Un algoritmo genético para el *strip-packing problem* de dos dimensiones y piezas regulares

Gonzalo Villagrán[1], Gustavo Gatica[1], Juan P. Sepúlveda[2],

Carlos Contreras Bolton[3], Víctor Parada[3]

[1] Escuela de Informática, Universidad Andrés Bello, 237, Av. República, Santiago, Chile
ggatica@unab.cl
[2] Escuela de Industrias, Universidad Andrés Bello, 237, Av. República, Santiago, Chile
[3] Departamento de Ingeniería Informática, Universidad de Santiago de Chile, 3659, Av. Ecuador, Santiago, Chile

Keywords: *Strip Packing Problem*, Algoritmos Genéticos, Metaheurísticas.

Resumen

Dado un conjunto de piezas rectangulares, y un contenedor de ancho fijo y largo infinito, el problema del *Strip Packing* en dos dimensiones, consiste en posicionar ortogonalmente todas las piezas dentro del contenedor, sin solaparlas, permitiendo su rotación en 90º, para así minimizar la altura alcanzada dentro del contenedor. Se han utilizado para su resolución los Algoritmos Genéticos, con representaciones numéricas, sin embargo, requieren de operadores genéticos especializados para preservar la factibilidad de las soluciones. Nosotros proponemos un enfoque binario del tipo genotipo-fenotipo que no altera el proceso evolutivo, éste permite el uso de operadores genéticos tradicionales. Los resultados computacionales se ejecutaron con un *benchmark* de la literatura, siendo prometedores al superar en un 100% los problemas resueltos por el Algoritmo Genético con representación numérica.

1. Introducción.

El problema de Strip Packing en dos dimensiones con rotación en 90º (2SPP-RF), se define como una región rectangular de ancho W definido y largo infinito, se debe disponer de todas las n piezas rectangulares $j \in J = \{1, 2,\ldots, n\}$ con ancho w_j y largo h_j definidos, permitiendo la rotación en 90º. El objetivo es minimizar la altura H obtenida, sin solapar las piezas en el contenedor. El 2SPP-RF se considera como NP-Hard debido a su explosión combinatoria a medida que el problema aumenta [1]. Este problema se presenta en industrias papeleras y madereras [2]. Para la resolución del SPP se han utilizado métodos exactos [3], heurísticos [4–6] y metaheurísticos, basados en solución inicial, tales como Tabu Search [7], [8], GRASP [2] y Simulated Annealing [9].

Un enfoque metaheurístico muy utilizado para la resolución del SPP, son los Algoritmos Genéticos (AG), los cuales simulan la teoría de la evolución [10]. Hopper & Turton [11] proponen un AG (GA+BLF), cuya representación consiste en permutar una cadena de enteros, que describe el orden de ingreso de las piezas a ser posicionadas en el contenedor, utilizando el algoritmo Bottom Left [12]. Su proceso evolutivo, requiere del operador de cruzamiento PMX [13] y una mutación del tipo Order-Based [14]. Sin embargo, debido el tipo de representación, los operadores genéticos dependen directamente de ésta, y se ven forzados a reparar las infactibilidades que surgen en el proceso evolutivo, alterando una normal evolución. Bortfeldt [15] presenta un AG (SPGAL), utilizando una representación compleja, basada en layouts por capas, sin lograr obtener solución óptima para las clases de problemas más pequeños de Hopper y Turton [16]. Otro AG para el SPP (MGA), desarrollado por Mancapa et al. [17], utiliza una representación que también considera piezas irregulares, mediante una cadena con valores numéricos, contiene tres tuplas por cada pieza, los cuales indican la posición, el orden de colocación y la rotación. Sin embargo, para el caso del 2-SPP el error promedio alcanza un 26%, en contraste con el 4,57% obtenido por Hopper & Turton [16]. Matayoshi [18], propone un nuevo método (CJ+EA), basado en un Algoritmo Evolutivo, donde la representación no tiene capacidad de resolver los problemas de mayor tamaño que los propuestos por Hopper & Turton [16], siendo su error promedio para 6 de los 7 grupos de problemas, un 7,47%.

