ALGORITMO HÍBRIDO CÚMULO DE PARTÍCULAS Y BÚSQUEDA EN VECINDARIO VARIABLE USADO EN LA SOLUCIÓN DEL PROBLEMA DE LA MOCHILA BIDIMENSIONAL

Hybrid algorithm particle swarm optimization and variable neighborhood search for the two-dimensional guillotineable single knapsack problem

RESUMEN
En este documento se presenta un algoritmo de optimización hibrido cúmulo de partículas y vecindario variable el cual utiliza una propuesta de codificación basada en árbol binario de cortes para resolver el problema de la mochila bidimensional guillotinada. Este es un problema clásico de optimización caracterizado por su alta complejidad computacional y de gran aplicación en el área de la producción. En este artículo se trataron dos de sus variantes, denominados: con y sin rotación de las piezas 90°. Con el fin de evaluar el desempeño del algoritmo se usaron sistemas de prueba de la literatura especializada obteniéndose excelentes resultados.

PALABRAS CLAVES: búsqueda en vecindario variable, cúmulo de partículas, mochila bidimensional guillotinada.

ABSTRACT
In this document we present an hybrid optimization algorithm Particle Swarm Optimization and Variable Neighborhood Search, using an encoding based in binary trees of cutting to solve the two-dimensional guillotine single knapsack problem. This is a well known classic optimization problem for its high computational complexity and its big applications in the field of production. We consider the problem with and without 90° rotations. The computational results were compared on a kind of benchmark problems, indicating that the algorithm reaches excellent solutions.

KEYWORDS: *particle swarm optimization, two dimensional guillotineable single knapsack problem, variable neighborhood search*

DAVID ÁLVAREZ MARTÍNEZ
Ingeniero de Sistemas y Computación.
Joven Investigador
COLCIENCIAS-Universidad Tecnológica de Pereira
davidalv@utp.edu.co

ELIANA TORO OCAMPO
Ingeniera Industrial, *M.Sc.*
Docente Asistente
Facultad de Ingeniería Industrial
Universidad Tecnológica de Pereira
elianam@utp.edu.co

RAMÓN GALLEGO RENDÓN
Ingeniero Electricista, *Ph.D.*
Docente Titular
Programa Ingeniería Eléctrica
Universidad Tecnológica de Pereira
ragr@utp.edu.co

1. INTRODUCCIÓN

Los grandes desperdicios que se generan en la industria que usa insumos básicos como la madera, el plástico, cartón, vidrio, se deben a un inadecuado control de éstos y a la carencia de una política operativa bien definida, implementada y ejecutada. La eliminación del desperdicio, se traduce en ahorros millonarios para las empresas, lo que les permite ser más competitivas en costos.

Los problemas de corte y empaquetamiento están inmersos en el manejo eficiente de las materias primas y revisten una alta complejidad matemática, enmarcados en la categoría de problemas NP-Duros. Esto es demostrado a través del caso especial donde todos los ítems tienen la misma longitud y diferente altura, el bien conocido problema de empaquetamiento óptimo unidimensional. El problema de empaquetamiento óptimo de una dimensión fue probado NP-Duro por Garey y Johnson [1] y Martello *et al.*[2]. El que un problema este catalogado en esta categoría no significa que no puede resolverse, sino que se deben proponer algoritmos de solución que exploten de forma eficiente la estructura matemática del mismo para que se encuentren soluciones a la mayoría de las instancias del problema en tiempos de ejecución relativamente pequeños.

En este estudio se propone un algoritmo que reúne las principales características de las técnicas cúmulo de partículas y búsqueda en vecindario variable. El algoritmo es verificado con casos de prueba de la literatura especializada obteniendo resultados de excelente calidad.

Este documento tiene la siguiente presentación: inicialmente se hace una descripción del problema, en la sección 3 se presenta el modelo matemático, en la sección 4 se describen las técnicas metaheurísticas utilizadas, en la sección 5 se presenta la metodología de solución, así como la codificación de árbol binario de cortes y el algoritmo híbrido, en la sección 6 se presentan el análisis de los resultados obtenidos de las distintas instancias evaluadas y finalmente en 7 se muestran las

Fecha de Recepción: 17 de junio de 2010
Fecha de Aceptación: 13 de Agosto de 2010

conclusiones, recomendaciones y propuestas para trabajos futuros.

