Solving the Two-Dimensional Knapsack Problem Considering Cutting-Time and Emission of Particulate Matter in the Metalworking Industry

P. Velasco-Carvajal, G. Camacho, D. Cuellar-Usaquen, and D. Álvarez-Martínez

***Abstract*—This article presents a metaheuristic to solve the two-dimensional knapsack problem on applications of metalworking industries. The proposed algorithm is based on a GRASP procedure and its performance is assessed through comparison of the solutions computed for the knapsack problem. We have used two references for comparison: instances reported in the literature and experimental data obtained with two variations on the cutting method. The first reference allows us to conclude about the used area and the computational time. The second reference let us to explore the performance of the solutions with respect to the cutting-time and the amount of particles emitted. The methodology is composed by four steps: (1) selection of the instances, (2) design and implementation of the algorithms for optimization and post-optimization, (3) computing of the solutions and registering of cutting-times and particles emitted for each solution and (4) performance validation through comparison of solutions. The results permit to conclude a good behavior of the proposed algorithm reaching the performance reported in some instances and improving others. Additionally, the algorithm exhibits a stable computational time even for large-scale instances. It is emphasized that the post-optimization stage reduces the cutting-times without incrementing the emission of particulate matter. As future work we propose: to include a vision system to close the cutting loop; to consider partial damages on the plate; and to contemplate an exact approach for the sequencing cutting problem.**

***Index Terms*—GRASP, Industrial Robotics, Knapsack Problem, Particulate Matter.**

I. Introducción

El problema de la mochila bidimensional consiste en el corte ortogonal de un conjunto de piezas a partir de una pieza contenedora o placa original, de tal forma que se seleccionen y se corten las piezas que maximizan el beneficio, garantizando las restricciones de cortar únicamente las piezas con sus tamaños demandados y sin llegar a exceder los límites de la placa.

P. Velasco-Carvajal, Departamento de Ingeniería Industrial, Universidad de La Salle, Colombia, pvelasco15@unisalle.edu.co

G. A. Camacho, Facultad de Ingeniería, Universidad del Valle, Colombia, guillermo.camacho@correounivalle.edu.co

D. Cuellar-Usaquen, Departamento de Ingeniería Industrial, Universidad de Los Andes, Colombia, dh.cuellar@uniandes.edu.co.

D. Álvarez-Martínez, Departamento de Ingeniería Industrial, Universidad de Los Andes, Colombia, d.alvarezm@uniandes.edu.co.

Este problema descrito en [1] y [2], tiene como objetivo maximizar el beneficio asociado a las piezas cortadas o minimizar el desperdicio de material. En este trabajo se presenta la adaptación del algoritmo GRASP [3] para resolver el problema de la mochila bidimensional y se explora el efecto de las soluciones sobre la emisión de material particulado.

La adaptación implementada consiste en un procedimiento iterativo en dos fases: una fase de construcción, realizada por un algoritmo voraz en donde se obtiene una solución factible a partir de la adición incremental de piezas, que son seleccionadas pseudo-aleatoriamente de una lista restricta de mejores candidatos y una fase de mejora, realizada por un algoritmo de búsqueda local, donde se busca mejorar la solución obtenida por la fase de construcción, a través de la remoción de piezas y un rellenado determinístico. El algoritmo cuenta con un esquema de autoajuste de parámetros, en especial el tamaño de la lista restricta de candidatos y el criterio de selección de soluciones promisorias para entrar a la etapa de mejora. El desempeño del algoritmo propuesto se evalúa en instancias clásicas reportadas en la literatura. Los resultados obtenidos son comparados con las mejoras soluciones publicadas teniendo en cuenta los criterios de: calidad de la solución y tiempo computacional. El algoritmo presenta un comportamiento sobresaliente logrando mejorar algunas soluciones reportadas.

Usualmente los trabajos en la literatura solo velan por alcanzar los patrones de corte óptimos, en este trabajo estos patrones son integrados con un sistema automático de corte basado en el uso de robots industriales, con el fin de analizar y modelar restricciones del problema en aplicaciones de la industria metalmecánica. De forma tal que al reducir el desperdicio de material se explore el desempeño del sistema respecto al tiempo de ejecución de la tarea y la cantidad de partículas emitidas que ponen en riesgo la seguridad de los operadores. Para esto, se propone una etapa de post-optimización que logra mejorar los tiempos de ejecución, sin aumentar la cantidad de partículas emitidas y sin perder la calidad de solución en términos de desperdicio.

El desarrollo computacional involucrado en este proyecto consiste en un módulo de software que permite resolver el problema de optimización, un módulo hardware que permite automatizar las tareas de corte a través de manipuladores industriales, sistemas de comunicación que integran las tareas de optimización y automatización, y un sistema de información

para la trazabilidad, seguimiento y aseguramiento de la calidad que permite a los operarios visualizar los patrones de corte y las trayectorias del robot.

La organización del trabajo es la siguiente. En la sección II se presenta el marco teórico que incluye una descripción del problema a resolver y la adaptación del algoritmo GRASP propuesta en este trabajo. En la sección III se resume la metodología empleada para validar el desempeño del algoritmo propuesto. En la sección IV se presenta el análisis comparativo de los resultados obtenidos. Por último, en la sección V se presentan las conclusiones y recomendaciones para trabajos futuros.