En este artículo, nosotros proponemos la utilización del clásico AG de Golberg [13], con un enfoque del tipo genotipo-fenotipo [19], [20], en el cual el genotipo es una cadena binaria y el fenotipo corresponde a la colocación final de las piezas sobre el contenedor. Nosotros presentamos los resultados numéricos con un grupo de instancias que corresponden a los benchmark de la literatura, mostrando prometedores resultados que 3 AG con operadores ad hoc al problema, y dependientes directamente de la representación.

En la siguiente sección de este artículo se describe la representación binaria, mientras que en la tercera sección se presentan y discuten los resultados computacionales. En la última sección son presentadas las conclusiones del estudio.

2. Materiales y métodos.

En esta sección presentamos el proceso evolutivo, la representación utilizada, el algoritmo de colocación, los parámetros genéticos utilizados, el hardware y software y los problemas que validan el experimento.

Proceso Evolutivo

Nosotros diseñamos e implementamos un AG que opera en una modalidad genotipo-fenotipo. Como genotipo se utiliza una cadena binaria β, la cual especifica el orden de ingreso a la función constructora, ésta se encarga posteriormente de decodificar β y mediante un algoritmo de colocación, ubica las piezas en el contenedor, representando el fenotipo del individuo tal como se esquematiza en la Figura 1.

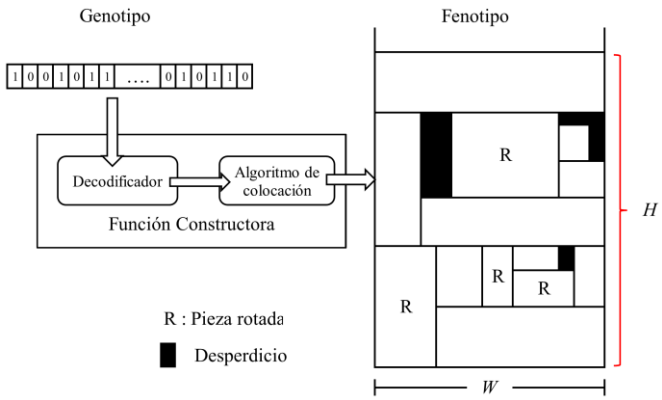

Figura 1: Genotipo-Fenotipo.

La evolución de la población se realiza con los operadores de selección por torneo binario entre dos individuos seleccionados aleatoriamente, con cruzamiento de un punto y mutación binaria [21]. La función objetivo corresponde a la diferencia entre la altura óptima conocida (H*) y H.

Representación

La representación propuesta, consiste en dividir a β en *n* segmentos, donde cada segmento *i* está numerado como {*n*, *n*-1,… ,1} y corresponde a la pieza codificada. Cada segmento se divide en tres partes leídas de izquierda a derecha. La primera parte, de largo de un *bit*, indica que si la pieza tiene valor 1, se rota. La segunda parte, de largo un *bit*, corresponde a la posición referencial, que indica el sentido de la decodificación del genotipo. Si tiene valor 1, se posiciona en el comienzo de *J*, y si tiene valor 0, se posiciona en el fin de *J*. La última parte, compuesta de un conjunto de *bits*, representa el valor de unidades de desplazamiento en *J*, no se contabilizan las piezas ya ingresadas a Θ, en el desplazamiento. La descripción antes descrita se observa en la Figura 2. Donde el tamaño de cada segmento está dado por el número de piezas que restan por identificar, puesto que a medida en que se avanza en la lectura, el número de piezas por identificar se reducen, donde el largo de cada segmento *i* se calcula $\lceil \log_2 i \rceil + 1$ y el largo de β queda expresada por $L = \sum_{i=1}^{n} (\lceil \log_2 i \rceil + 1)$.