2. DESCRIPCIÓN DEL PROBLEMA

Washer *et al*. en [3] caracterizan los problemas de la mochila como problemas de empaquetamiento con un surtido de piezas fuertemente heterogéneo que debe ser ubicado en un conjunto de contenedores. La disponibilidad de contenedores es limitada de tal manera que no es suficiente para ubicar todas las piezas. El valor de las piezas embaladas debe ser maximizado.

En Washer *et al*. [3] es generada la categoría *single knapsack problem* (SKP), dentro de esta se encuentran problemas como la mochila clásica (*One-Dimensional Knapsack Problem*), también llamada mochila 0-1 (Martello *et al*. [4]).

Caprara y Monaci [5] presentan el problema de una mochila bidimensional ortogonal, donde un conjunto de pequeños rectángulos de tamaño y costo asociado dado tiene que ser cortado de un gran rectángulo (mochila), con el fin de maximizar el beneficio de las piezas cortadas, siendo este el problema de estudio propuesto en este trabajo, con algunas restricciones y variantes adicionales:

- Solo se permite el uso de cortes tipo guillotina, es decir, los cortes van de un extremo al otro del rectángulo original (guillotinado).
- La orientación de las piezas a ser ubicadas, es decir, una pieza de alto *h* y ancho *w* es diferente de una pieza de longitud *w* y alto *h* (sin rotación). Si se considera que las dimensiones (*h*, *w*) y (*w*, *h*) representan las dimensiones de la misma pieza, se está abordando un problema con rotación.

Para instancias de pequeña y mediana escala algunos algoritmos han sido propuestos. Christofides y Whitlock [6] y Hifi y Zissimopoulos [7]. Recientemente, Fayard et al. [8] extendieron las ideas de Fayard y Zissimopoulos [9] desarrollando un algoritmo de propósito general para los problemas de la mochila.

Morabito *et al*. [10] desarrollaron la heurística DH (de las siglas en ingles *Depth-first search y Hill-climbing strategies*) basado en un proceso de búsqueda primero en profundidad y estrategias de aceptación de empeoramientos. El algoritmo KD (de las siglas en ingles *Knapsack problem* usando *Dynamic programming*) fue presentado por Fayard y Zissimopoulos [9] para resolver el problema basado en la solución de problemas de la mochila unidimensional. Diferentes aproximaciones heurísticas han sido desarrolladas para la solución del problema. Álvarez-Valdes *et al*. [11] presentan los algoritmos GRASP y Búsqueda Tabú. Mientras Gun *et al*. [12] propusieron un algoritmo que parte de una solución inicial de buena calidad y utiliza el algoritmo constructivo *bottom-up* para guiar el proceso.

3. MODELO MATEMÁTICO DE LA MOCHILA

Se presenta un modelo de programación lineal entera mixta para el problema de empaquetamiento óptimo de la mochila bidimensional guillotinada basado en la caracterización de los patrones de corte tipo guillotina y el uso de coordenadas donde pueden ser ubicadas las piezas. Este es una adaptación del modelo propuesto por Ben *et al*. [13] para el problema de empaquetamiento en rollos infinitos (*strip packing problem*).

Para obtener un modelo lineal entero, se usará el siguiente conjunto de variables binarias para representar la ubicación de las piezas en la hoja de material:

$$z_{i,j,k} = \begin{cases} 1 & \text{si la pieza } k \text{ es empacada en } (i,\ j) \\ 0 & \text{de lo contrario} \end{cases}$$

Las siguientes variables de decisión intermedias también son necesarias para garantizar que no existan traslapes entre piezas:

$$u_{i,j,i'} = \begin{cases} 1 & \text{si la pieza en } (i,\ j),\ \text{no excede} \\ & \text{(horizontalmente) } x_{i'} \text{ con } i' > i \\ 0 & \text{de lo contrario} \end{cases}$$

$$v_{i,j,j'} = \begin{cases} 1 & \text{si la pieza en } (i,\ j),\ \text{no excede} \\ & \text{(verticalmente) } y_{j'} \text{ con } j' > j \\ 0 & \text{de lo contrario} \end{cases}$$

