

Procedimientos constructivos adaptativos (GRASP) para el problema del empaquetado bidimensional

Jesús David Beltrán Cano, Jose Eduardo Calderón,
Rayco Jorge Cabrera, J. Marcos Moreno Vega
Departamento de Estadística, Investigación Operativa y Computación
Centro Superior de Informática
Universidad de La Laguna

Resumen: *Por empaquetado de rectángulos se entiende una clase de problemas de corte y empaquetado con variadas aplicaciones en la Industria. Uno de los problemas más conocidos de esta clase es el problema del empaquetado rectangular bidimensional no guillotina. En éste se pretende empaquetar, sin solapamientos, un conjunto dado de rectángulos en un objeto rectangular de anchura conocida y altura infinita. El propósito es obtener la distribución de menor altura. Dada su dificultad y su gran aplicabilidad, existe un creciente interés en disponer de procedimientos eficientes y eficaces para resolver este problema. En este trabajo, diseñamos e implementamos diferentes métodos constructivos para el problema. Además, comparamos éstos frente a la mejor propuesta conocida para resolver el problema. De los resultados obtenidos, se concluye la bondad de nuestras propuestas: obtenemos soluciones cercanas a la óptima (en algunos casos, con mayor calidad que la mejor solución conocida) en un tiempo inferior al requerido por el mejor procedimiento de solución conocido.*

Keywords: Empaquetado de rectángulos, Heurística, GRASP

1. Introducción

En un problema de empaquetado de rectángulos se dispone de un conjunto de *piezas* rectangulares con longitudes conocidas y se desean distribuir éstas en un *objeto* rectangular de forma que optimicen alguna función del área que ocupan. Una forma alternativa de interpretar el problema supone que se dispone de un objeto rectangular desde el que hay que obtener el conjunto de piezas realizando cortes perpendiculares a los ejes.

Las restricciones pueden imponer que la distribución siga un patrón preestablecido, que el número de veces que puede usarse cada pieza esté acotado o que las piezas tengan

asociado un beneficio. Por patrón entendemos el tipo de corte que está permitido. Un corte es tipo guillotina si atraviesa al objeto desde un lado del mismo hasta el lado opuesto. En un corte no guillotina, lo anterior no es cierto.

Pueden considerarse las siguientes tres situaciones referidas al conocimiento que se tiene del objeto en que hay que distribuir las piezas.

1. *El alto y el ancho del objeto son conocidos (figura 1(a))*: se pretende, por tanto, distribuir en el objeto aquellas piezas que optimicen la función objetivo considerada.
2. *Sólo es conocido el ancho (figura 1(b))*: se desea encontrar la distribución de piezas que minimiza la altura del objeto. Restricciones tecnológicas pueden imponer que el corte sea tipo guillotina, aunque, en general, esta restricción no suele considerarse. Ejemplos de estos problemas se encuentran en [1] y [4].
3. *No son conocidos ni el alto, ni el ancho (figura 1(c))*: el propósito es distribuir las piezas de tal forma que el rectángulo que esta distribución determina sea el de menor área. En [5] se emplea este problema para particionar una malla dada de procesadores en submallas a las que asignar tareas que pueden ejecutarse independientemente.

En lo que sigue se trata el problema del empaquetado rectangular bidimensional no guillotina que se formula como sigue. Dado un objeto rectangular de amplitud fija w y altura infinita, y un conjunto, $\mathcal{R} = \{R(w_1, h_1), \dots, R(w_n, h_n)\}$, de rectángulos con al menos uno de sus lados, w_i, h_i , menor que w , se desea empaquetar el conjunto \mathcal{R} en el objeto rectangular utilizando el menor espacio posible. En este problema está permitido rotar los objetos y los cortes pueden ser de tipo no guillotina (ver figura 1(b)).

En general, se suelen representar las soluciones de este problema a través de una permutación que indica el orden en que los rectángulos son considerados para su inclusión en el objeto, junto a un procedimiento que indica la posición que ocupa el rectángulo en el objeto. Entre los procedimientos comunmente empleados para obtener la posición se encuentra el conocido como Botton-Left. Consta de los siguientes pasos. Colocar el primer rectángulo en la esquina inferior izquierda del objeto. Colocar el siguiente rectángulo en la esquina superior derecha. Bajar el rectángulo tanto como sea posible. A continuación, moverlo hacia la izquierda tanto como sea posible y, en su caso, volver a bajarlo tanto como sea posible. Repetir lo anterior mientras queden rectángulos por colocar. En la figura 2(b) se muestra un ejemplo en el que los rectángulos 1, 2, 3, 4, 5 y 6 se han empaquetado, siguiendo este orden, con la estrategia Botton-Left.

