with variable-sized stock and inventory status data. *International Journal of Production Research,* **40**(10), 2289-2301.

LETCHFORD, A.N. and AMARAL, A., 2001. Analysis of upper bounds for the Pallet Loading Problem. *European Journal of Operational Research,* **132**(3), 582-593.

LEUNG, T.W., YUNG, C.H. and TROUTT, M.D., 2001. Applications of genetic search and simulated annealing to the two-dimensional non-guillotine cutting stock problem. *Computers & Industrial Engineering,* **40**(3), 201-214.

LIANG, K., YAO, X., NEWTON, C. and HOFFMAN, D., 2002. A new evolutionary approach to cutting stock problems with and without contiguity. *Computers & Operations Research,* **29**(12), 1641-1659.

LINDEROTH, J.T. and SAVELSBERGH, M.W.P., 1999. "A Computational Study of Search Strategies for Mixed Integer Programming,". *INFORMS Journal on Computing,* **11**, 173-187.

LINHARES, A. and YANASSE, H.H., 2002. Connections between cutting pattern sequencing, VLSI design and flexible machines. *Computers & Operations Research,* **29**, 1759-1772.

LINS, L.L.L., LINS, S.S.L. and MORABITO, R.N., 2003. An L-approach for packing (ℓ, w)-rectangles into rectangular and L-shaped pieces. *Journal of the Operational Research Society,* **54**(7), 777-789.

LINS, L., LINS, S. and MORABITO, R.N., 2002. An n-tet graph approach for non-guillotine packings of n-dimensional boxes into an n-container. *European Journal of Operational Research,* **141**(2), 421-439.

LODI, A., MARTELLO, S. and VIGO, D., 2004. Models and Bounds for Two-Dimensional Level Packing Problems. *Journal of Combinatorial Optimization,* **8**, 363-379.

LODI, A., MARTELLO, S. and VIGO, D., 2002. Heuristic algorithms for the three-dimensional bin packing problem. *European Journal of Operational Research,* **141**, 410-420.

LODI, A., MARTELLO, S. and VIGO, D., 1999. Approximation algorithms for the oriented two-dimensional bin packing problem. *European Journal of Operational Research,* **112**, 158-166.

LODI, A., MARTELLO, S. and VIGO, D., 2002. Recent advances on two-dimensional bin packing problems. *Discrete Applied Mathematics,* **123**(1-3), 379-396.

MACK, D., BORTFELDT, A. and GEHRING, H., 2004. A parallel hybrid local search algorithm for the container loading problem. *International Transactions in Operational Research,* **11**, 511-533.

MARCOTTE, O., 1986. An instance of the cutting stock problem for which the rounding property does not hold. *Operations Research Letters,* **4**(5), 239-243.

MARTELLO, S., MONACI, M. and VIGO, D., 2003. An Exact Approach to the Strip-Packing Problem. *Journal on Computing,* **15**, 310-319.

MARTELLO, S., PISINGER, D. and VIGO, D., 2000. The Three-Dimensional Bin Packing Problem256-267. *Operations Research,* **48**, 256-267.

MARTELLO, S. and TOTH, P., 1990. Knapsack Problems - Algorithms and Computer Implementations. Chichester: John Wiley & Sons.

MARTELLO, S. and TOTH, P., 1981. A bound and bound algorithm for the zero–one multiple knapsack problem. *Discrete Applied Mathematics,* **3**, 275–288.

MARTELLO, S. and VIGO, D., 1998. Exact Solution of the Two-Dimensional Finite Bin Packing Problem. *Management Science,* **44**, 388-399.

MARTI, R., 2003. Multi-Start methods. Handbook of Metaheuristics. In: F. GLOVER and G. KOCHENBERGER, eds, Dordrecht: Kluwer Academic Publishers, pp. 355.

MARTIN, A., 2001. *General mixed-integer programming: computational issues for branch-and-cut algorithms*., 2001, pp1-25.

MENON, S. and SCHRAGE, L., 2002. Order allocation for stock cutting in the paper industry. *Operations research,* **50, 2**, 324-332.