II. Marco Teórico

A. *El problema de la Mochila Bidimensional Rectangular*

En este trabajo se considera la variante del problema en donde tanto las piezas requeridas como la placa original, son rectangulares. El problema puede ser formulado a partir de la descripción de los objetos del sistema: R, P, y las funciones π y b [4]. $R = (L, W)$ es la pieza original (mochila) con largo L y ancho W. $P = \{P_1, \quad P_2, \quad \cdots \quad P_m\}$ es un conjunto de m piezas con $P_i = (l_i, w_i)$; es decir, una pieza de longitud l_i y ancho w_i, para $i = 1,2, \ldots, m$. Finalmente, π y b son funciones que asignan a cada pieza P_i un peso o beneficio π_i y un número máximo de ejemplares a cortar b_i, respectivamente. El propósito es cortar del rectángulo R, x_i copias de la pieza P_i cumpliendo las restricciones (1−2): (1) no exceder la cantidad de piezas demandadas, $0 \leq x_i \leq b_i$ para $i = 1,2, \ldots, m$, (2) evitar que las piezas ubicadas en la placa se traslapan entre sí o excedan los límites de la placa original. Teniendo como objetivo maximizar el beneficio total $\sum_{i=1}^{m} \pi_i x_i$. Una secuencia de cortes (patrón de corte) de R en pequeñas piezas rectangulares será referida como solución. En la Tabla 1 se resumen las variantes más comunes del problema.

TABLA 1
VARIANTES DEL PROBLEMA TRATADO

Característica	Posibles valores
Costo de la pieza	Ponderado (*weigthed*)
	No ponderado (*unweigthed*)
Orientación	**Fija (*fixed*)**
	Libre
Límite de ejemplares	Restricto
	Irrestricto
Patrón de corte	Guillotina
	Libre

La versión ponderada se caracteriza porque la función π es independiente del área de la pieza P_i. La versión con rotación se caracteriza porque (l_i, w_i) y (w_i, l_i) representan la misma pieza P_i. En la versión irrestricta todos los b_i son calculados como $\lfloor L/l_i \rfloor \lfloor W/w_i \rfloor$.

Finalmente, en la versión guillotina los cortes aplicados atraviesan la placa de un extremo a otro; note que esta última característica es propia de la tecnología de corte disponible para ejecutar el proceso. La variante del problema de corte abordado en este estudio se resalta en los valores de la Tabla 1, la cual corresponde a una aplicación práctica de una empresa de metalmecánica de la Industria colombiana.

B. *El problema de la Mochila Bidimensional Rectangular*

El algoritmo GRASP implementado combina un algoritmo constructivo con una estrategia de mejora. El algoritmo constructivo adiciona piezas P_i a la solución en cada iteración adecuando la versión presentada en [5] y [6] utilizando el concepto de espacios máximos. Los espacios máximos son una lista de los espacios vacíos de mayor área remanentes o disponibles para contener piezas P_i. Cada pieza que ingresa a la solución es asignada a un nuevo espacio, actualizando la lista de espacios máximos como se indica en la Figura 1. El Algoritmo 1 documenta la estrategia para resolver la etapa constructiva.

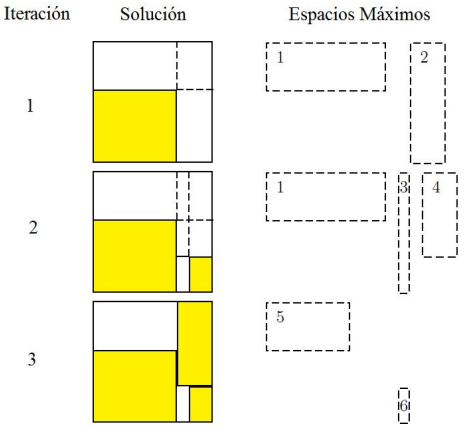

Fig. 1. Evolución de los espacios máximos en tres iteraciones de la fase constructiva. Fuente [7].

Este algoritmo utiliza como entradas una lista P con las piezas que esperan ser cortadas y el descriptor de la placa original R. Como variable local se declara e inicializa una lista de espacios máximos S y una lista $solución$. En la línea 3 el algoritmo aplica un mecanismo de selección para escoger un espacio máximo. Este mecanismo usa dos criterios: (1) distancia euclidiana mínima a una de las esquinas de R, (2) espacio máximo de menor área. Inicialmente se aplica el primer criterio, sólo en caso de empate, se aplica el segundo criterio. El primer criterio busca consumir espacios desde el extremo de la placa hacia el centro. El segundo criterio es la estrategia para usar los espacios máximos de mínima dimensión.

ALGORITMO 1. FASE CONSTRUCTIVA DEL ALGORITMO PROPUESTO

```
   function GRASP_Constructivo (P, R): List solución
1  inicializa S, solución
2  while P ≠ Ø or S≠ Ø do
3          s ←seleccioneMaximosEspacios(R, S)
4          p ←seleccionePieza(P,criterio,s, α)
5          solución←adicionePiezasASolucion(s,p,
           solución)
6          actualice(P,S,p,s)
7  end
8  return solución
   end
```

En la línea 4 se selecciona la pieza que será cortada utilizando dos criterios: (1) mejor encaje, (2) máxima área (*fitness)*. Para el mejor encaje, el algoritmo calcula la distancia entre el borde de cada pieza y el extremo del espacio vacío seleccionado.

Posteriormente, ordena la lista P en secuencia no decreciente de distancia. Finalmente, restringe la lista por medio del parámetro α (original del GRASP) y selecciona aleatoriamente una pieza de la lista de candidatos. El segundo criterio presenta el mismo comportamiento voraz y aleatorio, en el cual la pieza seleccionada aleatoriamente es una de los mejores candidatos que provocan mayor costo incremental para la función objetivo (máxima área o mayor beneficio). En la línea 5, se adiciona a la solución la pieza p seleccionada en la línea 4, en el espacio máximo s escogido en la línea 3. Cada elemento de la solución contendrá los siguientes descriptores: identificador, beneficio, dimensiones y localización de la pieza dentro de la placa original.