Los siguientes tres conjuntos de variables binarias son necesarios para garantizar las restricciones guillotina:

$$p_{i_1,i',j_1,j_2} = \begin{cases} 1, & \text{si no hay pieza entre } (i_1,\ j_1) \text{ y } (i'-1,\ j_2) \\ & \text{que exceda } x_{i'} \text{ (consecuentemente,un corte} \\ & \text{vertical en } x_{i'} \text{ no cruza ninguna pieza} \\ & \text{empacada entre } (i_1,\ j_1) \text{ y } (i'-1,\ j_2)),\ i' > i_1 \\ 0, & \text{de lo contrario} \end{cases}$$

$$q_{i_1,i_2,j_1,j'} = \begin{cases} 1, & \text{si no hay pieza entre } (i_1,\ j_1) \text{ y } (i_2,\ j'-1) \\ & \text{que exceda } y_{j'} \text{ (consecuentemente, un corte} \\ & \text{horizontal en } y_{j'} \text{ no cruza ninguna pieza} \\ & \text{empacada entre } (i_1,\ j_1) \text{ y} (i_2,\ j'-1)),\ j' > j_1 \\ 0, & \text{de lo contrario} \end{cases}$$

$$d_{i_1,i_2,j_1,j_2} = \begin{cases} 1, & \text{si existe mínimo una pieza} \\ & \text{empacada entre } (i_1,\ j_1) \text{ y } (i_2,\ j_2) \\ 0, & \text{de lo contrario} \end{cases}$$

La formulación completa del problema es la siguiente:

$$\text{Maximizar} \quad \sum_{i=1}^{n}\sum_{j=1}^{n} c_k z_{i,j,k} \quad \forall k, \qquad (1)$$

sujeto a

$$0 \le x_1 \le x_2 \le \text{K} \le x_n, \qquad (2)$$

$$0 \le y_1 \le y_2 \le \text{K} \le y_n, \qquad (3)$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} z_{i,j,k} \le 1 \quad \forall k, \tag{4}$$

$$\sum_{i=1}^{n}\sum_{k=1}^{n} z_{i,j,k} \le 1 \quad \forall j, \tag{5}$$

$$\sum_{j=1}^{n}\sum_{k=1}^{n} z_{i,j,k} \le 1 \quad \forall i, \tag{6}$$

$$x_{i'} - x_i - \sum_{k=1}^{n} w_k z_{i,j,k} \ge \left(u_{i,j,i'} - 1\right) W \quad \forall i, \forall j, \forall i' > i, \tag{7}$$

$$x_{i'} - x_i - \sum_{k=1}^{n} w_k z_{i,j,k} \ge u_{i,j,i'} W \quad \forall i, \forall j, \forall i' > i, \tag{8}$$

$$y_{j'} - y_j - \sum_{k=1}^{n} h_k z_{i,j,k} \ge \left(v_{i,j,i'} - 1\right) H \quad \forall i, \forall j, \forall j' > j, \tag{9}$$

$$y_{j'} - y_j - \sum_{k=1}^{n} h_k z_{i,j,k} \ge u_{i,j,j'} H \quad \forall i, \forall j, \forall j' > j, \tag{10}$$

$$\left(1 - d_{i_1,i_2,j_1,j_2}\right) n \ge \sum_{i=i_1}^{i_2}\sum_{j=j_1}^{j_2}\sum_{k=1}^{n} z_{i,j,k} - 1 \quad \forall i_1, \forall j_1, \forall i_2 > i_1, \forall j_2 > j_1, \tag{11}$$

$$p_{i_1,i',j_1,j_2} \le \sum_{j=j_1}^{j_2}\sum_{k=1}^{n} z_{i',j,k} \quad \forall i_1, \forall j_1, \forall j_2 > j_1, \forall i' > i_1, \tag{12}$$

$$p_{i_1,i',j_1,j_2} \le \sum_{i=i_1}^{i'-1}\sum_{j=j_1}^{j_2}\sum_{k=1}^{n} z_{i,j,k} \quad \forall i_1, \forall j_1, \forall j_2 > j_1, \forall i' > i_1, \tag{13}$$

$$\left(i' - i_1\right)\left(j_2 - j_1 + 1\right) p_{i_1,i',j_1,j_2} \le \sum_{i=i_1}^{i'-1}\sum_{j=j_1}^{j_2} u_{i,j,i'} \quad \forall i_1, \forall j_1, \forall j_2 > j_1, \forall i' > i_1, \tag{14}$$