MILLER, C., TUCKER, A. and ZEMLIN, R., 1960. Integer programming formulations and travelling salesman problems. *J.A.C.M,* **7**, 326-329.

MORABITO, R.N. and ARENALES, M.N., 2000. Optimizing the cutting of stock plates in a furniture indutry. *International Journal of Production Research,* **38, 12**, 2725-2742.

MORABITO, R.N. and GARCIA, V., 1998. The cutting stock problem in a hardboard industry: a case study. *Computers & Operations Research,* **25**(6), 469-485.

MOSCATO, P. and COTA, C., 2003. An Introduction to memetic algorithms. *Revista Iberoamericana de Inteligencia Artificial,* **19**, 131-148.

MUNIZ DE OLIVEIRA, A.C. and NOGUEIRA LORENA, L.A., 2002. 2-Opt Population Training for Minimization of Open Stack Problem. *Lecture Notes in Computer Science,* **2507**, 497-504.

(Nemhauser and Wolsey. 1988) NEMHAUSER, G.L. and WOLSEY, L., 1988. Integer and combinatorial optimization. Nueva York: John Wiley and sons.

NITSCHE, C., SCHEITHAUER, G. and TERNO, J., 1999. Tighter relaxations for the cutting stock problem. *European Journal of Operational Research,* **112**(3), 654-663.

PARREÑO, F., ÁLVAREZ-VALDÉS, R. and TAMARIT, J., A hybrid GRASP/VND algorithm for two- and three-dimensional bin packing. *Annals of Operations Research,* , 1-18.

PEETERS, M. and DEGRAEVE, Z., 2006. Branch-and-price algorithms for the dual bin packing and maximum cardinality bin packing problem. *European Journal of Operational Research,* **170**(2), 416-439.

PEETERS, M. and DEGRAEVE, Z., 2006. An linear programming based lower bound for the simple assembly line balancing problem. *European Journal of Operational Research,* **168**(3), 716-731.

PIERCE, J.F., 1969. Direct search algorithms for truck-dispatching problems. *Transportation Research,* **3**(1), 1-42.

PIERCE, J.F., 1964. Some Large-Scale Production Scheduling Problems in the paper industry. Englewood Cliffs, NJ: Prentice Hall.

PILEGGI, G.C.F., 2002. *Approaches for the integrated optimization of pattern generation and sequencing problems*, University of Sao Paulo.

POLDI, K.C. and ARENALES, M.N., 2009. Heuristics for the one-dimensional cutting stock problem with limited multiple stock lengths. *Computers & Operations Research,* **36**(6), 2074-2081.

PUCHINGER, J., RAIDL, G.R. and KOLLER, G., 2004. Solving a real-world glass cutting problem, J. GOTTLIEB and G.R. RAIDL, eds. In: April 2004 2004, LNCS pp162-173.

RAMESH BABU, A. and RAMESH BABU, N., 2001. A generic approach for nesting of 2-D parts in 2-D sheets using genetic and heuristic algorithms. *Computer-Aided Design,* **33**(12), 879-891.

RESENDE, M.G.C., RIBEIRO, C., GLOVER, F. and MARTI, R., 2009. Scatter Search and Path Relinking: Fundamentals, advances and applications.Michel Gendreau and Jean-Yves Potvin (Eds.). In: M. GENDREAU and J.Y. POTVIN, eds, ***Handbook of Metaheuristics (2nd Edition)***. Springer, .

RIETZ, J., SCHEITHAUER, G. and TERNO, J., 2002. Families of non-IRUP instances of the one-dimensional cutting stock problem. *Discrete Applied Mathematics,* **121**(1-3), 229-245.

RINALDI, F. and FRANZ, A., 2007. A two-dimensional strip cutting problem with sequencing constraint. *European Journal of Operational Research,* **183**(3), 1371-1384.

SANCHEZ, N., GIL, J.I. and PALACIOS, D.:, 1999. Importancia y estado actual del sector metal-mecánico en la Comunidad Valenciana. *Trabajo, Economía y Sociedad,* .

SCHAFFER, J.D., 1985. Multiple objective optimization with vector evaluated genetic algorithms, 1985, Lawrence Erlbaum.