En la línea 6 se realiza la actualización de las listas S y P. Existen dos casos de actualización para S: (1) creación de nuevos espacios máximos y (2) eliminación de espacios máximos. El primer caso se ejecuta cuando la pieza p seleccionada no encaja de manera perfecta en el espacio máximo s. El segundo caso se ejecuta cuando hay encaje perfecto de p en s. El proceso se repite hasta que S o P sean vacíos, es decir, no hay más espacios máximos o no hay más piezas para cortar. Finalmente, en la línea 8 se retorna la solución calculada.

Detalle del componente aleatorio. Para cada tipo de pieza P_i, el método seleccionePieza construye dos capas (una horizontal y otra vertical), basado en un criterio de selección (mejor encaje, máxima área). Cada capa es llamada candidato y el conjunto de posibles candidatos es agrupado en la lista restricta de candidatos RCL. Posteriormente, se selecciona un candidato de forma aleatoria, basado en el parámetro clásico α ($\alpha \in [0, 1]$).

Movimiento de mejora. Consiste en eliminar el k% de las piezas seleccionadas en la solución. El valor k fue seleccionado en el rango [30, 90]. Posteriormente, las piezas removidas y las piezas que no se incluyeron en la solución, son procesadas por una versión determinística del algoritmo constructivo ($\alpha = 1$). En esta mejora, el algoritmo constructivo utiliza ambas funciones objetivo: (1) mejor encaje, (2) máximo *fitness*. La etapa de mejora solo es invocada si la solución calculada por el algoritmo constructivo es una solución promisoria, como se indica en la línea 3 del Algoritmo 2. Las soluciones son restringidas a aquellas que son mayores que un límite dinámico. En la primera iteración, este límite toma el valor de la primera solución entregada por el algoritmo constructivo. El procedimiento es detallado en [5].

ALGORITMO 2. FASE DE MEJORA DEL ALGORITMO PROPUESTO.

```
    function GRASP_MovMejora(P, R): List mejorSolucion
1   for i in 0,1,2,...Imax do
2           solución ← GRASP_Constructivo(P, R)
3           if FilteringSolution(solución) then
4                   mejorSolucion←MovMejora (solución)
5           end
6   end
7   return mejorSolucion
    end
```

Criterio de parada**.** El esquema de búsqueda compuesto por la fase de construcción y de mejora se realiza un número máximo de iteraciones *Imax*. Esto se suele calibrar con respecto al número de ejemplares a cortar, en este caso 10 veces el número de tipos de piezas.

III. METODOLOGÍA

Para validar el desempeño del algoritmo propuesto, se aplicó una metodología basada en 4 etapas: (A) selección de instancias, (B) diseño y codificación de algoritmos (optimización y post-optimización), (C) cálculo de soluciones y registro de indicadores de desempeño y (D) validación de algoritmos mediante comparación de desempeño. La validación se realizó sobre cuatro indicadores de desempeño: porcentaje de volumen utilizado (PU), tiempo de cómputo de la solución, tiempo de corte y cantidad de partículas emitidas.

A. Selección de Instancias

Se definieron tres criterios de selección: (1) soluciones reportadas en la literatura para caso irrestricto y no ponderado, (2) tiempos de cómputo de solución reportados en literatura y (3) acceso público para la descarga de instancias. Se identificaron dos grupos de instancias con estas características.

El primer grupo etiquetado como UU fue presentado por la referencia [4]. Se trata de un grupo de 11 instancias generadas de forma aleatoria, con distribución uniforme y parámetros resumidos en la Tabla 2. Las instancias pueden ser descargas del repositorio en [8]. Las mejores soluciones reportadas en la literatura fueron obtenidas mediante un algoritmo de programación dinámica, este logra encontrar soluciones óptimas para problemas de pequeña escala y usa técnicas heurísticas para resolver problemas de gran porte, estas soluciones pueden ser consultadas en la referencia [9].

TABLA II
PARÁMETROS DE INSTANCIAS UU

Parámetro	Descripción	Rango
L, W	Largo L y ancho W de la placa contenedora	[500,4000]
l_i, w_i	longitud l_i y ancho w_i, para $i = 1,2, \dots, m$	[0.01 x L, 0.7 x W]
m	Tipos de piezas a cortar	[25,60]

El segundo grupo de instancias etiquetado como GCUT fue presentado inicialmente por [10] y complementado por [11]. En [10] presentan un grupo de 12 problemas (GCUT1-GCUT12) divididos en tres clases, cada una definiendo una placa contenedora cuadrada con $L \in \{250, 500, 1000\}$. Así mismo, cada clase se compone de cuatro instancias, una por cada número de tipos de pieza m, con $m \in \{10, 20, 30, 50\}$. Para estos problemas, la longitud l_i de cada pieza fue generada a partir del muestreo de un número entero de la distribución uniforme [L /4, 3 L /4], con el ancho w_i de cada pieza siendo generado por el muestreo de un entero de la distribución uniforme [W /4, 3 W /4]. Este grupo es complementado con la instancia mayor GCUT13, derivada de un problema práctico [10]. En [11] adicionan cuatro instancias al grupo (GCUT14-GCUT17) mediante unión de la instancia GCUT13 con las instancias GCUT9, GCUT10, GCUT11 y GCUT12, respectivamente; para las instancias adicionadas se consideró una placa contenedora cuadrada con $L = 3500$. El grupo

completo de instancias puede ser descargado del repositorio en [12], o [13]. Las soluciones óptimas fueron calculadas con técnicas exactas y pueden ser consultadas en la referencia [9].

Las características de las instancias son resumidas en la Tabla 3.

