Algoritmo GRASP para cortes de guillotina

Recepción: Agosto de 2006 / Aceptación: Noviembre de 2006

(1) María Ruiz Rivera
(2) Edgar Ruiz Lizama

RESUMEN

El presente trabajo se enfoca en el corte recto de guillotina, el cual debido al alto costo computacional que ocasiona al obtener soluciones exactas, se plantea utilizar un Algoritmo GRASP que permita encontrar buenas soluciones para cualquier instancia y en tiempos adecuados, teniendo como objetivo principal minimizar el residuo o de desperdicio de materiales que se generan en el proceso de corte. Esto permitirá el incremento de la productividad y reducción de costos haciéndolo atractivo para aplicarlo en el sector de la industria del papel, vidrio, metal y madera.

Palabras Clave: GRASP, heurística, parámetro de relajación, cortes en 2D.

A GRASP ALGORITHM FOR GUILLOTINE CUTS

ABSTRACT

This work focus upon the first one, which, due to its high computational cost that causes us to obtain exact solutions, the use of a GRASP Algorithm for straight guillotine cuts, which allows to find good solutions for any instance in suitable times is thought to be used, having as the main goal to minimize the lost or waste of materials generated in the cutting process. This will allow us the enhancement of productivity and the reduction of costs making it attractive for applying it in the paper, glass, metal and wood industry sector.

Key words: GRASP, heuristics, relaxing parameter, 2D cuts.

INTRODUCCIÓN

El problema de corte en dos dimensiones consiste en determinar un conjunto de patrones de corte variables de tal forma que satisfagan los requerimientos de piezas de material solicitados utilizando el menor número de láminas disponibles. Los cortes en dos dimensiones se clasifican por el tipo de herramienta a utilizar como el corte por guillotina y corte no guillotina, considerados cortes rectangulares aplicado en el sector de papel [CM 88] [HEIS 96], vidrio [DG 76], [Fa 83], [Ma 79], metal, madera [VM 92].

El principal objetivo de este problema es reducir la cantidad de residuo (desperdicio) de material que se origina al momento de realizar un determinado corte.

Los cortes de guillotina son rectos y de extremo a extremo de la lámina y siempre paralela a uno de los lados de la misma. En algunos casos se tendrá la necesidad de una rotación de 90° en determinados requerimientos para que sus lados siempre queden paralelos a los de la lámina donde se realizará el corte.

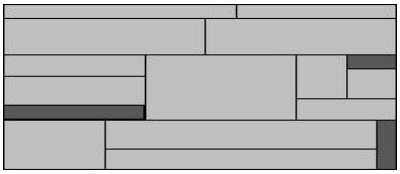
Dentro de este proceso de corte se considerarán las posibles combinaciones de los requerimientos; debido al alto grado de combinaciones que se obtiene, presenta un alto costo de procesamiento y consumo de memoria principal de la computadora. En este sentido el trabajo permitirá construir patrones o esquemas de orden de complejidad lineal, y para este tipo de problema se propone la utilización de un algoritmo metaheurístico GRASP (Greedy randomize adaptative search procedure).

El trabajo desarrolla un algoritmo meta heurístico para la solución del problema de cortes en dos dimensiones. Debido al alto costo computacional que demanda el obtener soluciones exactas, este problema es considerado por la misma algorítmica como NP-difícil, por ello se justifica el desarrollo de la heurística.

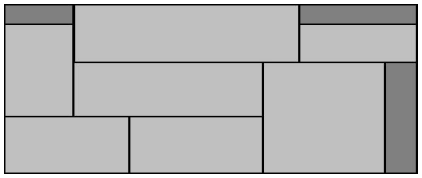
FUNDAMENTACIÓN TEÓRICA

El problema de cortes ha sido ampliamente estudiado en las dosúltimas décadas, debido a su gran aplicabilidad en sectores productivos tales como la industria del papel, vidrio, metal y madera.Existen dos tipos de cortes rectos en dos dimensiones, los cuales son mostrados en la figura 1.

(1) Licenciada en Computación. Profesora de la Facultad de Ingeniería de Sistemas e Informática, UNMSM. E-mail: merruri@hotmail.com
(2) Magíster en Informática. Profesor del Departamento de Ingeniería de Sistemas e Informática, UNMSM. E-mail: eruizl@unmsm.edu.pe



(a) Corte Recto de Guillotina



(b) Corte Recto No Guillotina

Figura 1. Modelos de cortes rectangulares

La dificultad inherente a este problema de corte en dos dimensiones, es el gran número de posibles patrones de corte que pueden ocurrir.