Nótese que al aplicar la estrategia Botton-Left, pueden formarse áreas no aprovechables (desperdicios) (ésto ocurre en el ejemplo de la figura 2(b) al incluir el rectángulo 4). Botton-Left-Fill (*BLF*) es otra estrategia de colocación que, antes de colocar un rectángulo según la estrategia Botton-Left, comprueba si es posible colocar éste en alguno de los desperdicios que se han generado hasta el momento.

Con cualquiera de las representaciones anteriores se pueden diseñar varias de las metaheurísticas de búsqueda más conocidas: Simulated Annealing, Algoritmos Genéticos, ... En [4] se proponen varias de éstas metaheurísticas y se analiza el comportamiento de las

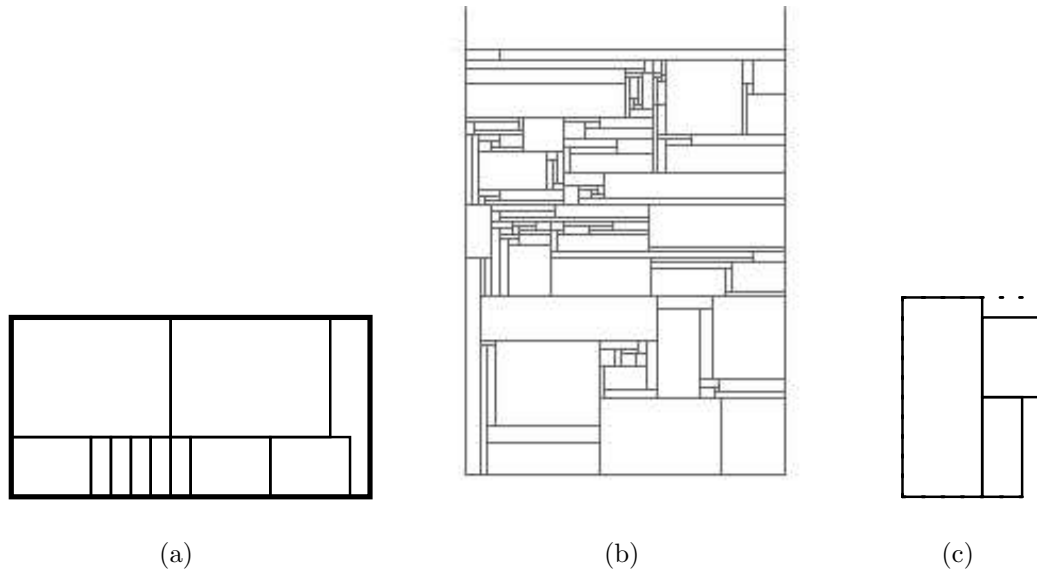


Figura 1. Problemas de empaquetado de rectángulos

mismas. Además, se enumeran los trabajos más importantes relativos al problema del empaquetado rectangular bidimensional no guillotina.

2. Métodos constructivos

En un método constructivo se añade iterativamente elementos a una estructura, inicialmente vacía, hasta obtener una solución del problema. La elección del elemento a incluir se basa en una evaluación *heurística*, que mide la conveniencia de considerar este elemento como parte de la solución. La función heurística es dependiente del problema y expresa el conocimiento que sobre el mismo se tiene. Si la evaluación de un elemento depende de los elementos previamente incluidos en la solución se dice que el método es *adaptativo*.

Además de la función heurística, es necesaria una estrategia que indique qué elemento se escoge. Una de las estrategias más conocidas es la *greedy* en la que se selecciona el elemento que optimiza la función heurística. Esta estrategia suele dar pobres resultados en la mayoría de los casos. Por ello se han propuesto estrategias alternativas. Una de ellas consiste en elegir, no el mejor elemento, sino uno de los mejores al azar. Al conjunto de los mejores elementos se le llama *Lista Restringida de Candidatos (LRC)*.