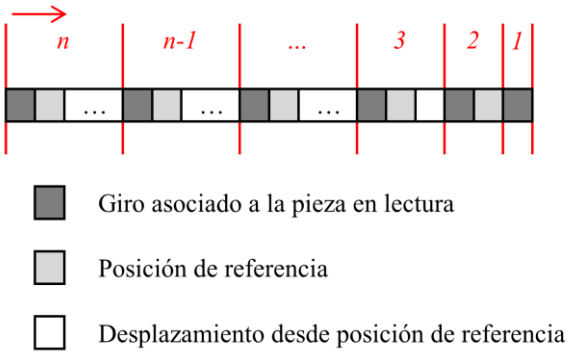

Figura 2: Representación binaria.

En la Figura 3, se ilustra un problema de 5 piezas, dado un β aleatorio, se requiere de L=13 para ser representado. El primer segmento, tiene dimensión 4 bits, no se rota y su posición de referencia es el inicio de J, el desplazamiento es de 3 unidades, entonces se agrega la pieza 4 a Θ y se descarta de J. El segundo segmento, tiene dimensión 3 bits, se rota y su posición de referencia es el fin de J, el desplazamiento es 1 unidad, agregando la pieza 3 a Θ y se descarta de J. El tercer segmento, tiene dimensión 3 bits, la pieza no se rota y su posición de referencia es el inicio de J, el desplazamiento es 0, se agrega la pieza 1 a Θ y se descarta de J. En el cuarto segmento, la pieza no está rotada, la posición de referencia es el fin, y no existe desplazamiento, se agrega a Θ la pieza 5, y en último lugar, la pieza 2 no está rotada y es la última en ingresar a J.

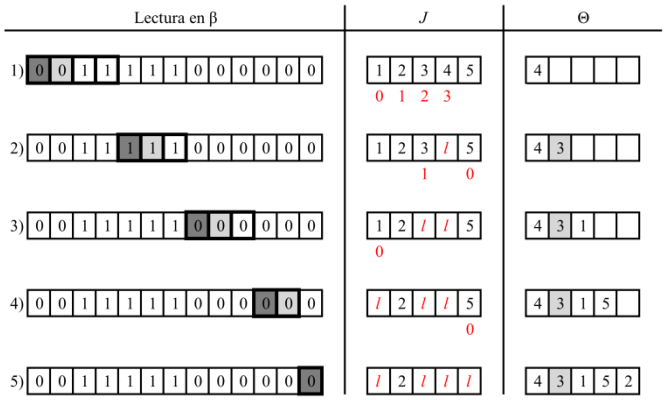

Figura 3: Decodificación de una cadena binaria para un problema de 5 piezas.

Algoritmo de colocación

El algoritmo corresponde a una implementación del Best-fit propuesta por Burke, Kendall, & Whitwell [22], donde la entrada de las piezas corresponde a Θ. No se consideró en la implementación la lógica que modifica la colocación inicial de cada pieza, para no alterar el proceso evolutivo.

Parámetros genéticos

Para la obtención de los parámetros genéticos, se realizó una sintonización fina de la probabilidad de mutación y probabilidad de cruzamiento, estableciéndose en 5% y 85% respectivamente; luego de experimentos preliminares se establece utilizar 500 generaciones y 500 individuos como tamaño de población. Cada AG se ejecutó 10 veces para cada problema.

Hardware y software

Los experimentos se realizaron en un computador Intel Core 2 Duo CPU E 7500 de 2,93 GHz, con 4 GB de memoria RAM, con la distribución GNU/Linux Ubuntu 12.0.4 LTS, el AG fue desarrollado en lenguaje de programación C.

Problemas utilizados

En este artículo se utiliza un benchmark para el SPP, que corresponde a Hopper & Turton [16], que consiste a un conjunto 7 clases de problemas, que van desde 16 a 197 piezas.