Se han reportado diferentes aplicaciones existentes para el problema de cortes en 2D: Vidrio Dyson & Gregory (1976), Farley (1983), Madsen (1979) y Grafos And/Or Joseph D. Touch (1985); Madera Morabito & García (1997), Venkateswarlu & Martyn (1992); Papel Harjunkoski et alt (1996), Westernlund et alt (1995); Tapetes Liton (1977); Lona Farley (1990).

Conceptos sobre Heurísticas

Dada la dificultad práctica para resolver de forma exacta toda una serie de problemas combinatorios, y para los cuales es necesario ofrecer alguna solución, comenzaron a aparecer algoritmos que proporcionan soluciones factibles (es decir, que satisfacen las restricciones del problema), las cuales, aunque no optimicen la función objetivo, se supone que al menos se acercan al valor óptimo en un tiempo de cálculo razonable. Este tipo de algoritmos se denominan heurísticos.

La heurística trata de métodos o algoritmos exploratorios durante la resolución de problemas en los cuales las soluciones se descubren por la evaluación del progreso logrado en la búsqueda de un resultado final. Son fáciles de programar y tienen una gran capacidad de cálculo.

Un algoritmo heurístico en un procedimiento de búsqueda de soluciones cercanas al óptimo a un costo computacional razonable, sin ser capaz de garantizar la optimalidad de las soluciones empleadas, ni determinan a que distancia de la solución óptima se encuentran [Reeves, 1995].

Existen varios factores que hacen interesante la utilización de algoritmos heurísticos para la resolución de un problema, tales como:

- inexistencia de un método exacto de resolución o éste requiere mucho tiempo de cálculo o memoria.
- no se necesita la solución óptima.
- los datos son pocos fiables.
- hay limitaciones de tiempo o espacio para el almacenamiento de datos.
- paso intermedio en la aplicación de otro algoritmo.

Sus ventajas son:

- Permiten una mayor flexibilidad para el manejo de las características del problema.
- No suele resultar complejo diseñar algoritmos heurísticas que en lugar de considerar funciones lineales utilicen funciones no lineales.
- Ofrecen más de una solución, lo cual permite ampliar las posibilidades de elección del que decide, sobre todo cuando existen factores no cuantificables que no han podido ser añadidos en el modelo.

A pesar de las ventajas de los métodos heurísticos no cabe duda que cuando una técnica exacta esté disponible debe ser preferida a cualquier tipo de heurística, sobre todo cuando se da el caso del manejo de grandes cantidades de dinero y, por lo tanto, pequeñas variaciones respecto al óptimo representan millones de pérdidas económicas.

Un inconveniente en el uso de métodos heurísticos, es no poder conocer la calidad de la solución; es decir, precisar cuan cerca se está de la solución optima.

Según Silver, Vidal, De Werra, 1980 [9] existen varios tipos de heurísticas:

- **Métodos constructivos.** Consiste en ir paulatinamente añadiendo componentes individuales a la solución hasta llegar a obtener una solución factible. El más conocido de este método es el algoritmo goloso o devorador (greedy).
  Un ejemplo sencillo del algoritmo greedy para el problema de TSP(Travell Salesman Problem), podría consistir en elegir una ciudad al azar, y luego desplazarse sucesivamente a la más cercana aún no visitada, hasta cerrar el círculo.

- **Métodos de descomposición (divide y vencerás).** Se trata de dividir el problema en subproblemas más pequeños, siendo que la salida de uno sea la entrada del siguiente, de forma que al resolver todos estos problemas obtengamos una solución para el problema global.
  Un ejemplo de partición en TSP consiste en dividir el plano en varias regiones pequeñas y resolver el TSP para cada una de las ciudades que se encuentran en cada una de las regiones, uniendo finalmente todas las soluciones de las regiones, para obtener la solución del problema global.

- **Métodos de reducción.** Tratan de identificar alguna características que presumiblemente debe poseer la solución óptima y de ese modo simplificar el problema.
  Ejemplo: para un problema de programación lineal entera, su relajación lineal consiste en ignorar la restricción de que las variables sean enteras.