TABLA III
RESUMEN DE CARACTERÍSTICAS DE LAS INSTANCIAS UTILIZADAS

Caso	L	W	m	$\min(l_i w_i)$	ρ
GCUT1	250	250	10	6020	10,4
GCUT2	250	250	20	5610	11,1
GCUT3	250	250	30	5041	12,4
GCUT4	250	250	50	4554	13,7
GCUT5	500	500	10	22968	10,9
GCUT6	500	500	20	24948	10,0
GCUT7	500	500	30	18963	13,2
GCUT8	500	500	50	18864	13,3
GCUT9	1000	1000	10	116963	8,5
GCUT10	1000	1000	20	125580	8,0
GCUT11	1000	1000	30	90706	11,0
GCUT12	1000	1000	50	94336	10,6
UU1	500	500	25	22729	11,0
UU2	750	800	30	36567	16,4
UU3	1100	1000	25	65603	16,8
UU4	1000	1200	38	65043	18,4
UU5	1450	1300	50	78338	24,1
UU6	2050	1457	38	185712	16,1
UU7	1465	2024	50	130287	22,8
UU8	2000	2000	55	244650	16,3
UU9	2500	2460	60	297686	20,7
UU10	3500	3450	55	546780	22,1
GCUT13	3000	3000	32	50924	176,7
GCUT14	3500	3500	42	50924	240,6
GCUT15	3500	3500	52	50924	240,6
GCUT16	3500	3500	62	50924	240,6
GCUT17	3500	3500	82	50924	240,6
UU11	3500	3765	25	46879	281,1

En esta Tabla, las instancias han sido agrupadas en dos categorías: (1) baja densidad y (2) alta densidad. La densidad es calculada como la relación entre el volumen de la placa contenedora y el mínimo volumen de los ítems a cortar como se indica en la Ecuación 1. El valor densidad de referencia que se seleccionó para separar los grupos fue 100.

$$\rho = \frac{LW}{\min(l_i w_i)} \qquad (1)$$

B. Diseño y Codificación de Algoritmos

1) Algoritmo de Optimización

El algoritmo descrito en la sección II-B fue implementado en C++ y se adicionaron rutinas para: visualizar la solución en una imagen (ver Figura 2Fig. 2) y calcular indicadores asociados al desempeño de la solución entregada: porcentaje de área utilizada y tiempo de cómputo de la solución. En su versión actual (v1.0), el software recibe información descriptiva de las piezas a través de un archivo de texto. La solución calculada es almacenada en archivo de texto de salida, con un formato específico. El desarrollo software fue registrado con el nombre de GRASPCutting.

Fig. 2. Patrón de corte para el caso GCUT13. Fuente [10].

2) Algoritmo de Post-Optimización

Este algoritmo busca mejorar el desempeño del proceso de corte mediante la eliminación de trayectorias de corte redundantes. El desempeño es medido en función del tiempo de corte y la cantidad de partículas emitidas durante el proceso. El algoritmo se fundamenta en una búsqueda iterativa de trayectorias de corte que presenten intersecciones y orientación común. Para estas trayectorias, el algoritmo calcula una única trayectoria, como resultado de la unión de las trayectorias individuales. El proceso inicia con la descomposición del patrón de corte calculado en dos listas: verticales y horizontales; cada lista representa un conjunto de líneas rectas. Posteriormente inicia la fase de búsqueda de líneas superpuestas; los casos de superposición para listas horizontales son resumidos en la Fig. 3. Finalmente, se realiza la unión de las líneas superpuestas. De esta forma, el algoritmo calcula el conjunto de rutas mínimo para implementar el patrón de corte calculado. En [14] se presenta una explicación detallada de este algoritmo.

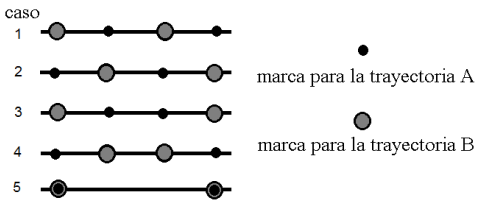

Fig. 3. Casos de intersección entre trayectorias paralelas.

C. Cálculo de Soluciones y Registros de Indicadores de Desempeño

Para calcular las soluciones de las instancias seleccionadas se utilizó una máquina con procesador Intel® Core(TM) i7-4790 CPU 3.60GHz, memoria RAM de 8GB y sistema operativo Windows 7 ® 64-bits. Cabe observar que la herramienta propuesta se ejecuta en un único hilo, para el cual

la CPU tiene un rating de 2286 – mayores números significan mejores CPU's [15]. Los dos primeros indicadores (porcentaje de volumen utilizado, tiempo de cómputo de la solución) fueron registrados para los 24 casos de prueba seleccionados. Los dos últimos indicadores (tiempo de corte y cantidad de partículas emitidas) solo fueron registrados para la instancia GCUT10 utilizando dos métodos de solución: (1) con post-optimización y (2) sin post-optimización. La Fig. 4 presenta la secuencia de procesamiento para estos dos métodos.

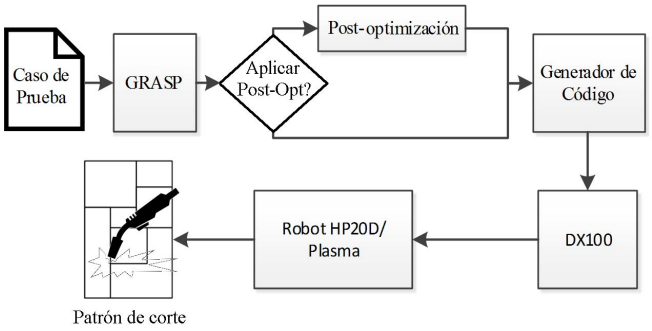

Fig. 4. Métodos de procesamiento utilizados.