3. Resultados y discusión.

A medida que avanza el proceso evolutivo, se van generando individuos cada vez mejores, hasta llegar a las etapas finales en las cuales hay una convergencia hacia poblaciones que tienen individuos de calidad diversa, pero con la mayoría de ellos teniendo un valor de fitness cercano al mejor valor de la población. La Figura 5 muestra un caso típico del proceso evolutivo, con el problema C7-P2 de Hopper & Turton [16]. En el eje de las ordenadas se presenta el valor de fitness y en el eje de las abscisas se indica el número de generación. En este caso, la población generada aleatoriamente tenía valores de fitness entre 26 a 91 aproximadamente, al cabo de 160 generaciones el valor de fitness del mejor individuo es cercano al mejor valor obtenido de la evolución; a partir de la generación 175, el valor promedio y el valor del mejor individuo están próximos entre sí. El mejor individuo obtenido para esta corrida se encontró en la generación 162. El peor individuo durante el proceso evolutivo disminuyó, estabilizándose en un valor de fitness entre 27 a 44.

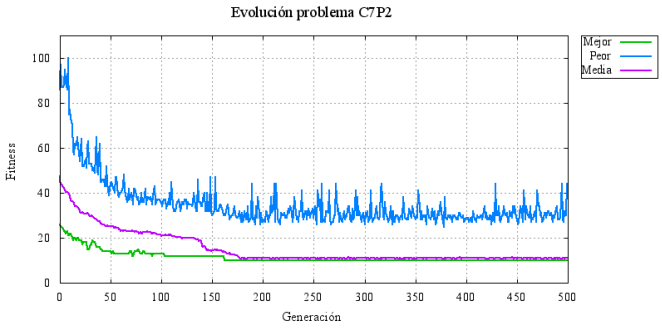

Figura 5: Decodificación cadena binaria.

En la Tabla 1, se presentan los resultados obtenidos con los problemas de Hopper & Turton [16]. La primera columna compuesta, corresponde al problema utilizado, las siguientes dos columnas, a la solución óptima conocida, y el número de piezas del problema, las siguientes columnas corresponden a los resultados obtenidos por GA+BLF, SPGAL, MGA, CJ+EA, y la heurística BF_{BCC} [6]. La última columna, RB presenta el mejor valor obtenido por nuestro AG. El símbolo "-", indica que no se registra en detalle los resultados computaciones por cada problema, y solo se ha presentado el valor promedio del grupo según la clase. Los resultados numéricos proporcionan un 2,02% de error relativo promedio. El algoritmo propuesto obtiene resultados con menor error relativo promedio que los AG que dependen directamente de la representación. GA+BLF presenta un 4,57% de error relativo promedio, MGA un 26,26%, CJ+EA un 7,47% y la heurística BF_{BCC}, un 4,19%. SPGAL obtiene un 0,64% de error promedio, dado su compleja y especializada representación, que utiliza adicionalmente un proceso constructivo, que repara las soluciones.

4. Conclusiones.

En este artículo se presenta una representación del tipo genotipo-fenotipo que permite la utilización de un AG para la resolución del 2SPP-RF. Los resultados encontrados son superiores a un AG cuyos operadores dependen directamente de la representación utilizada, preservándose así el proceso evolutivo. Como trabajo futuro se pretende utilizar otra función de colocación para así mejorar las calidades de las soluciones.

5. Referencias.

[1] M. R. Garey and D. S. Johnson, *Computers and intractability: a guide to the theory of NP-completeness*. New York: W. H. Freeman & Co., 1979.

[2] R. Alvarez-Valdés, F. Parreño, and J. M. Tamarit, "Reactive GRASP for the strip-packing problem," *Computer Operation Research*, vol. 35, no. 4, pp. 1065–1083, 2008.

[3] R. Alvarez-Valdés, F. Parreño, and J. Tamarit, "A branch and bound algorithm for the strip packing problem," *OR Spectrum*, vol. 31, no. 2, pp. 431–459, 2009.