- **Manipulación del modelo.** Modifican la estructura del modelo con el fin de hacerlo más sencillo al momento de resolver, deduciendo, a partir de su solución, la solución del problema original.
  Ejemplo: Reducir espacio de soluciones agrupando variables para disminuir el número de ellas, imponer nuevas restricciones, etc.

Algoritmo GRASP

Desarrollado originalmente por Feo y Resende [1], al estudiar un problema de cobertura de alta complejidad combinatoria. Cada iteración del GRASP consta generalmente de dos pasos: 1) La fase de construcción y 2) el procedimiento de búsqueda local.

- Se construye una solución tentativa, que luego es mejorada mediante un procedimiento de intercambio hasta que se llega a un óptimo local.
  En esta fase de construcción suele mantenerse una lista de candidatos formados por elementos de alta calidad. La solución inicial se construye iterativamente considerando un elemento cada vez.
- En cada iteración del procedimiento constructivo, un elemento es elegido de forma aleatoria de la lista de candidatos para añadirlo a la lista elegida como parte de la solución que se construye.
  La adición de un elemento a la lista de candidatos se determina mediante una función de tipo devorador, mientras la selección de un elemento de esa lista de candidatos depende de los que se hayan elegido previamente. Es decir, este método es a la vez goloso, aleatorio y adaptativo.

MÉTODOS EXISTENTES PARA RESOLVER EL PROBLEMA DE CORTES

El problema de cortes en dos dimensiones es un problema de optimización combinatoria. Se sabe que la única garantía de encontrar la solución óptima para problemas de optimización combinatoria sería enumerando todas las alternativas.

Para resolver grandes problemas de optimización combinatoria se puede elegir entre dos caminos. El primero consiste en buscar la optimalidad con el riesgo de obtener grandes tiempos de respuesta computacionales, en algunos casos posiblemente impracticable; el segundo, trata de obtener soluciones con rapidez, aun con el riesgo de caer en la suboptimalidad. Entre los que siguen la primera opción destacan los métodos de enumeración y las técnicas de programación dinámica. La segunda opción da lugar a los algoritmos de aproximación, también llamados con frecuencia algoritmos heurísticos.

Algoritmos exactos

Son algoritmos destinados a encontrar soluciones exactas para problemas NP-Complejos; construidos a partir de métodos capaces de determinar límites superiores o inferiores para la solución buscada en un esquema de enumeración. Para cada instancia, los límites inferiores y superiores (las soluciones factibles), van siendo calculados, hasta que cierta instancia tenga que ser dividida en subproblemas, de tal forma que la unión de éstos conduzca a soluciones factibles del problema inicial. Los subproblemas son procesados de forma similar.

Dentro de los algoritmos exactos se tienen: Método de Wang [10] y Método AAO(And/Or) Pearl, J. [8], entre otros.

Algoritmos aproximados

Son algoritmos que utilizan alguna heurística para encontrar las soluciones, también son denominados algoritmos sub-óptimos. Cuando se estudian estos tipos de algoritmos, la duda frecuente es, si existe alguna forma de determinar cuán distante se encuentra la solución encontrada de la solución óptima.

Para que estos algoritmos sean más empleados que los algoritmos exactos, deben de procurar tener tiempos de ejecución, más corto y ofrecer soluciones razonables. Entre algunos de estos tenemos, Algoritmo Goloso Miope [Edmonds, 1971], Algoritmo de Región de Confianza [Mauricio & Maculan, 1997].

Metaheurísticas

Las heurísticas para la solución de problemas de

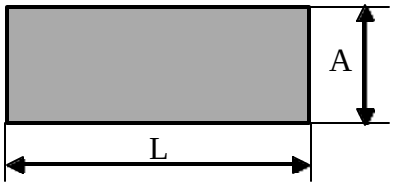


Figura 2 Lámina nueva

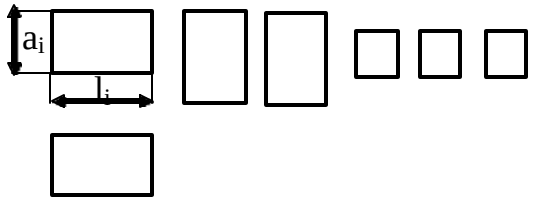


Figura 3. Requerimientos de iguales o diferentes tamaños

optimización discreta generalmente no consiguen encontrar una solución óptima global, pero si consiguen encontrar un óptimo local. A partir de unóptimo local, no se consigue generar ninguna mejora una vez alcanzada aquella solución. Una posibilidad para enfrentar esto es la perturbación del flujo de búsqueda de soluciones el cual admita, mediante pasos intermedios, soluciones con valores superiores del mínimo ya encontrado, aumentando así la oportunidad de no recorrer por caminos del espacio de soluciones que lleven a mínimos menores y también escapando de un mínimo local. Lo fundamental de este enfoque es la introducción de alguna aleatoriedad en el procedimiento.