GRASP (Greedy Randomized Adaptive Search Procedure) [3] es un procedimiento heurístico que consta de varias etapas. A una fase constructiva, en la que se escoge al azar un elemento de la lista restringida de candidatos, le sigue una fase de postprocesamiento en la que se mejora la solución obtenida en la fase anterior. Como postprocesamiento suele emplearse una simple búsqueda local descendente. Los anteriores pasos se reiteran hasta que se cumpla el criterio de parada. La mejor solución obtenida es la propuesta por

el algoritmo.

Los elementos que determinan completamente la técnica GRASP son: la función heurística, la forma en que se construye la lista restringida de candidatos, el método de postprocesamiento y el criterio de parada.

En nuestros procedimientos, se finaliza la búsqueda después de un número dado, n_{iter} , de pasadas del bucle anterior. En la fase de postprocesamiento, si existe, se determina la mejor colocación de los últimos k (parámetro) rectángulos incluidos en la solución. Para ello se emplea el procedimiento descrito en 2.3. Para definir las listas restringidas de candidatos, y, por tanto, las correspondientes funciones heurísticas, es preciso introducir el concepto de contorno.

2.1. Contorno

La inclusión de un rectángulo cualquiera en el objeto, determina un contorno superior rectangular como el que se muestra en la figura 2(a). Además, es posible que se obtengan áreas no aprovechables, llamadas desperdicios, como el que se obtiene al incluir el rectángulo 4 en el objeto de la figura 2(a). El contorno, C , puede representarse por medio del conjunto de segmentos horizontales (tomados de izquierda a derecha) que lo forman. Es decir:

$$C = \{(y^1, x_1^1, x_2^1), (y^2, x_1^2, x_2^2), \dots, (y^c, x_1^c, x_2^c)\}$$

con

$$\begin{aligned} y^i &\equiv \text{altura del } i\text{-ésimo segmento} \\ x_1^i &\equiv \text{punto inicial del } i\text{-ésimo segmento} \quad . \quad \text{Además, } x_1^1 = 0, x_2^c = w. \\ x_2^i &\equiv \text{punto final del } i\text{-ésimo segmento} \end{aligned}$$

Nótese que, intuitivamente, es preferible un contorno formado por pocos niveles a otro con muchos niveles. Esto es así, ya que, en general, la posibilidad de obtener desperdicios aumenta con el número de niveles.

2.2. Lista restringida de candidatos

Sea t la iteración actual del proceso constructivo y supongamos que $\mathcal{R} = \mathcal{R}_1 \cup \mathcal{R}_2$, siendo \mathcal{R}_1 el conjunto de los rectángulos previamente incluidos en el objeto y $\mathcal{R}_2 = \mathcal{R} \setminus \mathcal{R}_1$. Sea $C(t)$ el contorno determinado por los rectángulos de \mathcal{R}_1 . Evaluaremos la conveniencia de incluir un rectángulo de \mathcal{R}_2 en el objeto por la forma que tendrá el contorno $C(t)$ tras su inclusión. Las diferentes evaluaciones que proponemos pretenden aprovechar mejor el espacio disponible y suavizar el contorno.

1. *Lista restringida de candidatos 1*: sea dado $\alpha_1 \in [0, 1]$ y supongamos que (y^i, x_1^i, x_2^i) es el segmento del contorno con menor altura. La lista restringida de candidatos se construye como sigue:

$$LRC^1 = \{R(w_j, h_j) \in \mathcal{R}_2 : (0 \leq x_2^i - x_1^i - w_j \leq \alpha_1) \vee (0 \leq x_2^i - x_1^i - h_j \leq \alpha_1)\}.$$

Es decir, la lista está formada por aquellos rectángulos que mejor se ajustan al ancho del segmento inferior del contorno. El ajuste viene determinado por el valor de α_1 .

2. *Lista restringida de candidatos 2:* sea dado $\alpha_2 \in [0, 1]$, (y^i, x_1^i, x_2^i) el segmento del contorno con menor altura y supongamos que los segmentos anterior y posterior a éste, respectivamente $(y^{i-1}, x_1^{i-1}, x_2^{i-1})$ y $(y^{i+1}, x_1^{i+1}, x_2^{i+1})$, son tales que $y^i < y^{i+1} < y^{i-1}$ (ver figura 2(a)). La lista restringida de candidatos se construye como sigue:

$$LRC^2 = \{R(w_j, h_j) \in LRC^1 : (0 \leq y^{i+1} - y^i - w_j \leq \alpha_2) \vee (0 \leq y^{i+1} - y^i - h_j \leq \alpha_2)\}.$$

Esto es, la lista está constituida por los rectángulos que mejor se ajustan al hueco formado los puntos (x_1^i, y^i) , (x_2^i, y^i) , (x_1^{i+1}, y^{i+1}) y (x_1^i, y^{i+1}) . El ajuste viene dado por los valores de α_1 y α_2 .