SCHEITHAUER, G. and TERNO, J., 1997. Theoretical investigations on the modified integer round-up property for the one-dimensional cutting stock problem. *Operations Research Letters,* **20**(2), 93-100.

SCHEITHAUER, G. and TERNO, J., 1995. The modified integer round-up property of the one-dimensional cutting stock problem. *European Journal of Operational Research,* **84**(3), 562-571.

STADTLER, H., 1990. A one-dimensional cutting stock problem in the aluminium industry and its solution. *European Journal of Operational Research,* **44**(2), 209-223.

STEUDEL, H.J., 1984. Generating pallet loading patterns with considerations of item stacking on end and side surfaces. *Journal of Manufacturing Systems,* **3**(2), 135-143.

STEUDEL, H.J., 1979. Generating pallet loading patterns: a special case of the two-dimensional cutting stock problem. *Management Science,* **25**, 997-1004.

SULIMAN, S.M.A., 2001. Pattern generating procedure for the cutting stock problem. *International Journal of Production Economics,* **74**(1-3), 293-301.

SWEENEY, P.E. and HAESLER, R.W., 1990. One-dimensional cutting stock decisions for rolls with multiple quality grades. *European Journal of Operational Research,* **44**(2), 224-231.

SWEENEY, P.E. and PATERNOSTER, E.R., 1992. Cutting and packing problems: A Categorised, Application- Orientated Research Bibliography. *Journal of the Operational Research Society,* **43**, 691-706.

TAKAHARA, S., KUSUMOTO, Y. and MIYAMOTO, S., 2003. Solution for textile nesting problems using adaptive meta-heuristics and grouping. *Soft Computing - A Fusion of Foundations, Methodologies and Applications,* **7**(3), 154-159.

UMETANI, S., YAGIURA, M. and IBARAKI, T., 2003. One-dimensional cutting stock problem to minimize the number of different patterns. *European Journal of Operational Research,* **146**(2), 388-402.

VAHRENKAMP, R., 1996. Random search in the one-dimensional cutting stock problem. *European Journal of Operational Research,* **95**(1), 191-200.

VALÉRIO DE CARVALHO, J.M., 2002. LP models for bin packing and cutting stock problems. *European Journal of Operational Research,* **141**(2), 253-273.

VALÉRIO DE CARVALHO, J.M., 1998. Exact solution of cutting stock problems using column generation and branch-and-Bound. *International Transactions in Operational Research,* **5**(1), 35-44.

VANCE, P.H., BARNHART, C., JOHNSON, E.L. and NEMHAUSER, G.L., 1994. *Solving binarycutting stock problems by column generation and branch-and-bound*. *Comput. Opt. Appl.,* **3**, 111-130.

VANDERBECK, F., 2000. Exact algorithm for minimizing the number of set ups in the one dimensional cutting stock problems. *Operations research,* **48**(6), 915-926.

VANDERBECK, F., 1999. Computational study of a column generation algorihm for bin and cutting stock problems. *Mathematical Programming,* **A 86**(3), 565-594.

VANDERBECK, F. and SAVELSBERGH, M.W.P., 2006. A generic view of Dantzig–Wolfe decomposition in mixed integer programming. *Operations Research Letters,* **34**(3), 296-306.

VANDERBECK, F. and WOLSEY, L.A., 1996. An exact algorithm for IP column generation. *Operations Research Letters,* **19**(4), 151-159.

VASKO, F.J., NEWHART, D.D. and STOTT, J.,KENNETH L., 1999. A hierarchical approach for one-dimensional cutting stock problems in the steel industry that maximizes yield and minimizes overgrading. *European Journal of Operational Research,* **114**(1), 72-82.

WAGNER, B.J., 1999. A genetic algorithm solution for one-dimensional bundled stock cutting. *European Journal of Operational Research,* **117**(2), 368-381.

WÄSCHER, G., HAUßNER, H. and SCHUMANN, H., 2007. An improved typology of cutting and packing problems. *European Journal of Operational Research,* **183**(3), 1109-1130.