1) Porcentaje de Volumen Utilizado y Tiempo de Cómputo

El PU se calcula mediante la Ecuación 2. El tiempo de cómputo se calcula mediante funciones del tipo *tic-toc* programadas en la herramienta software. Donde A es el área total de la placa, a_i es el área de la pieza i ubicada en la placa; i pertenece al subconjunto de piezas empacadas en *solución.*

$$PU(\%) = \frac{\sum_{i \in Solución} a_i}{A} \qquad (2)$$

$$CP\left[\frac{\mu g}{m^3}\right] = \frac{pesoinicial[\mu g] - pesofinal[\mu g]}{tiempo[min] \times \text{caudal}[\frac{m^3}{min}]} \qquad (3)$$

En el método de procesamiento 1 no se aplica post-optimización, diferente al método de procesamiento 2 donde sí se aplica.

2) Tiempo de Corte y Cantidad de Partículas Emitidas

El tiempo de corte se midió utilizando un cronometro digital. El tiempo registrado incluye las operaciones de aseguramiento de la placa a la mesa de corte, ejecución del corte por parte del robot y extracción de piezas cortadas. La medición de partículas se realizó con una bomba de muestreo personal marca Sensidyne [16], una balanza analítica de precisión [17] y filtros del tipo estándar [18]. La Ecuación 3 permite calcular la cantidad de partículas emitidas; se configuró la bomba de muestreo con un caudal de 2 m^3/min.

Para realizar la medición del tiempo y la emisión de partículas se diseñó un experimento de dos tratamientos. En la Tabla IV se presenta la ficha de caracterización del experimento, los tratamientos mencionados hacen referencia al método de procesamiento utilizado (ver Fig. 4). Se tomaron 6 muestras aleatorias del tiempo y de la emisión de partículas para cada uno de los métodos. Se desea saber si las medias de ambos métodos se pueden considerar estadísticamente iguales, para ello se plantean las siguientes hipótesis:

$$H_o : \mu_1 - \mu_2 = 0$$
$$H_A : \mu_1 - \mu_2 \neq 0$$

En donde μ_x representa el valor medio de la variable de respuesta y x representa el método utilizado. Las corridas experimentales se realizaron en orden estrictamente aleatorio para reducir relación directa entre los datos en el primer tratamiento y los datos en el segundo tratamiento.

TABLA IV
FICHA DE CARACTERIZACIÓN DEL EXPERIMENTO

Objetivo	Establecer si existen diferencias entre las variables de respuesta al utilizar los dos niveles del factor seleccionado.
Variable de respuesta	Tiempo Cantidad de partículas
Factores	Método de corte
Unidad experimental	Placa
Nivel del factor	Método 1 Método 2
Tamaño del experimento	6 placas por cada método de corte

La celda de corte se configuró con dos equipos: (1) sistema de corte Cebora Prof 163acc [19] alimentado con presión de 45psi y 40 Amp de corriente, (2) robot industrial HP20D de Motoman y controlador DX100 [20] configurado con velocidad de avance de 66(mm/s) y una distancia entre la boquilla de la antorcha y la placa de 4mm. La placa utilizada para el corte es del tipo acero *cold rolled* calibre 18 con dimensiones 40x40 cm. El banco de trabajo utilizado se presenta en la Fig. 5.

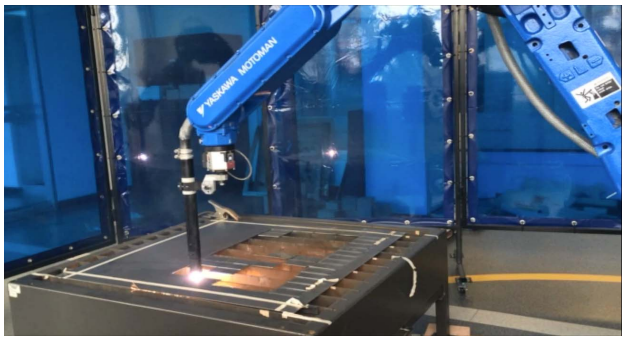

Fig. 5. Banco de trabajo para ejecución de las pruebas de corte. Un video de la prueba se presenta en https://goo.gl/jONLXr.

D. Validación de Algoritmos Mediante Comparación de Desempeño

La validación de calidad y el tiempo de cómputo de la solución se realizó mediante comparación entre las soluciones calculadas por el algoritmo propuesto, versus las mejores soluciones reportadas en la literatura. Los resultados obtenidos se presentan en la Tabla V.

La validación de tiempos de corte y cantidad de partículas emitidas se realizó mediante comparación entre los dos métodos de corte (ver Figura 5) para la instancia GCUT10. Los resultados obtenidos se presentan en las Figuras 6 - 9 y las Tablas 5 y 6.

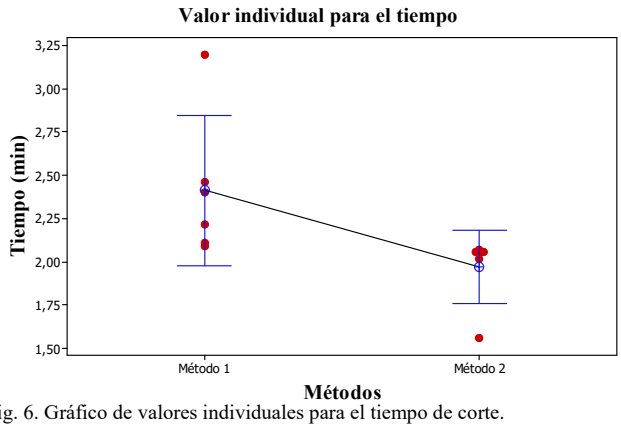

Fig. 6. Gráfico de valores individuales para el tiempo de corte.