Si $LRC^1 \cap LRC^2 = \emptyset$, hacer $LRC^2 = LRC^1$.

3. *Lista restringida de candidatos 3:* en las condiciones anteriores, si $LRC^2 = \emptyset$, se construye la lista restringida de candidatos como sigue:

$$LRC^3 = \{R(w_j, h_j) \in LRC^1 : (0 \leq y^{i-1} - y^i - w_j \leq \alpha_3) \vee (0 \leq y^{i-1} - y^i - h_j \leq \alpha_3)\}$$

Ahora, la lista está formada por los rectángulos que mejor se ajustan al hueco que determinan los puntos (x_1^i, y^i) , (x_2^i, y^i) , (x_2^i, y^{i-1}) y (x_1^i, y^{i-1}) . El ajuste viene dado por los valores de α_1 y α_3 .

Si $LRC^1 \cap LRC^3 = \emptyset$, hacer $LRC^3 = LRC^1$.

Para que las definiciones anteriores tengan sentido, debe haber, al menos, un rectángulo de \mathcal{R}_2 , digamos $R(w_r, h_r)$, tal que $(0 \leq x_2^i - x_1^i - w_r \leq \alpha_1) \vee (0 \leq x_2^i - x_1^i - h_r \leq \alpha_1)$. Si ningún elemento de \mathcal{R}_2 cumple la anterior condición, se ubica dentro del objeto el rectángulo que mejor se ajusta a $x_2^i - x_1^i$, y se reconstruye el contorno. Si no existe tal rectángulo, se reconstruye el contorno eliminando, convenientemente, el segmento (y^i, x_1^i, x_2^i) .

2.3. Postprocesamiento

Una de las situaciones anómalas que puede presentarse al aplicar los métodos constructivos anteriores se muestra en la figura 2. Consideremos la ubicación del rectángulo 6. Cualquiera de los métodos anteriores lo ubicaría según se indica en la figura 2(b). La bondad de esta nueva situación depende del instante en que se produce. En las primeras iteraciones del método, la situación es aconsejable. No obstante, en las últimas iteraciones puede producir soluciones de baja calidad. En particular, si nos encontramos en la última iteración, sería preferible ubicarlo como se muestra en la figura 2(c). Por ello, se propone el siguiente procedimiento de mejora, que se aplica a la solución obtenida en la fase constructiva.