WHITE, D.J., 1992. A complementary greedy heuristic for the knapsack problem. *European Journal of Operational Research,* **62**(1), 85-95.

YANASSE, H.H., 1997. On a **pattern** sequencing problem to minimize the maximum number of open stacks. *European Journal of Operational Research,* **100**, 453-463.

YANASSE, H.H. and PINTO LAMOSA, M.J., 2007. An integrated cutting stock and sequencing problem. *European Journal of Operational Research,* **183**(3), 1353-1370.

YANG, C., SUNG, T. and WENG, W., 2006. An improved tabu search approach with mixed objective function for one-dimensional cutting stock problems. *Advances in Engineering Software,* **37**(8), 502-513.

YUEN, B.J., 1995. Improved heuristics for sequencing cutting patterns. *European Journal of Operational Research,* **87**(1), 57-64.

YUEN, B.J. and RICHARDSON, K.V., 1995. Establishing the optimality of sequencing heuristics for cutting stock problems. *European Journal of Operational Research,* **84**(3), 590-598.

ZUKERMAN, M., JIA, L., NEAME, T. and WOEGINGER, G., 2001. A polynomially solvable special case of the unbounded knapsack problem. *Operations Research Letters,* **29**, 13-16.

ANEXO
IMPLEMENTACIONES EN ®MATLAB

En el presente anexo se describen las diferentes funciones utilizadas para la implementación de las técnicas y componentes de la metodología desarrollada para la resolución del problema de corte (capítulo 5) y del problema de secuenciación (capítulo 6). Dichas funciones se han implementado utilizando el software matemático comercial ®*MATLAB* en su versión R2007b.

X.1 IMPLEMENTACIONES PARA LA RESOLUCIÓN DEL PROBLEMA DE CORTE

X.1.1 Generador de problemas

Se ha creado una función específica llamada `GeneradorProblemas` que a partir los parámetros de instancia devolverá una estructura de datos completa de diferentes problemas de test.

Los datos de la instancia se introducen en un vector con los siguientes parámetros definidos por el usuario[1]:

[1] Se observará que se introducen ligeros cambios en a la notación respecto a utilizada en los capítulos de la tesis por comodidad de implementación de las funciones

```
VecDatos = [P,N,lmin,lmax,d,M,Lmin,Lmax,S]
```

`P:` Cantidad de problemas diferentes a generar

`N:` Tipos diferentes de vigas

`lmin,lmax:` Longitudes máximas y mínimas de las vigas demandadas

`d:` demanda media de cada viga para el periodo de fabricación

`M:` Tipos diferentes de pérfiles en stock

`Lmin,Lmax:` Longitudes máximas y mínimas de los perfiles

`S:` Stock disponible medio de cada perfil en stock

Al llamar a la función `GeneradorProblemas` pasándole los valores de la instancia, ésta devuelve una estructura de datos `X` que contiene tanto los datos de partida de la instancia como los problemas de test generados sobre esta. Los elementos que forman parte de esta estructura son:

`X.Ins:` Datos de la instancia (9 elementos->Los de `VecDatos`)

`X.p:` Problema p (p=1..P)

`X.p.l:` Vector de longitudes de los ítems (vigas) del problema p (dim. N)

`X.p.d:` Vector de demandas de los ítems (vigas) del problema p (dim. N)

`X.p.L:` Vector de longitudes de los objetos (perfiles) del problema p (dim. M)

`X.p.S:` Vector de stocks de los objetos (perfiles) del problema p (dim. M)

`X.p.G:` Vector de número de genes (que forman el cromosoma) asociados a las longitudes de los perfiles del problema p (dim. M)

Con la finalidad de poder aplicar en futuros pasos un algoritmo genético sobre estos datos, la función también devuelve un vector de datos en el que se calcula la cantidad media de genes por perfil (G_m) que debería tener el cromosoma para satisfacer la demanda total de una viga cuya longitud sea la media ponderada en demanda de las longitudes de las vigas demandadas

$$G_m = \left\lceil \frac{L_m}{\frac{\sum_{i=1}^{N} d_i * l_i}{\sum_{i=1}^{N} d_i}} + 0,5 \right\rceil, \qquad m = 1,...,M \tag{5.1}$$