En general todos los enfoques denominados metaheurísticos son de aplicación general. Son alternativas para la resolución de problemas más genéricos. Entre algunos de estos tenemos, Algoritmo Goloso Randómico Adaptativo (GRASP) (Feo&Resende [2]), Algoritmo Genético (Holland [5]), Simulated Annealing (Parada, V. [7]), Algoritmo de Búsqueda (Tabú Glover y Laguna [4]), Algoritmo Genético Evolutivo (Parada V., [6]).

DESCRIPCIÓN DEL PROBLEMA DE CORTES DE GUILLOTINA

Considere un número ilimitado de láminas rectangulares con dimensiones de largo L y ancho A y un conjunto de n requerimientos rectangulares de largo y ancho $(l_1,a_1),...(l_n,a_n)$ respectivamente con $l_i \leq L$ y $a_i \leq A$. El objetivo del problema consiste en realizar cortes de extremo a las láminas de forma de atender todos los requerimientos usando el menor número de láminas.

Este problema de corte algunas veces es subdividido en dos sub-problemas; el primero de ellos tiene por objetivo determinar un subconjunto de láminas (barras) de los disponibles para satisfacer la demanda de piezas (the assortment problem), y el segundo problema tiene por objetivo determinar el patrón de corte adecuado para cumplir con la demanda a partir del stock de láminas (barras) minimizando las pérdidas o desperdicio (the trim-loss problem). Se conoce como constrained stock cutting problem cuando el número de piezas de un determinado tipo solicitado es limitado.

Algunos aspectos teóricos del problema de cutting stock son referenciados por Golden (1976) y Hinxman (1980)

Láminas: Pieza rectangular nueva de vidrio, papel, metal o madera con dimensiones de largo L y ancho A sobre la cual se efectuan los cortes, según requerimientos. Cada vez que se necesite una nueva lámina esta será del mismo tamaño (L x A), mostrada en la figura 2.

Requerimientos u órdenes: Pieza rectangular con un determinado largo (li) y ancho (ai) solicitado por los usuarios según necesidad y que serán cortados sobre una lámina nueva o residuo de una determinada lámina, como muestra la figura 3.

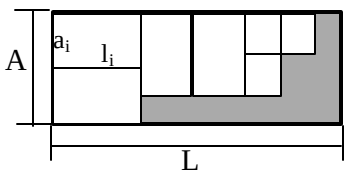


Figura 4. Atención de requerimientos

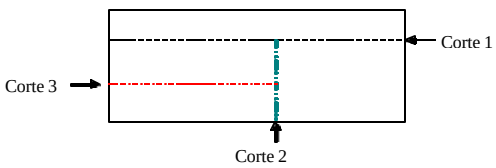


Figura 5. Modelo de corte por guillotina

Las dimensiones de la lámina, como el tamaño de cada uno de los requerimientos son datos de entrada por lo tanto serán ingresados por el usuario al inicio.

Patrones: Esquemas o modelos que muestran la manera más adecuada para cumplir con la atención de todos los requerimientos, sobre una o varias láminas como muestra la figura 4. Este patrón muestra la atención de los siete requerimientos indicados anteriormente, en una sola lámina.

Objetivos del problema

- Determinar el patrón de corte adecuado para cumplir con los requerimientos a partir de las existencias de láminas minimizando las pérdidas o desperdicios.
- Determinar un subconjunto de láminas de los disponibles para satisfacer los requerimientos de piezas (órdenes).
- Incrementar la productividad y reducir costos.

Corte recto por guillotina

Existen problemas particulares de cortes de dos dimensiones resultantes de la utilización de ciertas máquinas que cortan el material en línea recta de un extremo de la lámina al otro extremo opuesto, como por ejemplo la guillotina, usada para cortar papel y debido fundamentalmente a que la forma de los objetos (planchas, láminas) y de los requerimientos son rectangulares o cuadrangulares; los patrones de corte en este caso son denominados ortogonales. La complejidad de estos problemas depende del número de cambios de dirección de los cortes (períodos) y del número de cortes paralelos que se efectúa por período.