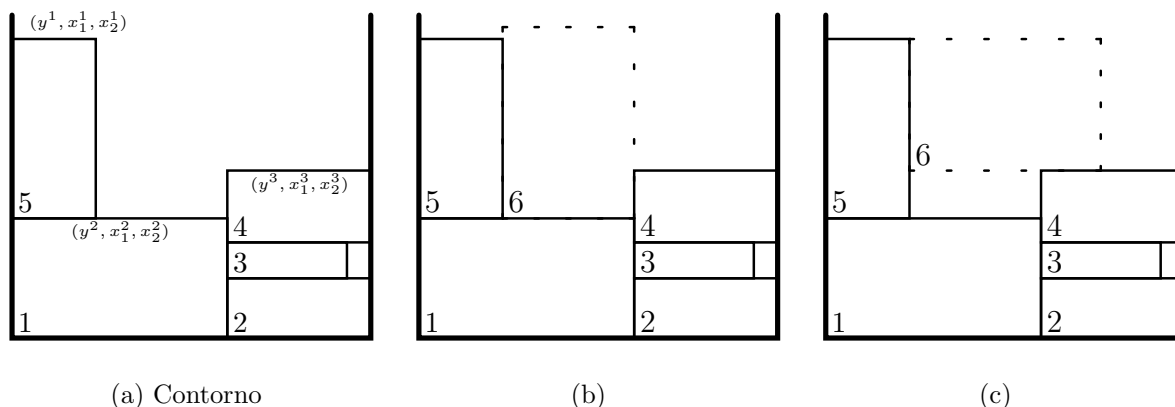


Figura 2. Procedimiento de mejora

Procedimiento de mejora: extraer los últimos k (parámetro) rectángulos de la solución. Supongamos, por simplicidad, que son $\{R_1, R_2, \dots, R_k\}$. Para cada permutación, $(R_{i_1}, R_{i_2}, \dots, R_{i_k})$, de los rectángulos: 1.) Hacer $j = 1$. Colocar el rectángulo R_{i_j} en la posición más profunda del objeto y con la orientación que suponga una menor altura relativa. Actualizar el contorno. 2.) Hacer $j = j + 1$. Tomar el rectángulo R_{i_j} de la permutación y empaquetarlo siguiendo el proceso anterior. 3.) Si $j = k$, ir al paso 4; en caso contrario repetir el paso 2. 4.) Devolver la mejor de las soluciones obtenidas con el método anterior.

2.4. Métodos

Combinando las listas restringidas de candidatos LRC^1 , LRC^2 y LRC^3 y la anterior técnica de postprocesamiento se obtienen varios GRASPs. En $GRASP_i$, $i = 1, 2, 3$, en cada iteración, se escoge al azar un elemento de LRC^i , y se reconstruye el contorno. El proceso anterior se reitera mientras queden rectángulos por incluir en el objeto.

En $GRASP_i$, $i = 4, 5$, tras una fase constructiva empleando, respectivamente, los métodos $GRASP_i$, $i = 2, 3$, se aplica la técnica de postprocesamiento descrita en 2.3.

3. Experiencia computacional

La experiencia computacional se desarrolló en dos etapas. Estas coinciden con los objetivos principales de cualquier experimentación con técnicas heurísticas: ajustar convenientemente los parámetros que definen la técnica, y comparar el comportamiento de la heurística propuesta frente a otros procedimientos de resolución conocidos para el problema. Se emplearon diferentes tipos de problemas para cada etapa: problemas generados aleatoriamente para la primera y problemas tests para la segunda.

El lenguaje empleado para programar los procedimientos fué C (compilador Turbo C), la máquina sobre la que se ejecutaron los programas un $K6II$ a $450Mhz$ con $64Mb$ de

n	w	h_{opt}	$GRASP_1$	$GRASP_2$	$GRASP_3$	n	w	h_{opt}	$GRASP_1$	$GRASP_2$	$GRASP_3$		
50	30	45	0,008	0,000	0,000	200	60	100	0,074	0,004	0,001		
			T_0	$T_{0,0,1}$	$T_{0,0}$				T_0	$T_{0,2,0,2}$	$T_{0,1,0,2}$		
			($T_{0,0,2}$)					($T_{0,2,0,2}$)					
		60	0,022	0,005	0,244			130	0,007	0,000	0,000	0,000	
			T_0	$T_{0,1,0,2}$	=				T_0	$T_{0,0,2}$	$T_{0,0}$	$T_{0,0}$	
			($T_{0,0,1}$)	($T_{0,0,1}$)					($T_{0,0}$)	($T_{0,0}$)	($T_{0,0,2}$)	($T_{0,0,2}$)	
		60	90	0,022	0,026		0,001		100	90	0,007	0,000	0,000
			T_0	$T_{0,1,0,2}$	$T_{0,1,0}$					T_0	$T_{0,0}$	$T_{0,1,0,2}$	$T_{0,1,0,2}$
			($T_{0,0}$)	($T_{0,0,2}$)	($T_{0,0,2}$)					0,022	0,000	0,000	($T_{0,0,2}$)
			100	0,819	0,000		0,000		140	0,022	0,000	0,000	0,000
			=	$T_{0,0,2}$	$T_{0,1,0}$			T_0	$T_{0,0}$	$T_{0,0,1}$	($T_{0,0,2}$)		
			($T_{0,0,1}$)	($T_{0,0,2}$)						($T_{0,0,2}$)			
	90	120	0,007	0,002	0,010		200	150	0,015	0,000	0,000		
		T_0	$T_{0,1,0}$	$T_{0,0,2}$					T_0	$T_{0,0}$	$T_{0,1,0}$		
		($T_{0,0}$)								($T_{0,0,2}$)			
		150	0,022	0,001	0,156		200	0,022	0,000	0,000	0,000		
		T_0	$T_{0,0,1}$	=				T_0	$T_{0,0}$	$T_{0,0}$	$T_{0,0}$		
		($T_{0,0,2}$)	($T_{0,0,2}$)							($T_{0,0,2}$)			
100	50	70	0,007	0,001	0,000	300	80	150	0,022	0,000	0,000		
		T_0	$T_{0,0}$	$T_{0,0}$							T_0	$T_{0,0}$	$T_{0,0}$
		($T_{0,0,2}$)	($T_{0,0,2}$)						0,007	0,000	0,000		
		110	0,007	0,000	0,000			200	0,007	0,000	0,000		
		T_0	$T_{0,0,2}$	$T_{0,0,2}$					T_0	$T_{0,0}$	$T_{0,0}$		
		($T_{0,0}$)	($T_{0,0}$)						0,022	0,000	0,000		
	80	90	0,022	0,003	0,032			140	120	0,022	0,000	0,000	
		T_0	$T_{0,0,1}$	$T_{0,0}$						T_0	$T_{0,0}$	$T_{0,0}$	
		($T_{0,0,2}$)	($T_{0,0,2}$)	($T_{0,0,2}$)					0,022	0,000	0,000		
		140	0,007	0,001	0,009			200	200	0,007	0,000	0,000	
		T_0	$T_{0,0,2}$	$T_{0,0,1}$					T_0	$T_{0,0}$	$T_{0,0}$		
		($T_{0,0,2}$)	($T_{0,0,2}$)	($T_{0,0,2}$)						$T_{0,0}$	$T_{0,0}$		
		100	180	0,022	0,003	0,000		280	0,022	0,000	0,000		
		T_0	$T_{0,0}$	$T_{0,0}$					T_0	$T_{0,0}$	$T_{0,0}$		
		($T_{0,0,2}$)	($T_{0,0,2}$)										
		200	0,247	0,000	0,000								
		=	$T_{0,0}$	$T_{0,1,0}$									
		($T_{0,0,2}$)	($T_{0,0,2}$)										