$$q_{i_1,i_2,j_1,j'} \le \sum_{i=i_1}^{i_2}\sum_{k=1}^{n} z_{i,j',k} \quad \forall i_1, \forall j_1, \forall i_2 > i_1, \forall j' > j_1, \tag{15}$$

$$q_{i_1,i_2,j_1,j'} \le \sum_{i=i_1}^{i_2}\sum_{j=j_1}^{j'-1}\sum_{k=1}^{n} z_{i,j,k} \quad \forall i_1, \forall j_1, \forall i_2 > i_1, \forall j' > j_1, \tag{16}$$

$$\left(j' - j_1\right)\left(i_2 - i_1 + 1\right) q_{i_1,i_2,j_1,j'} \le \sum_{i=i_1}^{i_2}\sum_{j=j_1}^{j'-1} v_{i,j,i'} \quad \forall i_1, \forall j_1, \forall j_2 > i_1, \forall j' > j_1, \tag{17}$$

$$d_{i_1,i_2,j_1,j_2} + \sum_{i'=i_1+1}^{i_2} p_{i_1,i',j_1,j_2} + \sum_{j'=j_1+1}^{j_2} q_{i_1,i_2,j_1,j'} \ge 1 \quad \forall i_1, \forall j_1, \forall i_2 > i_1, \forall j_2 > j_1, \tag{18}$$

$$W \ge x_i + \sum_{j=1}^{n}\sum_{k=1}^{n} w_k z_{i,j,k} \quad \forall i, \tag{19}$$

$$H \ge y_j + \sum_{i=1}^{n}\sum_{k=1}^{n} h_k z_{i,j,k} \quad \forall j, \tag{20}$$

$$z_{i,j,k} \in \{0,1\} \quad \forall i, \forall j, \forall k, \tag{21}$$

$$u_{i,j,i'} \in \{0,1\} \quad \forall i, \forall j, \forall i' > i, \tag{22}$$

$$v_{i,j,j'} \in \{0,1\} \quad \forall i, \forall j, \forall j' > j, \tag{23}$$

$$d_{i_1,i_2,j_1,j_2}, p_{i_1,i_2,j_1,j_2}, q_{i_1,i_2,j_1,j_2} \in \{0,1\} \quad \forall i, \forall j, \forall i_2 > i_1, \forall j_2 > j_1, \tag{24}$$

En la formulación anterior, las restricciones (4), (5) y (6) aseguran que cada posición horizontal o vertical sea ocupada por exactamente una pieza y cada pieza es empacada exactamente una vez.

Las restricciones [(7)-(10)] son restricciones para garantizar que no existan traslapes entre piezas. Las restricciones [(11)-(18)] son restricciones guillotina para cada área rectangular. Si las restricciones guillotina son satisfechas para cada área rectangular, entonces también se cumple para todo el patrón de corte.

Las restricciones (19) y (20) garantizan que ninguna pieza exceda horizontalmente el ancho *W* y que ninguna pieza exceda verticalmente la altura *H*, considerando que la función objetivo es maximizar la sumatoria del costo asociado de las piezas empacadas en la placa (ver ecuación (1)).

4. TÉCNICAS DE OPTIMIZACIÓN

El algoritmo de optimización cúmulo de partículas (*Particle Swarm Optimization*, PSO) usa un simple mecanismo que imita el comportamiento de las bandadas de pájaros y los cardumes de peces en cuanto a la forma en que guían sus desplazamientos a fin de encontrar alimento y refugio, considerados como su objetivo primordial [14].

Este algoritmo puede ser computacionalmente ineficiente por que puede quedar atrapado fácilmente en óptimos locales cuando resuelve problemas cuyo espacio de solución es multimodal, por lo cual para mejorar la eficiencia y acelerar el proceso de búsqueda es fundamental determinar el estado de evolución y los mejores valores para los parámetros, con el fin de evitar posibles óptimos locales en el estado de convergencia. La tabla 1 presenta el algoritmo básico PSO.

PSO
1. Generar una población inicial de partículas, con posiciones y velocidades en el espacio del problema *d* dimensional (generada de forma aleatoria). 2. Repita: 2.1. Evaluar la función objetivo a cada partícula 2.2. Comparar el valor de la función de adaptación (*fitness*) de la partícula con el *pbest* de la partícula. Si su valor corriente es mejor que el *pbest* pasa a ser igual al valor de la *fitness* de la partícula y la localización de la *pbest* pasa a ser igual a la localización actual del espacio *d* dimensional 2.3. Comparar el valor de la función *fitness* con el mejor valor de adaptación de la población. Si el valor actual es mejor que el g*best*, actualizar el valor del g*best* 2.4. Calcular la velocidad y actualizar la posición. 3. Hasta la condición de parada.