Un periodo, es el cambio de dirección de un determinado corte realizado por la guillotina, el cambio puede ser de corte horizontal a corte vertical o de corte vertical a corte horizontal.

Un ejemplo de corte por periodo se muestra en la figura 5, donde los cortes son enumerados en el orden en que deben ser efectuados.

El corte por guillotina limita mucho el modelo de corte de dos dimensiones. Este tipo de corte por guillotina puede ser resuelto usando métodos de programación dinámica. Una fórmula de recurrencia usada para este fin es dado por Gilmore y Gomory [3].

Existen subclases especiales de los cortes por guillotina menos difíciles de modelar y que corresponden a los cortes usados en ciertas industrias.

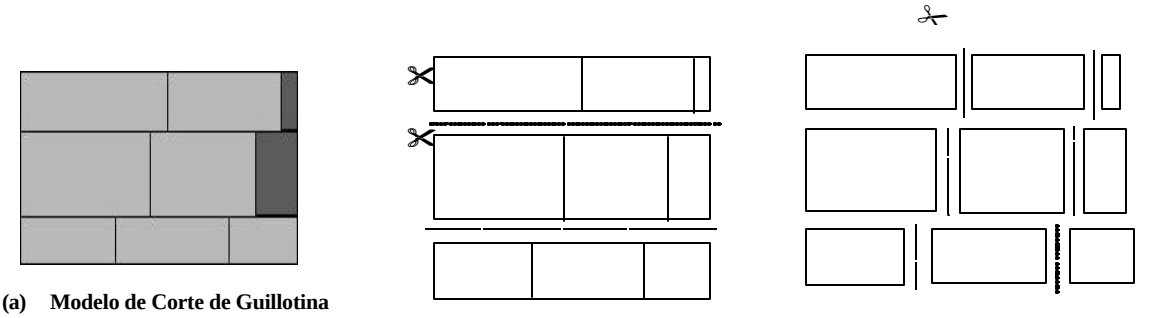
Una subclase muy importante de modelos de corte 2D por guillotina es aquella en que los cortes son considerados en dos periodos, como se muestra en la figura 6, donde el rectángulo A x L es cortado según el ancho en tiras y luego cada tira cortado según el largo, como se detalla en la figura 6: (a) Modelo de corte de guillotina en dos periodos; después de haberse realizado los cortes horizontales llegamos a (b) Corte del primer periodo (corte horizontal), y finalmente cambia la dirección del corte (periodo) de horizontal a vertical como se muestra en (c) Corte del segundo periodo (corte vertical).

En ocasiones es necesario contar con tres o más periodos, como se muestra en la figura 7.

De una forma inductiva es posible definir cortes por guillotina en n periodos como un estilo de corte en el que se considera cada tira como una nueva lámina en la que es posible efectuar un corte por guillotina en n-1 periodos.

La dificultad se presenta en los siguientes aspectos:

- El número de patrones es excesivamente grande.
- Pertenece a la geometría combinatoria.

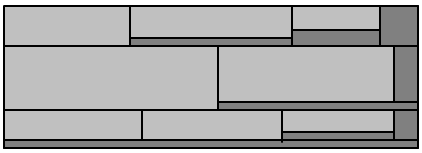


(a) Modelo de Corte de Guillotina en Dos Periodos

(b) Corte del primer periodo (Corte Horizontal)

(c) Corte del segundo periodo (Corte Vertical)

Figura 6. Procedimiento de cortes por periodos



Modelo de Corte de Guillotina en Tres Periodos

Modelo de Corte de Guillotina en Cinco Periodos

Figura 7. Modelos de cortes por periodos

1. Leer (n, $l_1a_1, l_2a_2, ..., l_na_n$, L, A)
2. Ordenar(l_i, a_i) tal que: $l_1a_1 \geq l_2a_2 \geq ... \geq l_na_n$
3. $H_1 := V_1 := \hat{O}$; $F_1 := \{((0,0), (L,A))\}$; m := 0
4. Para k := 1, ..., n
- 4.1. $B_k := (l_k a_k)$
- 4.2. $\beta \max = m+1$
- 4.3. $\beta_{\min} = \min_{1 \leq r \leq m+1} \{r : B_k \subseteq H_r \cup V_r \cup F_r\}$
- 4.4. $RCL = \{r : \beta_{\min} \leq r \leq \beta_{\min} + \alpha(\beta_{\max} - \beta_{\min}), B_r \subseteq H_r \cup V_r \cup F_r\}$
- 4.5. $i = random(RCL)$
- 4.6. Si i = m + 1 Entonces
- 4.7. Corte_L ámina_Nueva(B_k, i), m:=m+1, $F_{m+1} := \{((0,0), (L,A))\}$
- 4.8. Sino
- 4.9. Corte_Lámina_Usada(B_k, i)
- 4.10. Fin_Si
- 4.11. Fin_Para