X.1.2 Generador de patrones

Para la implementación de un algoritmo genético que permita la búsqueda de patrones de corte eficientes se ha utiliza la función `GA` del *toolbox* de MATLAB *Genetic Algorithms and Direct Search*. La función GA resuelve problemas del tipo:

$$Min\ F(X)$$

s. a:

$A*X \leq B,\ A_{eq}*X = B_{eq}$ (restricciones lineales)

$C(X) \leq 0,\ C_{eq}(X) = 0$ (restricciones no lineales)

$LB \leq X \leq UP$ (cotas superiores e inferiores, *lower & upper bounds*)

Los parámetros de entrada al ejecutar la función `GA` cuando se trata de un algoritmo genético sin restricciones son la función de evaluación, el número de variables de las soluciones y las opciones para la ejecución del algoritmo.

X = GA(FITNESSFCN,NVARS,options)

Para conseguir la codificación deseada de los patrones y para que estos puedan ser manipulados por la función `GA`, se desarrollan diferentes funciones auxiliares que permiten que la función `GA` pueda realizar las operaciones de Creación, Recombinación, Mutación y Evaluación sobre la población de patrones de corte.

- `CreacionPatrones`: crea una población inicial de patrones para introducir al algoritmo. Cada patrón (cromosoma) está compuesto por un número *G* de genes, tal que se cumple la condición de no negatividad de restos.

- `crossoverscattered2`: realiza la operación de recombinación sobre una población de patrones padre, generando la cantidad de patrones hijo definida por el parámetro *Xfraction*. La selección de los patrones padres se realiza de manera aleatoria sobre la población de la generación actual. Si se crean patrones repetidos estos se obvian y se vuelve a generar de manera aleatoria un nuevo hijo.

- `mutationelimina`: eliminará los individuos con peor valor en la función de evaluación, (*1-Xfraction*) de la población y los reemplazará por individuos nuevos generados aleatoriamente. El uso de esta función garantiza la diversidad de las soluciones.

- `mutationuniform2`: se utiliza a continuación de la anterior y modifica los individuos (*1-Xfraction*) introduciendo cambios aleatorios en sus genes con una probabilidad parametrizada por la tasa *mutationRate*

Las funciones de eficiencia implementadas son las siguientes:

- `trimloss`: permite evaluar la población de patrones en función de la cantidad absoluta de resto que genera cada uno de ellos.

- `trimlossef`: permite evaluar la población de patrones en función de la proporción de resto que genera cada uno de ellos

- `cantitems`: permite evaluar la población de patrones en función de la cantidad total de vigas que cortan

- `cantstacks`: calcula la cantidad total de vigas diferentes que corta cada patrón (útil en el problema de MOSP, capítulo 6)

Estas funciones de eficiencia permiten realizar en un futuro tanto evaluaciones ponderadas de los patrones como determinar frentes de Pareto en el caso del planteamiento de problemas multiobjetivo.

X.1.3 Algoritmo genético de resolución del problema de corte

Se ha creado la función *CreacionSoluciones* que obtiene aleatoriamente soluciones al problema de corte, estas soluciones se codifican en términos de cantidades de cada patrón de forma que sea posible su manipulación por parte del algoritmo genético. Una solución del problema será una secuencia de patrones. Se parte de una base de patrones y de éstos se seleccionan unos cuantos de manera aleatoria. Con el objetivo de mantener la diversidad de las soluciones, se establecen dos valores (máximo y mínimo) entre los que oscilará el número de patrones de cada solución.

```
M= CreacionSoluciones (PopSize,P,S,Gmax,Gmin)
```

`PopSize`: Cantidad de soluciones aleatorias que se van a generar

`P`: vector cantidad total de patrones de cada perfil (dim N)

`S`: vector stock disponible de cada perfil en stock (dim N)

`Gmax y Gmin`: cantidades máximas y mínimas de patrones contendrá la solución (si se hace *Gmax=Gmin*, entonces G será siempre el mismo). Se pueden calcular las cotas superior e inferior en función de los patrones con el mayor y menor número de vigas cortadas, ejecutando la función `cotassoluciones.m`

La función devuelve una estructura `M` contiene las matrices `Y,Z,X` y `G`.