Tabla 1. Algoritmo optimización con cúmulo de partículas

La búsqueda de vecindario variable (*Variable Neighborhood Search*, *VNS*) es una metaheurística reciente para resolver problemas de optimización cuya idea básica es el cambio sistemático de vecindario dentro de una búsqueda local [15]. La tabla 2 presenta el algoritmo básico VNS.

La VNS está basada en tres hechos simples:

- Un mínimo local con una estructura de vecindad no lo es necesariamente con otra.
- Un mínimo global es mínimo local con todas las posibles estructuras de vecindarios.
- Para muchos problemas, los mínimos locales con la misma o distinta estructura de vecindad están

relativamente cerca (este último es una aserción empírica).

VNS Básico
1. Seleccionar un conjunto de estructuras de entornos N_k; $k = 1,2,..., k_{max}$, que se usarán en la búsqueda 2. Encontrar una solución inicial S 3. Repita: 3.1. Para $k = 1$ hasta k_{max} haga: a. Generar al azar una solución S' del k-ésimo vecindario de S ($S' \in N_k(x)$) b. Aplicar algún método de búsqueda local con S' como solución inicial; denótese con S'' el mínimo local así obtenido c. Si la solución obtenida S'' es mejor que S, haga $S = S''$ y $k = 1$ d. De lo contrario, haga $k = k + 1$ 4. Hasta la condición de parada.

Tabla 2. Algoritmo búsqueda en vecindario variable

5. METODOLOGÍA

La codificación presentada se basa en la adecuación de la metodología propuesta en [16] denominada codificación de árbol binario de cortes.

Esta consiste en dividir la mochila en subespacios. Para garantizar que los subespacios creados presenten cortes de tipo guillotina, se define una codificación de árbol binario complementario de la siguiente forma:

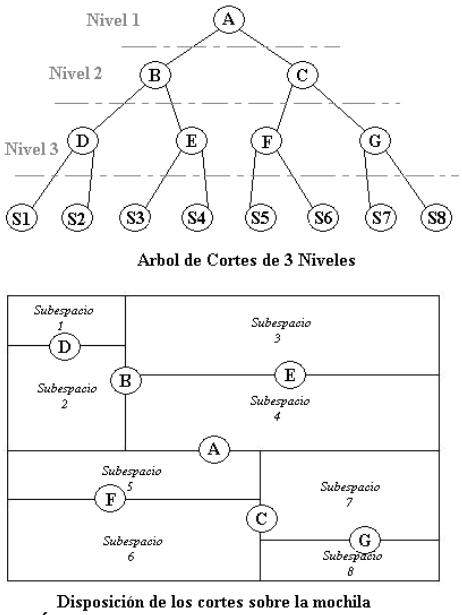

Figura 1. Árbol y disposición de cortes sobre la mochila.

Para representar la estructura que generan los subespacios son definidos dos árboles de tamaño igual al número de cortes: un árbol O de tipo binario que define la orientación de corte (vertical u horizontal); el segundo, D, es un árbol con valores reales entre 0 y 1 que determina la distancia porcentual desde la esquina inferior de la mochila disponible para determinar la posición del corte.

Toro et al. en [17] sugieren como resultado de un estudio estadístico, delimitar el árbol de cortes al uso de solo árboles binarios completos con tres niveles. En la figura 1 se muestra una alternativa de distribución de la mochila.

El esquema de optimización para el problema es ilustrado en la figura 2, en este el algoritmo *I* realiza una búsqueda exhaustiva sobre el árbol *O*, mientras que el algoritmo *II* recibe todos árboles *O* posibles. El algoritmo *II* corresponde al algoritmo de optimización metaheurístico, que consiste en encontrar los valores óptimos del árbol *D*.

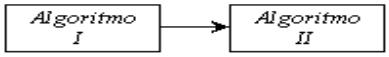

Figura 2. Esquema de la metodología.