Figura 8. Algoritmo Construcción GRASP (FFD), con un parámetro de relajación

1. Hallar el máximo entero

Si Max $\left\{\left[\frac{L}{l_k}\right]\left[\frac{A}{a_k}\right], \left[\frac{L}{a_k}\right]\left[\frac{A}{l_k}\right]\right\} = \left[\frac{L}{a_k}\right]\left[\frac{A}{l_k}\right]$ entonces (l_ka_k) := (a_kl_k) /* Rotamos la pieza en 90º para tener la mejor ubicación*/

2. $B_k := (l_ka_k)$
3. Si $a_k < A$ Entonces $H_i := \{(L, A-a_k)\}$ /*Corte horizontal*/
4. Si $l_k < L$ Entonces $V_i := (L-l_k, a_k)$ /*Corte vertical*/
5. $F_i = \hat{O}$

Figura 9. Procedimiento Corte_Lámina_Nueva(Bk, i)

1. Buscar_Primera_Pieza $P := (\bar{l}, \bar{a})$ /. $B_k \subseteq H_i \cup V_i \cup F_i$
2. Hallar máximo entero

   Si $Max\left\{\left[\frac{\bar{l}}{l_k}\right]\left[\frac{\bar{a}}{a_k}\right], \left[\frac{\bar{l}}{a_k}\right]\left[\frac{\bar{a}}{l_k}\right]\right\} = \left[\frac{\bar{l}}{a_k}\right]\left[\frac{\bar{a}}{l_k}\right]$ entonces $(l_k a_k) := (a_k, l_k)$

   /*Rotación sobre el requerimiento B_k en la pieza P.*/
3. $B_k := (l_k a_k)$
4. Si $a_k < \bar{a}$ Entonces $H_i := \{(\bar{l}, \bar{a} - a_k)\}$
5. Si $l_k < \bar{l}$ Entonces $V_i := \{(\bar{l} - l_k, a_k)\}$
6. Si $l_k a_k = \bar{l}\,\bar{a}$ Entonces La primera pieza que contiene a B_k = Ô

Figura 10. Procedimiento Corte_Lámina_Usada (Bk,i)

- Cuando los items no tienen forma rectangular el problema aún es más difícil.
- Es un problema NP-difícil.

PRESENTACIÓN DEL ALGORITMO PROPUESTO

Algoritmo Construcción GRASP (FFD)
Se muesta en la figura 8 y considera un parámetro de relajación. El tamaño de la lámina (L, A) y los requerimientos $l_i a_i$ son datos de entrada. Estos son ordenados de menor a mayor. Los conjuntos H_r (corte horizontal) y V_r (corte vertical) para $r = 1$ son un conjunto vació. F_r es el conjunto compuesto por L y A; m es el número de láminas a utilizar, el cual se inicializa en 0.

Siendo n la cantidad de requerimientos, se itera desde k = 1 hasta n, en β_{min} se determina el mínimo residuo para el requerimiento B_k que pertenezca al conjunto H_r, V_r y F_r. En RCL se encuentran los residuos seleccionados utilizando el parámetro de relajación α dentro del cual se elegirá aleatoriamente el residuo o la lámina completa donde se realizará el corte horizontal o vertical

Es recomendable que á å <0,1>. Cuando α =0, el RCL estará conformado sólo por el mejor candidato y la fase de construcción se comportará como un algoritmo goloso. Cuando α =1, el RCL estará conformado por todos los elementos candidatos de F y la selección de uno de estos será totalmente aleatorio.