Cuadro I. Nivel de significancia para la hipótesis nula de igualdad de tratamientos, y tratamiento con mejor comportamiento (entre paréntesis el segundo mejor tratamiento).

memoria RAM y el paquete estadístico usado para el análisis de los resultados el *SPSS* para Windows Versión 8.0.1S.

3.1. Ajuste de parámetros

Para obtener problemas aleatorios se implementó un generador que, dado el ancho del objeto rectangular, w , el número de rectángulos, n , y el valor objetivo óptimo, h_{opt} , suministra un conjunto de n rectángulos que pueden ubicarse en un objeto rectangular de amplitud w utilizando una altura h_{opt} . En la figura 1(b) se muestra uno de los problemas obtenidos con este generador.

Dada la naturaleza estocástica de los métodos propuestos, ejecuciones repetidas sobre el mismo problema y con la misma elección de los parámetros pueden suministrar diferentes resultados. Por ello, una vez fijado los valores de los parámetros, cada problema fue resuelto 5 veces. Se tomó $n_{iter} = 40$. Se generaron problemas aleatorios de diferentes tamaños (ver tabla I) para cubrir una amplia variedad de situaciones. Los mismos problemas fueron usados para cada uno de los GRASPs. Como variable respuesta se tomó el valor objetivo medio de las n_{iter} pasadas del bucle del GRASP.

Los parámetros interés de estudio fueron α_1 , (α_1, α_2) y $(\alpha_1, \alpha_2, \alpha_3)$ para los métodos $GRASP_1$, $GRASP_2$ y $GRASP_3$, respectivamente. Para cada uno de los parámetros se seleccionaron tres niveles: $\alpha_1 = 0, 0,1, 0,2$, $\alpha_2 = 0, 0,1, 0,2$ y $\alpha_3 = 0, 0,1, 0,2$ ($\alpha_2 = \alpha_3$), y se consideraron todas las combinaciones posibles como tratamientos a estudiar. Se

obtienen, por tanto, 3 tratamientos para $GRASP_1$, y 9 para $GRASP_2$ y $GRASP_3$. Estos son, respectivamente, T_{α_1} y T_{α_1, α_2} .

Se optó por la aplicación de métodos no paramétricos para el análisis de los datos, ya que los contrastes previos de normalidad e igualdad de varianzas dieron respuestas negativas. Se usó el test de Friedman (ver, por ejemplo, [2]) para el análisis de los tratamientos con las 5 ejecuciones como bloques. Cuando se rechazó la hipótesis nula de igualdad de tratamientos se empleó el test de comparaciones múltiples de Friedman ([2] pág. 274) para determinar las diferencias entre los tratamientos.