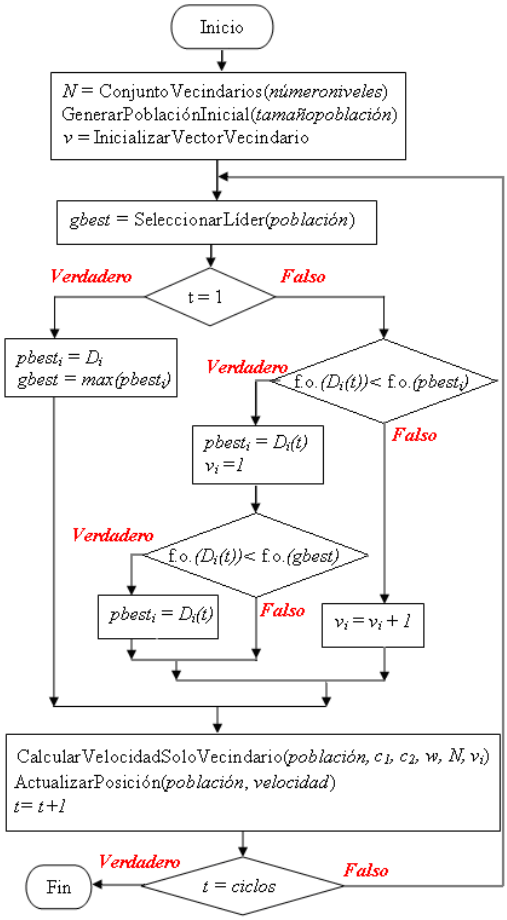

Figura 3. Algoritmo Híbrido PSO y VNS.

Algoritmo Híbrido Cúmulo de Partículas y Búsqueda en Vecindario Variable ($A_{PSO+VNS}$)

Este algoritmo es una combinación entre el algoritmo PSO y VNS. El algoritmo PSO presenta muchos parámetros que deben ser calibrados, la inadecuada

calibración de estos puede llevar el proceso a una convergencia prematura. Se adapta el algoritmo VNS con el fin de controlar el número de nodos del árbol de distancias a los cuales se les actualiza la posición.
La inclusión de este procedimiento en la metodología presenta como desventaja la desmejora en el tiempo computacional y como ventaja una mejora en la diversificación de la población. Lo anterior se logra dado que en cada iteración solo se calculan las velocidades de los nodos que hacen parte del vecindario de la partícula. La figura 3 ilustra en el diagrama del algoritmo $A_{PSO+VNS}$.

Calibración de parámetros

Una etapa de gran importancia en la aplicación de las técnicas metaheurísticas es el ajuste de parámetros. En general no existe un método exacto y eficiente para realizar la calibración de parámetros de las diferentes técnicas metaheurísticas, comúnmente estos algoritmos son parametrizados a través de la combinación de una búsqueda exhaustiva y una análisis estadístico de la calidad de los resultados. (Zhi-Hui *et al*. [14]) presenta un rango de valores reducido para los parámetros del algoritmo PSO. Teniendo como base los rangos presentados por Zhi-Hui el tamaño de la malla se reduce considerablemente. Los valores resultantes de la calibración de parámetros son ilustrados en la tabla 3.

Algoritmo	Parámetros	Valor
$A_{PSO+VNS}$	Tamaño de la Población	200
	Número de Ciclos	500
	c1 (Conocimiento Individual)	2,05
	c2 (Conocimiento Grupal)	2,05
	w (Inercia)	0,7

Tabla 3. Parámetros y valores usados ($A_{PSO+VNS}$).

6. ANÁLISIS DE RESULTADOS

Fueron seleccionados 15 casos de prueba para el problema de la mochila bidimensional guillotinada, *TH1* y *TH2* propuestos por Tschöke y Holthöfer [18], *CHW1* y *CHW2* propuestos por Christofides y Whitlock [6] y [*CW1-CW11*] propuestos por Fayard *et al.* [8]. Estos casos presentan 15 tipos de mochilas diferentes con distribuciones entre 23 y 168 piezas, la base de datos es presentada por Fayard *et al*. [8] y disponible en línea en Hifi [19]. Diferentes estudios han utilizado estos casos de prueba para realizar una especie de benchmark de las metodologías propuestas.

Todos los algoritmos fueron desarrollados en Delphi 7,0 ® y ejecutados sobre un ordenador con unas especificaciones de un procesador Pentium ® IV de 3,0 GHz y una memoria RAM de 1 GB.