`M.Y`: son las soluciones codificadas en términos de posiciones de patrones, los patrones de todos los perfiles se indexan de manera sucesiva.

`M.Z`: perfil al que corresponde el patrón

`M.X`: son las soluciones codificadas en términos de cantidades de patrones

`M.G`: es un vector columna que indica el valor de la longitud del cromosoma para cada solución i

Se utilizará la función GA pero en este caso además de definir la función de eficiencia y establecer el *NVARS*, también se impondrán restricciones de demandas y almacén.

En este caso el algoritmo genético se aplicará sobre una población inicial (*InitialPopulation*) de soluciones. Al ejecutar el fichero PruebaCreacionSoluciones.m, se llama a la función CreaciónSoluciones a la vez que se obtienen las matrices que formarán las restricciones de demanda y almacén. Se genera una estructura de soluciones (M) para el problema de corte a la vez que se obtienen los niveles de inventario y de almacén de materias primas (M.Inventario y M.Materiaprima). Además, se generan también las matrices Base para el modelo matemático (Base.evaluacion, Base.items y Base.perfiles)

Se ha definido la función de eficiencia Wagnercorte donde, la primera sólo tiene en cuenta los restos de material generados por cada solución en términos absolutos, mientras que la segunda impone penalización por sobreproducción y por no satisfacción de demandas. Así mismo también se han creado unas nuevas funciones (crossoverscattered3 y mutationeliminaproblemacorte) para la recombinación y la mutación que tienen en cuenta la codificación de las soluciones.

Por último, las soluciones obtenidas por el genético se almacenan en la estructura Soluciones.

X.1.4 Algoritmo para la búsqueda de Patrones Incompletos de corte

A partir de los resultados obtenidos en la estructura soluciones, se crea un fichero llamado *Reajustessobreproduccion.m* cuya finalidad es (a partir de un conjunto de K candidatos o soluciones seleccionadas) obtener el conjunto de patrones por candidato que contribuyen a la sobreproducción y en que medida lo hacen. Al ejecutar este fichero, se obtiene la estructura para cada k (k=1:K) que consta de dos partes: *Nuevospatrones.Completos* y *Nuevospatrones.Incompletos*, así como los vectores *PatronIndex* y *numeroetapas*, en los que para cada k se muestran los patrones seleccionados como incompletos.

X.1.5 Algoritmo de Agrupamiento (A1)

La implementación en ®*MATLAB* de este algoritmo se realiza en el fichero algoritmo1.m. Al ejecutarlo, se crea una estructura llamada *NuevaBase1* que contiene los nuevos patrones generados por el algoritmo de agrupamiento (un patrón por solución). Al igual que ocurría con la base de patrones del algoritmo genético, esta estructura consta de las partes *NuevaBase1.evaluacion, NuevaBase1.perfiles, NuevaBase1.items* en las que se almacena la información correspondiente a cada patrón (resto que genera, perfil a partir del que se corta, vigas o vigas que corta).

X.1.6 Algoritmo genético residual (A2)

Este algoritmo ejecutará de nuevo los algoritmos genéticos de la generación de patrones y de resolución del problema de corte para el conjunto de los K candidatos seleccionados. La ejecución de este algoritmo se ha implementado en tres ficheros: *algoritmo21.m, algoritmo22.m y algoritmo23.m*.

En *algoritmo21* se genera la estructura *Probs* que es equivalente a la que obtenía el generador de problemas y que contiene los elementos *.l, .L, .d, .G* y *.S* para cada uno de los candidatos (habrá que ejecutar los genéticos tantas veces como candidatos se hayan seleccionado).

En *algoritmo22* se llama a la función *GA* para ejecutar los algoritmos genéticos de patrones y de corte.

Por último el *algoritmo23* genera, con la mejor de las soluciones obtenidas para cada candidato (satisfaciendo todas las nuevas demandas), una estructura llamada *NuevaBase2* equivalente a la devuelta por el algoritmo de agrupamiento y que contiene toda la información relativa a los patrones obtenidos por el algoritmo.