TABLA V
COMPARACIÓN DE DESEMPEÑO ENTRE GRASPCUTTING Y LAS MEJORES SOLUCIONES REPORTADAS

Caso	GRASPCutting		Solución en [11]		Solución en [9]	
	PU %	Tiempo (seg)	PU %	Tiempo (seg)	PU %	Tiempo (seg)
GCUT1	90,34	0,5	90,34	0	**93,57**	0,72
GCUT2	95,74	1,4	96,86	0	**97,83**	0,96
GCUT3	96,58	1,9	97,66	0	**98,04**	4,21
GCUT4	**99,32**	2,9	98,72	0	99,07	20,22
GCUT5	98,40	0,5	98,40	0	**98,40**	0,67
GCUT6	94,13	1,1	95,60	0	**97,44**	0,74
GCUT7	**98,35**	1,7	97,03	0	97,72	0,92
GCUT8	98,90	3,0	98,65	0	**99,13**	32,57
GCUT9	97,11	0,6	97,11	0	**97,11**	0,68
GCUT10	98,20	0,9	98,20	0	**98,20**	0,76
GCUT11	97,46	2,0	98,01	0	**98,01**	1,92
GCUT12	98,00	3,1	98,00	0	**98,00**	16,95
GCUT13	99,68	3,2	99,98	22,03	**99,98**	0,72
GCUT14	99,83	4,4	99,96	118,45	**99,96**	46,84
GCUT15	99,84	5,8	99,97	120,86	**99,97**	876,49
GCUT16	99,83	7,2	99,99	161,47	**99,99**	1492,92
GCUT17	99,78	10,2	99,99	212,66	**99,99**	1763,37
UU1	98,04	1,2	–	–	**98,08**	5,48
UU2	98,54	2,2	–	–	**99,21**	34,24
UU3	98,05	1,4	–	–	**98,92**	16,51
UU4	98,76	2,7	–	–	**99,26**	330,16
UU5	**99,47**	3,4	–	–	99,21	3057,23
UU6	98,53	2,1	–	–	**98,79**	61,50
UU7	**99,37**	3,6	–	–	99,28	7620,64
UU8	**98,44**	3,7	–	–	99,27	1822,15
UU9	**99,45**	4,1	–	–	99,20	0,88
UU10	98,82	4,1	–	–	**99,01**	0,70
UU11	99,44	3,0	–	–	**99,85**	244,20
Promedio GCUT	97,73	**2,97**	97,91	37,38	**98,38**	250,69
Promedio UU	98,81	**2,86**	–	–	**99,09**	1199,42
Promedio Total	98,15	**2,92**	97,91	37,38	**98,66**	623,4

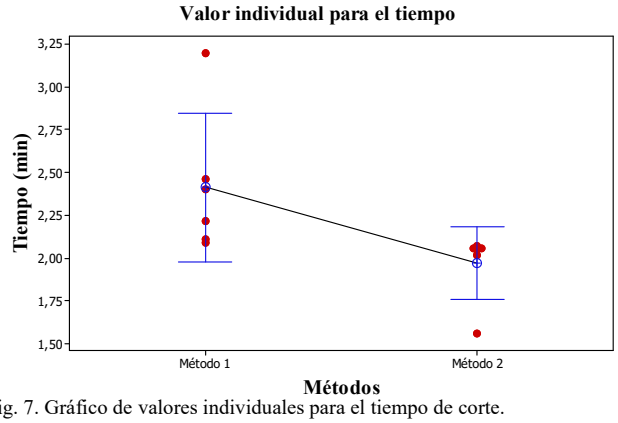

Fig. 7. Gráfico de valores individuales para el tiempo de corte.

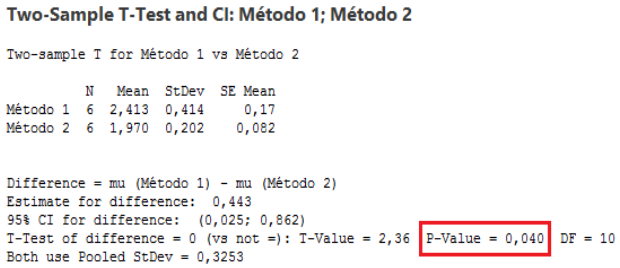

Fig. 8. Valores obtenidos prueba t para tiempo.

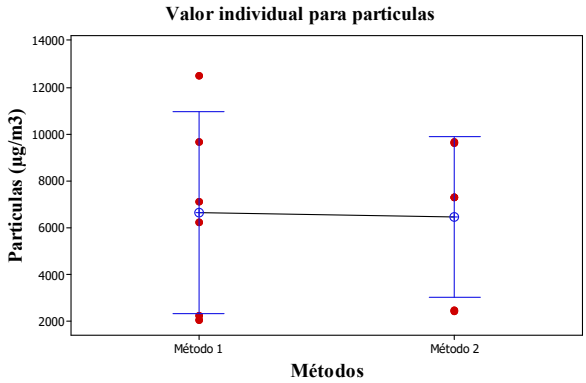

Fig. 9. Gráfico de valores individuales de la emisión de partículas.

TABLA VI
RESULTADOS DE PARTÍCULAS (µG/M3)

No.	Método 1	Método 2
1	4785	2475
2	4878	9662
3	7444	2033
4	6250	7282
5	7109	7282
6	2252	9615

IV. ANÁLISIS DE RESULTADOS

En la Tabla V se presentan los resultados obtenidos al ejecutar el programa GRASPCutting para los casos GCUT y UU. La primera y segunda columna contienen los resultados obtenidos por GRASPCutting; la tercera y cuarta columna presentan los resultados de la solución reportada en [11] y la quinta y sexta presentan los resultados reportados en [9] con el algoritmo L.