Se presentan a continuación para todos los casos de prueba de cada tipo de problema la mejor solución reportada en la literatura especializada (*Best Known Solution*). Los resultados de Álvarez-Valdes *et al*. [11] presentan las mejores soluciones aunque trabajos como Fayard *et al*. [8] alcanzan buenos resultados. Para el problema de la mochila con rotación no tiene respuesta reportada debido a que la mayoría de aproximaciones a estos casos de gran complejidad matemática fueron realizadas mediante el uso de técnicas exactas donde el esfuerzo computacional es demasiado alto para su solución. Por lo tanto, algunos autores omiten reportarlas.

Caso	***Best Known Solution***	**Solución Propuesta $A_{PSO+VNS}$**		**Tiempo Utilizado (Segundos)**	
		Sin Rotación	**Con Rotación**	**Sin Rotación**	**Con Rotación**
CW1	6402	6402	**6764**	74	124
CW2	5354	5354	**5540**	68	126
CW3	5689	5687	5689	104	202
CW4	6170	**6175**	**7466**	94	186
CW5	11644	11644	**11659**	100	202
CW6	12923	12651	**12959**	162	320
CW7	9898	9898	**10880**	134	310
CW8	4605	4504	**4736**	184	366
CW9	10748	10748	**11479**	144	280
CW10	6515	6515	**6705**	142	280
CW11	6321	6321	**6604**	124	250
TH1	4620	4620	**4630**	86	168
TH2	9700	9560	9683	54	104
CHW1	2892	2730	2858	26	52
CHW2	1860	1860	**1900**	68	126

Tabla 4. Resultados alcanzados por A_{PSO+SA}.

La tabla 4 presenta los resultados obtenidos por el algoritmo propuesto en este estudio y los tiempos empleados para alcanzar estas respuestas.

	Sin rotación	Con Rotación
Igualados	10	1
Superados	1	12
Inferiores	4	2

Tabla 5. Comparación de resultados.

La tabla 5 resume los resultados obtenidos por el algoritmo propuesto y son comparados con las mejores respuestas reportadas en la literatura especializada, como resultado de esta comparación se tiene:

- Igualados, cuando la respuesta obtenida por el algoritmo propuesto es igual a la *best known solution*
- Superados, cuando la respuesta obtenida por el algoritmo propuesto es mejor a la *best known solution*

- Inferiores, cuando la respuesta obtenida por el algoritmo propuesto es de inferior calidad a la *best known solution*

Cuando se soluciona el problema de la mochila sin rotación de piezas para instancias de alta complejidad matemática usando el algoritmo propuesto, se alcanzan respuestas de excelente calidad y en tiempos computacionalmente razonables. En especial para el caso CW4 se mejora la respuesta reportada en la literatura.

Además, cuando se soluciona el problema de la mochila con rotación de piezas, usando el algoritmo propuesto, se obtiene un comportamiento excelente, al comparar respecto a las respuestas reportadas en la literatura para el problema sin rotación, la inclusión de rotación de las piezas reviste un gran aumento en la complejidad del problema y un aumento en los tiempos de cómputo pero permite alcanzar soluciones de mejor calidad.

7. CONCLUSIONES Y RECOMENDACIONES

Se resolvió el problema de la mochila bidimensional guillotinada, con y sin rotación de las piezas mediante un algoritmo híbrido de cúmulo de partículas, y búsqueda en vecindario variable, obteniéndose resultados de excelente calidad.

Se utilizó un tipo de codificación en árbol, llamada árbol de cortes, que combina un árbol de valores binarios, en el cual se orientan los cortes, con un árbol de valores reales y el cual determina las distancias de los cortes, presentando un gran desempeño para este tipo de problemas.

Fue implementado un algoritmo de optimización que combina las principales características de cúmulo de partículas y búsqueda en vecindario variable. El primero se considera como el algoritmo principal y el segundo se usa como mecanismo limitante del número de características de las partículas a las que se le calcula la velocidad. Esto permite mejorar el mecanismo de exploración del algoritmo PSO básico.

El método de solución usado presentó gran desempeño en la solución del problema considerando las dos variantes con y sin rotación de piezas.