En la tabla I se recoge el p -valor asociado con el estadístico de Friedman para los problemas considerados. Además, para aquellos problemas en los que se rechazó la hipótesis nula de igualdad de tratamientos, se muestra el tratamiento con menor rango promedio y, cuando el test de comparaciones múltiples de Friedman no detectó diferencias significativas entre ellos, el tratamiento con el segundo menor rango promedio (entre paréntesis).

De los resultados obtenidos se concluye que:

1. $GRASP_1$: el valor apropiado de α_1 es 0, es decir, el que se corresponde con un mejor ajuste;
2. $GRASP_2$: los valores de los parámetros dependen del tamaño del problema. Así, para problemas con 200 rectángulos o más, el tratamiento que presenta mejor comportamiento es $T_{0,0}$, con la única excepción del problema ($n = 200$, $w = 60$, $h_{opt} = 100$). Para problemas con 100 rectángulos, no existe una conclusión clara, ya que podría escogerse entre los tratamientos $T_{0,0}$ y $T_{0,0,2}$. Nos hemos decantado por esta última elección, teniendo en cuenta el número de veces que aparecen $T_{0,0}$ y $T_{0,0,2}$ como el mejor o el segundo mejor tratamiento. Cuando $n = 50$, las conclusiones son aún menos claras si comparamos por el número de veces que aparece un tratamiento como el mejor. Tener en cuenta el segundo mejor tratamiento, clarifica: el tratamiento apropiados es $T_{0,0,1}$.
3. $GRASP_3$: usando las medidas anteriores se sigue que, para problemas con 200 rectángulos o menos, las elecciones recomendadas son $\alpha_1 = 0$ y $\alpha_2 = \alpha_3 = 0,2$. Para problemas con 300 rectángulos, $\alpha_1 = \alpha_2 = \alpha_3 = 0$.

En los métodos $GRASP_4$ y $GRASP_5$, se fijó el valor de α_1 , α_2 y α_3 según la información obtenida en el análisis anterior, y se tomó $k = 6$ en todos los casos.

3.2. Comparativa y resultados para problemas grandes

Los problemas tests usados para comparar el comportamiento de nuestras propuestas aparecen en el trabajo de Hopper y Turton [4] (disponibles en la siguiente dirección: <http://mscmga.ms.ic.ac.uk/jeb/orlib/stripinfo.html>). Se trata de una batería de 21 problemas agrupados en 7 categorías de 3 problemas cada una. En las cuatro primeras columnas de la tabla II se recogen las características de estos problemas. La columna 5 de dicha tabla muestra el valor objetivo obtenido y el tiempo requerido (en minutos) por un Recocido Simulado que emplea la estrategia Botton-Left ($SA + BLF$) (los valores han

Procedimientos constructivos adaptativos para el problema del empaquetado bidimensional

Categoría	n	w	h_{opt}	$SA + BLF$	$GRASP_1$	$GRASP_2$	$GRASP_3$	$GRASP_4$	$GRASP_5$
C_1	16 ó 17	20	20	20,8	22,6	22	22	21,66	21,66
				0,7				0,21	0,19
C_2	25	40	15	15,9	17	17	16,33	16,33	16,33
				2,4				0,20	0,20
C_3	28 ó 29	60	30	31,5	33,66	35,33	33,66	33,66	33,33
				4				0,31	0,34
C_4	49	60	60	61,8	62,66	64,33	63	63,33	63
				33				0,49	0,35
C_5	72 ó 73	60	90	92,7	94,33	94	93	92,66	92,33
				115				0,43	0,35
C_6	97	80	120	123,6	125,33	124,33	124	123	123,33
				382				0,47	0,63
C_7	196 ó 197	160	240	249,6	247	245	246	244,66	245
				4181				0,77	0,80

Cuadro II. Mejores valores objetivos y tiempo requerido (valores promedios por categoría).

n	w	h_{opt}	$GRASP_4$		$GRASP_5$	
1000	300	400	403	3,02	402,6	2,536
1000	300	450	453	3,069	452,6	2,496
1000	350	400	403	3,098	403	2,9
1000	350	450	453	3,034	453	2,438
1000	400	500	503	3,054	503	2,702
1000	400	550	553	2,802	553	2,624

Cuadro III. Resultados obtenidos con $GRASP_4$ y $GRASP_5$ en problemas grandes.

sido extraídos del trabajo de Hopper y Turton [4]). Este procedimiento es el que mejor comportamiento presenta de entre todas las propuestas de Hopper y Turton [4]. Según nuestro conocimiento, es el mejor procedimiento conocido hasta el momento.