Los resultados muestran que el algoritmo GRASPCutting consigue resolver instancias de pequeño y gran porte en tiempos computacionales razonables, conservando una calidad de solución cercana a las mejores publicadas en la literatura. Además de esto, cabe resaltar que el tiempo computacional de las metodologías propuestas por otros autores crece exponencialmente con relación al tamaño del problema (número de piezas demandadas) y la relación de tamaños entre la pieza original y las piezas demandadas o densidad. Lo anterior se puede notar en la Tabla V para los algoritmos propuestos por [11] y [9] donde ocurre un aumento drástico en sus tiempos computacionales para los casos UU, GCUT15-GCUT17, debido a que la relación de tamaño pieza original y piezas demandadas es casi 30 veces.

Se puede inferir entonces que para aplicaciones industriales donde existe un gran número de piezas demandadas y la relación de tamaño no es controlable, el algoritmo propuesto es ideal debido a que su tiempo de respuesta es estable al igual que la calidad de su solución. Es evidente que el algoritmo con mejor calidad publicado en la literatura es el presentado por [9], dado que este fue ejecutado en un CPU con un rating de 2983 [15] no es posible realizar un comparación justa en términos de tiempos computacionales; el rating fue calculado respetando el paralelismo del software CPLEX Optimizer® versión 7.0 utilizado en [9]. Mientras que con respecto a la calidad promedio de solución existe una diferencia del 0.51% del porcentaje de utilización a favor de [9].

En las Figuras 7 a 10 se pueden observar los resultados obtenidos del experimento, estos datos fueron ingresados en el software Minitab® 17.1.0 para su respectivo análisis.

En la Figura 7 se muestra el gráfico de valores individuales para los resultados de tiempo de corte, en este se pueden observar valores atípicos tanto para el método 1 (4.03 y 3.20) como para el método 2 (1.56), sin embargo, el resto de los datos muestran poca dispersión entre ellos; en especial el método 2. La línea de conexión de las medias nos muestra una variación entre los dos métodos permitiendo asociar menor tiempo en el método 2.

El análisis estadístico de hipótesis se hizo en dos etapas: (a) inicialmente se estableció un intervalo de confianza del 95% se obtiene un *valor-p* de 0.141, al ser mayor que 0.05 se acepta la *Ho*, es decir, el experimento tiene varianzas iguales. En la etapa (b) se procede a realizar una prueba *t de Student* a los datos (ver Fig. 8), en esta se obtiene un *valor-p* de 0.040. Como el *valor-p* es menor que el *alfa* se rechaza la *Ho*, es decir, sí existen diferencias entre las medias de los tratamientos, con una confianza del 95%, además se puede observar que la diferencia estimada entre estos es de 0.443min.

TABLA VII
RESULTADOS DE TIEMPO DE CORTE (min)

No.	Método 1	Método 2
1	2.09	2.02
2	2.05	2.07
3	4.03	2.46
4	3.20	2.06
5	2.11	2.06
6	2.22	1.56

La Figura 9 presenta datos asociados a partículas emitidas, se destaca la presencia de valores atípicos y una leve dispersión entre los datos y la línea de conexión de las medias muestra una ligera variación entre los dos métodos obteniendo menor emisión de partículas el método 2. Al ser tan notoria la diferencia se hace la verificación de la varianza y según esta la respectiva validación de las hipótesis. El análisis estadístico de hipótesis se hizo en dos etapas: (a) se estableció un intervalo de confianza del 95% obteniendo un *valor-p* de 0.631, al ser mayor que 0.05 se acepta la H_o, es decir, el experimento tiene varianzas iguales.

En la siguiente etapa (b) se realizó una prueba *t* de *Student* (ver Fig. 10) en la cual se obtiene un *valor-p* de 0.937; valor mayor que el valor de alfa. En consecuencia, se acepta la H_o de medias, por lo tanto, se dice que con un 95% de confianza no hay diferencia entre las medias de los tratamientos para las partículas emitidas.

Two-Sample T-Test and CI: Método 1; Método 2

```
Two-sample T for Método 1 vs Método 2

          N  Mean  StDev  SE Mean
Método 1  6  6634   4107     1677
Método 2  6  6459   3274     1337


Difference = mu (Método 1) - mu (Método 2)
Estimate for difference:  175
95% CI for difference:  (-4602; 4953)
T-Test of difference = 0 (vs not =): T-Value = 0,08  P-Value = 0,937  DF = 10
Both use Pooled StDev = 3713,8761
```

Fig. 10. Valores obtenidos prueba t para partículas.

V. CONCLUSIONES

A partir de los resultados obtenidos para los casos GCUT y los casos UU, se puede concluir que GRASPCutting presenta un buen desempeño en cuanto al porcentaje de utilización de la placa original. GRASPCutting calcula soluciones eficientes en un menor tiempo de cómputo, esto no es totalmente comparable debido a la diferencia de arquitecturas de cómputo y programación, sin embargo, sí se resalta el comportamiento estable de los tiempos del algoritmo, en especial su independencia con las relaciones de tamaño entre pieza original y piezas demandadas.

La etapa de post-optimización propuesta logra mejorar los tiempos de ejecución de la tarea de corte sin aumentar la emisión de partículas, esto se evidencia en el estudio de tiempos y emisiones presentado. Cabe aclarar que el protocolo experimental aquí usado puede ser mejorado mediante el uso de herramientas de medición más sofisticadas como balanzas de mayor precisión. De la misma forma, si se aumenta la cantidad de muestras se podrían obtener valores más precisos.