Este problema es aplicable en diversos sectores de la economía, entre los que destacan los sectores de industria, transporte y comercio. Así por ejemplo, su aplicación se presenta en la industria textil, metalmecánica, papelera, vidriera, cuerera, transporte y almacenaje de mercancías, entre otras.

La metodología propuesta es aplicable a problemas como la mochila irrestricta, *bin packing*, *strip packing*, entre otros. También puede ser extendible a problemas de empaquetamiento en tres dimensiones.

8. BIBLIOGRAFÍA

[1] M. R. Garey y D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, San Fran., Calif., USA, 1979.
[2] S. Martello, M. Monaci, y D. Vigo, "An exact approach to the strip-packing problem", *INFORMS Journal on Computing*, vol. 15, no. 3, pp. 310–319, 2003.
[3] G. Wäscher, H. Haußner y H. Schumann, "An improved typology of cutting and packing problems". Euro. J. Oper. Res., vol. 183, pp. 1109–1130, 2007.
[4] S. MARTELLO, D. PISINGER, y P. TOTH, "New trends in exact algorithms for the 0-1 knapsack problem", *Euro. J. Oper. Res.*, vol. 123, pp. 325–332, 2000.
[5] A. CAPRARA, y M. MONACI, "On the two-dimensional knapsack problem", *Oper. Res. Letters*, vol. 32, pp. 5–14, 2004.
[6] N. Christofides, y C. Whitlock, "An algorithm for two-dimensional cutting problems". *Oper. Res.*, vol. 25, pp. 31-44, 1977.
[7] M. Hifi y V. Zissimopoulos, "A recursive exact algorithm for weighted two-dimensional cutting". *Euro. J. Oper. Res.*, vol. 91, pp. 553–564, 1996.
[8] D. Fayard, M. Hifi y V. Zissimopoulos, "An efficient approach for large-scale two-dimensional guillotine cutting stock problems", *J. of the Oper. Res. Society*, vol. 49, pp. 1270-1277, 1998.
[9] D. Fayard y V. Zissimopoulos, "An approximation algorithm for solving unconstrained two-dimensional knapsack problems". *Euro. J. of Oper. Res.*, vol. 84, pp. 618-632, 1995.
[10] R. Morabito, M. Arenales y V. Arcaro, "An and-or-graph approach for two-dimensional cutting problems". *Euro. J. of Oper. Res.*, vol. 58, pp. 263-271, 1992.
[11] R. Alvarez-Valdés, A. Parajón y J. M. Tamarit, "A tabu search algorithm for large-scale guillotine (un)constrained two-dimensional cutting problems", *Computers & Operations Research*, vol. 29, pp. 925-947, 2002.
[12] Y. Gun, M. Kang y J. Seong, "A best-first branch and bound algorithm for unconstrained two-dimensional cutting problems", *Oper. Res. Letters*, vol. 31, pp. 301–307, 2003.
[13] S. Ben, C. Chu y M. L. Espinouse, "Characterization and modelling of guillotine constraints", *Euro. J. of Oper. Res.*, vol. 191, pp. 112–126, 2008.
[14] Z. Zhi-Hui, Z. Jun, L. Yun y S. Henry, "Adaptive Particle Swarm Optimization", *IEEE Trans On Sys, Man, And Cyber—Part B: Cyber*, vol. 39, Issue 6, pp. 1362-1381, 2009.
[15] N. Mladenović y P. Hansen, "Variable neighborhood search", *Computers and Operations Research*, Vol. 24, pp. 1097–1100, 1997.
[16] D. F. Wong, H. W. Leong y C. L. Liu, *Simulated Annealing for VLSI Design*, Kluwer Academic Publishers, 1988.
[17] E. Toro, A. Garcés y H. Ruiz, "Solución al problema de empaquetamiento bidimensional usando un algoritmo híbrido constructivo de búsqueda en vecindad variable y recocido simulado", *Rev. Ing. Univ. de Ant.*, Vol. 46, pp. 119-131, 2008.
[18] S. Tschöke y N. Holthöfer, "A new parallel approach to the constrained two-dimensional cutting stock problem". *Techical Report*, Univ. of Padeborn, D.C.S. 33095 Padeborn. Germ., 1996.
[19] M. Hifi, *Problem instances for the Cutting/Packing Problems*, [Online]. Available: ftp://cermsem.univ-paris1.fr/pub/CERMSEM/hifi/2Dcutting/.