Desde la columna 6 hasta la 10 se presentan las mejores soluciones y el tiempo (en segundos) requerido por nuestras propuestas. Para los métodos $GRASP_1$, $GRASP_2$ y $GRASP_3$ no se muestran tiempos, ya que éstos son insignificantes. Hay que destacar la considerable disminución que se produce en el tiempo cuando se emplea cualquiera de nuestras propuestas. Para la categoría C_7 se pasa de 4181 minutos a 0,80 segundos y se obtienen soluciones mejores a las obtenidas por $SA + BLF$. Otra característica destacable es el lento crecimiento que se produce en el tiempo con el aumento del tamaño del problema.

Por último, para mostrar el comportamiento de nuestras propuestas en problemas grandes, se resolvieron con $GRASP_4$ y $GRASP_5$ problemas con 1000 rectángulos. En la tabla III se muestran los resultados obtenidos. En las primeras tres columnas se recogen el número de rectángulos, la anchura del objeto rectangular y el objetivo óptimo. Las últimas cuatro columnas muestran la mejor solución obtenida y el tiempo (en segundos) requerido por $GRASP_4$ y $GRASP_5$. Hay que destacar la calidad de las soluciones obtenidas y el poco tiempo empleado en obtenerlas.

4. Conclusiones y Trabajos Futuros

Hemos diseñado e implementado diferentes procedimientos constructivos para el problema del empaquetado rectangular bidimensional no guillotina. Además, hemos analizado estadísticamente la influencia de los correspondientes parámetros sobre la calidad de las soluciones. Este estudio nos permitió ajustar convenientemente los valores de aquellos. A continuación, comparamos la calidad de las soluciones obtenidas frente a las propuestas del trabajo de Hopper y Turton [4]. De los resultados obtenidos se sigue que nuestros métodos presentan un comportamiento superior al de tales propuestas: suministran soluciones de alta calidad en mucho menos tiempo. Como trabajos futuros podemos enumerar:

1. *Mejorar el postprocesamiento.* Tras obtener una solución, aplicamos un postprocesamiento en el que se realiza una búsqueda exhaustiva del mejor orden de empaquetado para los últimos k rectángulos de la solución. Debido a la explosión combinatoria que se origina, hemos tenido que limitar esta búsqueda a unos pocos rectángulos. Pretendemos mejorar esta fase aplicando otras estrategias de búsqueda: multiarranque, recocido simulado, ... como postprocesamiento.
2. *Determinar apropiadamente el parámetro para la fase de postprocesamiento.* Hasta ahora el valor de k para el postprocesamiento se determina a priori por el usuario. Sería conveniente disponer de un método que, atendiendo a las características de la solución, determine cuál es el rectángulo a partir del cual realizar el postprocesamiento.
3. *Adaptar los métodos propuestos a otros problemas de empaquetado de rectángulos.* Con ligeras modificaciones, los métodos propuestos se podrían aplicar al resto de los problemas de empaquetado de rectángulos enunciados en la introducción. Es nuestro propósito realizar y valorar esta adaptación.

Referencias

- [1] **S. Benati.** An algorithm for a cutting stock problem on a strip. *Journal of Operational Research Society* Vol. 48 pp. 288-294 (1997)
- [2] **W. W. Daniel.** *Applied Nonparametric Statistics.* PWS-Kent Publishing Company, Boston, 1990.
- [3] **T. A. Feo, M.G.C. Resende.** Greedy Randomized Adaptive Search Procedures. *Journal of Global Optimization* Vol. 6 pp. 109-133 (1995)
- [4] **E. Hopper, B. Turton.** An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem. *European Journal of Operational Research* Vol. 128 pp. 34-57 (2001)
- [5] **I. Hwang.** An efficient processor allocation algorithm using two dimensional packing. *Journal of Parallel and Distributed Computing* Vol. 42 pp. 75-81 (1997)