En este estudio además de resolver un problema complejo de optimización como es la mochila bidimensional, propone una solución eficiente para las aplicaciones de este en la industria metalmecánica, en un contexto actual como es el uso de manipuladores industriales. Como trabajo futuro se propone: considerar el problema de procesar piezas con daños parciales en su superficie, explorar la aplicabilidad de algoritmos exactos para el problema de secuenciamiento de corte.

AGRADECIMIENTOS

Este trabajo ha sido apoyado por la Universidad de La Salle a través del proyecto 24326204, COLCIENCIAS, a través de la convocatoria Jóvenes Investigadores e Innovadores en alianza SENA 2015 y la Universidad de Los Andes a través del proyecto P16.245722.008/01.

REFERENCIAS

[1] L. V. Kantorovich, "Mathematical Methods of Organizing and Planning Production," *Manage. Sci.*, vol. 6, no. March, pp. 366–422, 1960.
[2] P. C. Gilmore and R. E. Gomory, "A Linear Programming Approach to the Cutting Stock Problem - Part II," *Operations Research*, vol. 11. pp. 863–888, 1963.
[3] T. A. Feo and M. G. C. Resende, "A probabilistic heuristic for a computationally difficult set covering problem," *Operations Research Letters*, vol. 8, no. 2. pp. 67–71, 1989.
[4] D. Fayard, M. Hifi, and V. Zissimopoulos, "An efficient approach for large-scale two-dimensional guillotine cutting stock problems," *J. Oper. Res. Soc.*, vol. 49, no. 12, pp. 1270–1277, 1998.
[5] D. Alvarez-Martínez, R. Alvarez-Valdes, and F. Parreño, "A GRASP algorithm for the container loading problem with multi-drop constraints," *Pesqui. Operacional*, vol. 35, no. 1, pp. 1–24, Apr. 2015.
[6] G. A. Camacho-Muñoz, D. Cuellar-Usaquen, and D. Alvarez-Martinez, "Heuristic Approach for the Multiple Bin-Size Bin Packing Problem," *IEEE Lat. Am. Trans.*, vol. 2, p. 7, 2018.
[7] F. Parreño, R. Alvarez-Valdes, J. F. Oliveira, and J. M. Tamarit, "Neighborhood structures for the container loading problem: A VNS implementation," *J. Heuristics*, vol. 16, no. 1, pp. 1–22, 2010.
[8] CERMSEM, "Repositorio de instancias," 2004. .
[9] E. G. Birgin, R. D. Lobato, and R. Morabito, "Generating unconstrained two-dimensional non-guillotine cutting patterns by a Recursive Partitioning Algorithm," *J. Oper. Res. Soc.*, vol. 63, no. 2, pp. 183–200, 2012.
[10] J. E. Beasley, "Algorithms for Unconstrained Two-Dimensional Guillotine Cutting," *J. Oper. Res. Soc.*, vol. 36, no. 4, pp. 297–306, 1985.
[11] G. F. Cintra, F. K. Miyazawa, Y. Wakabayashi, and E. C. Xavier, "Algorithms for two-dimensional cutting stock and strip packing problems using dynamic programming and column generation," *Eur. J. Oper. Res.*, vol. 191, no. 1, pp. 59–83, 2008.
[12] J. E. Beasley, "OR-Library," 1990. .
[13] E. G. Birgin, R. D. Lobato, and R. Morabito, "Recursive partitioning approach for the Distributor's Pallet Loading Problem," 2012. .
[14] G. Camacho, D. A. Martínez, and P. Velasco-Carvajal, "Two Dimensional Knapsack problem using industrial Robots," in *EngOpt 2016 5th. International Conference on Engineering Optimization*, 2016, pp. 558–567.
[15] PassMark®, "CPU Benchmarks," 2016. .
[16] Sensidyne, "Abatement Air Sampling Pump," *Gillian - Industrial Health & Safety Instrumentation*, 2017. .
[17] Ohaus, "Analytical Balance," *Pioneer - Analytical and Precision Balance*, 2017. .
[18] Tqlaboratorios, "Papel filtro-hoja," *TQ - Labaware & Supplies*, 2017. .
[19] Cebora, "Instruction Manual For Plasma Cutter," *Plasma Prof 163 acc*, 2015. .
[20] Yaskawa, "Instructions Manual for HP20D/HP20F," *Motoman Robotics*, 2016.

Paula A. Velasco Carvajal received the Engineering degree in Industrial Engineering from Universidad de La Salle-Bogotá, Colombia, in 2018. Her current research interest includes optimization of process, competitiveness and innovation in enterprise of services. In addition to process improvement using lean six sigma methodologies.

Guillermo Alberto Camacho Muñoz received the B.Sc. degree in Automation Engineering from the University of Cauca, Colombia in 2008 and the M.Sc. degree in Mechatronics Systems from the University of Brasilia, Brazil in 2012. He was Associated Professor with Department of Industrial Engineering, University of La Salle, Bogotá, Colombia, since January 2013 until December 2018. He is currently with Universidad del Valle as PhD student in the program of doctorate in Electrical and Electronic Engineering. His research interest includes industrial Robotics applications, Optimization algorithms applied to Logistic problems, image processing algorithms for detection and feature extraction of objects.

Daniel Cuellar Usaquen received the Engineering degree in Industrial Engineering from Universidad de La Salle- Bogotá, Colombia, in 2017. He is a magister student in Industrial Engineering at the Universidad de Los Andes (Colombia) since 2018. His current research interest includes optimization and simulation for production, transport and logistics problems. In addition to process improvement using lean six sigma methodologies.

David Álvarez-Martínez received a PhD from Universidade Estadual Paulista "Júlio de Mesquita Filho" (Brazil) in 2014. He is a Professor of Industrial Engineering at Universidad de Los Andes (Colombia) since 2016. His current research interest includes optimization, simulation and industrial automation (robotics) for transportation and logistics problems.