Cuadro 1. Resultados numéricos con el parámetro de relajación $\alpha = 0{,}25;0{,}5;0{,}75$

a	**Requerimientos (l_i,a_i)**					**Desperdicio cm**
	B_1	**B_2**	**B_3**	**B_4**	**B_5**	
0,25	l_1 = 20 a_1 = 6 d = 5	l_2 = 10 a_2 = 12 d = 2	l_3 = 17 a_3 = 9 d = 10	l_4 = 5 a_4 = 13 d = 3	l_5 = 19 a_5 = 7 d = 7	**2685**
0,50	l_1 = 20 a_1 = 6 d = 5	l_2 = 10 a_2 = 12 d = 2	l_3 = 17 a_3 = 9 d = 10	l_4 = 5 a_4 = 13 d = 3	l_5 = 19 a_5 = 7 d = 7	**2685**
0,75	l_1 = 20 a_1 = 6 d = 5	l_2 = 10 a_2 = 12 d = 2	l_3 = 17 a_3 = 9 d = 10	l_4 = 5 a_4 = 13 d = 3	l_5 = 19 a_5 = 7 d = 7	**3094**

Procedimiento Corte_Lámina_Nueva(Bk, i) Aquí se halla el máximo entero del requerimiento (pieza) el cual es rotado a 90º a fin de tener una mejor ubicación para realizar un corte horizontal o vertical (ver figura 9).

Procedimiento Corte_Lámina_Usada (Bk,i) En este procedimiento, se busca la primera pieza (P) que pertenezca al conjunto $H_i \cup V_i \cup F_i$, luego busca el máximo entero a fin de tener una rotación del requerimiento Bk en la pieza (residuo), tal como se muestra en la figura 10.

RESULTADOS NUMÉRICOS

Para probar el algoritmo se consideraron tres valores distintos para α = 0,25; 0,50 y 0,75. Se tomaron como muestra cinco requerimientos (B_1, B_2, B_3, B_4 y B_5), cada uno con su respectiva cantidad de demanda (d) por requerimiento. La lámina donde se realizarán los cortes es de la siguiente medida:

Largo (L) = 90
Altura (A) = 50

Como se observa en el cuadro 1, para un α entre 0,25 y 0,50; el desperdicio es igual a 2685 cm, que es menor que el de un $\alpha = 0{,}75$.

Es decir, cuando el parámetro de relajación es menor igual a 0,5 el patrón de cortes se acerca a una solución optima por generar menor desperdicio.

CONCLUSIONES Y RECOMENDACIONES

Los algoritmos GRASP son recomendables cuando el conjunto de datos a trabajar es grande.

Si bien es cierto el algoritmo expuesto no obtiene la solución optima, sin embargo llega a una solución aproximada con un ahorro en tiempo de ejecución en el proceso y disminución de costo al tener un menor desperdicio.

Asimismo, el algoritmo expuesto puede aplicarse a sectores diversos de la industria que tengan que ver con el problema del corte de guillotina.

REFERENCIAS BIBLIOGRÁFICAS

1. Feo, T., Resende (1989). A probabilistic heuristic for a computationally difficult set convering problem. Operations Research Letters, 8:67-71, (1).

2. Feo, T., Resende, M. (1995). Greedy Randomized Adaptative Search Procedure. Journal of Global Optimization Vol. 6, pp. 109-133. (4)

3. Gilmore P., and Gomory R. (1965). Multistage cutting problems of two and more dimensions. Operation Research 13 94-119. (9)

4. Glover, F& Laguna, M. (1997). Tabu Search, Kluwer Academic Publishers, USA. (7)

5. Holland, J. (1975), Adaptation in Natural and Artificial Systems, University of Michigan Press, Michigan, USA (5)

6. Parada, V., Muñoz, R. y Gómes de Alvarenga, A. (1995). A Hybrid Genetic Algorithm for the Two-Dimensional Guillotine Cutting Problem, in Evolutionary Algorithms in Management aplications, Eds. Jorg Biethahn & Volker Nissen, Springer, Berlín. (8)

7. Parada, V., Sepúlveda, M.; Solar, M. &Gomes, A. (1997). Solution for the Constrained Guillotine Cutting Problem by Simulated Annealing, Computers & Operations Research. (6)

8. Pearl, J. (1984). Heuristics: Intelligent Search Strategies for Computer problem Solving. Addison-Wesley Publishing Company, Reading M.A. (3)

9. Silver, R.V.V. Vidal and D. de Werra (1980). A tutorial on heuristic methods, European Journal of Operational Research 5, 153.

10. Wang, P.Y. (1983). Two Algorithms for Constraint Two Dimensional Cutting Stock Problems, Operations Research, 31, 573-586. (2)