X.1.7 Algoritmo de longitudes menores (A3)

Para la implementación de este algoritmo, se han desarrollado las funciones *combinaciones.m* y *seleccioncombinaciones.m* que obtienen todas las combinaciones posibles de vigas no producidas (Γ) y la mejor asignación de éstas a los Patrones Incompletos respectivamente. Una vez asignadas todas las vigas no producidas se obtiene la *NuevaBase3*, equivalente a las anteriores pero con las soluciones obtenidas mediante la aplicación del algoritmo de longitudes menores. La implementación de éste se ha desarrollado en los ficheros *algoritmo31.m*, *algoritmo32.m* y *algoritmo3sinImenos.m*

X.1.8 Comparación de los algoritmo (A1, A2 y A3)

Obtenidas las *NuevasBases* para cada algoritmo se comparan y se selecciona en cada caso (k) la mejor alternativa de las 3 y se almacena en la estructura *NuevaBaseMix*. En la estructura *Final* se obtienen las soluciones mejoradas para cada k con su información relativa a restos, patrones e inventarios. Es una estructura equivalente a la estructura *Soluciones* obtenida por el *GACorte*.

X.2 IMPLEMENTACIONES PARA LA RESOLUCIÓN DEL PROBLEMA DE SECUENCIACIÓN

X.2.1 Implementación en ®MATLAB

Al igual que en el punto anterior X.1, la implementación de los algoritmos genéticos propuestos en 6.2.2 y 6.2.3 se ha realizado con el software matemático comercial ®*MATLAB* en su versión R2007b.

En este caso, y por tratarse de algoritmos muy sencillos de implementar, no se ha utilizado ningún *toolbox* específico ni ninguna función predefinida por el software, sino que se han creado funciones concretas.

Previamente a la resolución de los problemas de secuenciación, es necesario convertir las soluciones obtenidas del problema de corte en soluciones manipulables por el algoritmo de secuenciación. A tal efecto se ha desarrollado una función que convierte los datos resultado de los algoritmos de 5.3 en datos de entrada para los algoritmos genéticos de secuenciación. Dicha función está desarrollada en el fichero `matrizdepartida.m` y tiene como salida una estructura `travel` que contiene datos sobre: mejor solución de entre las obtenidas por el genético de corte; matriz de relación entre patrones y vigas, la denominada como P en el apartado 5.1 y cantidad de patrones distintos que formarán la secuencia.

Para la implementación de los tres algoritmos genéticos propuestos en el capítulo 6, se han creado sus tres funciones correspondientes: `mosp_ga.m`, que permitirá resolver el problema de la cantidad máxima de paquetes; `morp_ga.m`, que permitirá resolver el problema de la extensión media de pedido; y la función `integrated_ga.m`, que integra ambos problemas obteniendo un óptimo de pareto como se ha indicado en 5.2.3. En los tres casos los parámetros de entrada a la función son:

- matriz de relación entre patrones y vigas `travel.xy`
- tamaño de la población `pop_size`
- número de iteraciones `num_iter`
- el parámetro de graficado `show_res`

```
[solintegrated]=integrated_ga(xy,pop_size,num_iter,show_res)
    [solucionmorp]=morp_ga(xy,pop_size,num_iter,show_res)
    [solucionmosp]=mosp_ga(xy,pop_size,num_iter,show_res)
```

En caso de no toma ningún valor de tamaño de población o número de iteraciones, la función coge por defecto un tamaño de población de 100 y un número de iteraciones máximo de 1000.

Los resultados se guardan en tres estructuras que contienen datos sobre:

- mejor solución para cada iteración `hist_dist`
- mejor solución global `min_dist`

- Secuencia de los patrones de la mejor solución `opt_rte`
- Representación gráfica de la evolución de los datos del histórico

A continuación y a modo ilustrativo, se muestra una imagen de captura de pantalla del entorno de ®*MATLAB* en el que se han desarrollado las funciones descritas a lo largo del anexo.

El entorno de trabajo de ®*MATLAB* permite manejar en diferentes ventanas los distintos elementos sobre los que se trabaja. Por un lado desde la ventana *CurrentDirectory* se tiene acceso a la exploración de ficheros: *.m*, funciones programadas; *.mat*, datos guardados; etc. En la ventana *Workspace* se van almacenando las variables y estructuras de datos generadas mediante la ejecución de las funciones a lo largo de la sesión de trabajo. A través del *ArrayEditor* se tiene acceso a los valores de los elementos de estas estructuras, matrices o vectores de datos. Por otro lado, las diferentes funciones y ficheros de código se editan en el *Editor*, desde el cual además pueden ejecutarse. En la ventana *CommandWindow* se pueden ejecutar comandos concretos sobre los datos del Workspace.

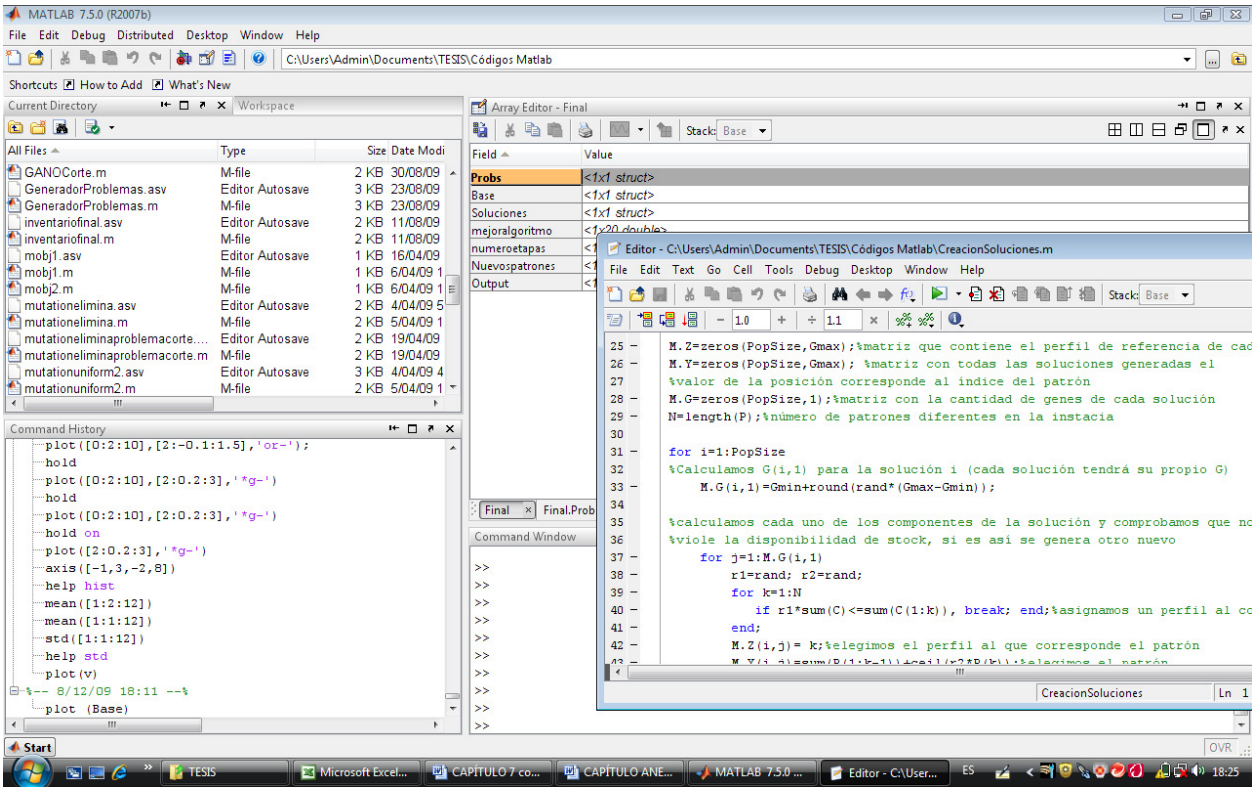

Figura X.1 Entorno de trabajo ®MATLAB