[4] E. K. Burke, M. R. Hyde, and G. Kendall, "A squeaky wheel optimisation methodology for two-dimensional strip packing," *Computers & Operations Research*, vol. 38, no. 7, pp. 1035–1044, 2011.

[5] D. Chen, Y. Fu, M. Shang, and W. Huang, "A Quasi-Human Heuristic Algorithm for the 2D Rectangular Strip Packing Problem," in *Information Science and Engineering, 2008. ISISE '08. International Symposium on*, 2008, vol. 2, pp. 392 –396.

[6] V. M. Kotov and D. Cao, "A heuristic algorithm for the non-oriented 2D rectangular strip packing problem," *Buletinul Academiei de Stiinte a republicii moldova. matemátia*, vol. 2, pp. 81–88, 2011.

[7] R. Alvarez-Valdés, F. Parreño, and J. M. Tamarit, “A tabu search algorithm for a two-dimensional non-guillotine cutting problem,” *European Journal of Operational Research*, vol. 183, no. 3, pp. 1167–1182, 2007.

[8] G. Gómez-Villouta, J.-P. Hamiez, and J.-K. Hao, “Tabu search with consistent neighbourhood for strip packing,” in *Proceedings of the 23rd international conference on Industrial engineering and other applications of applied intelligent systems - Volume Part I*, Berlin, Heidelberg, 2010, pp. 1–10.

[9] T. Dereli and G. Sena Daş, “A Hybrid simulated-annealing algorithm for two-dimensional strip packing problem,” in *Adaptive and Natural Computing Algorithms*, vol. 4431, B. Beliczynski, A. Dzielinski, M. Iwanowski, and B. Ribeiro, Eds. Berlin: Springer Berlin Heidelberg, 2007, pp. 508–516.

[10] E.-G. Talbi, *Metaheuristics: from design to implementation*, vol. 10. Hoboken: John Wiley & Sons Inc., 2009.

[11] E. Hopper and B. Turton, “A genetic algorithm for a 2D industrial packing problem,” *Computers and Industrial Engineering*, vol. 37, no. 1–2, pp. 375–378, 1999.

[12] S. Jakobs, “On genetic algorithms for the packing of polygons,” *European Journal of Operational Research*, vol. 88, no. 1, pp. 165–181, 1996.

[13] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston: Addison-Wesley Longman Publishing, 1989.

[14] G. Syswerda, “Schedule optimisation using genetic algorithms,” in *Handbook of Genetic Algorithms*, New York: Van Nostrand Reinhold, 1991, pp. 332–349.

[15] A. Bortfeldt, “A genetic algorithm for the two-dimensional strip packing problem with rectangular pieces,” *European Journal of Operational Research*, vol. 172, no. 3, pp. 814–837, 2006.

[16] E. Hopper and B. Turton, “An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem,” *European Journal of Operational Research*, vol. 128, no. 1, pp. 34–57, 2001.

[17] V. Mancapa, T. I. Van Niekerk, and T. Hua, “A genetic algorithm for two dimensional strip packing problems,” *South African Journal of Industrial Engineering*, vol. 20, no. 2, pp. 145 – 162, 2009.

[18] M. Matayoshi, “The 2D strip packing problem: A new approach with verification by EA,” in *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on, Istanbul, 10-13 Octubre 2010*, Istanbul, 2010, pp. 2492 –2499.

[19] E. Falkenauer, *Genetic Algorithms and Grouping Problems*. New York, NY, USA: John Wiley & Sons, Inc., 1998.

[20] F. Rothlauf, *Representations for Genetic And Evolutionary Algorithms*. Netherlands: Springer, 2006.

[21] M. Affenzeller, S. Winkler, S. Wagner, and A. Beham, *Genetic Algorithms and Genetic Programming: Modern Concepts and Practical Applications*. New York: Chapman & Hall/CRC, 2009.

[22] E. K. Burke, G. Kendall, and G. Whitwell, “A new placement heuristic for the orthogonal stock-cutting problem,” *Operations Research*, vol. 52, no. 4, pp. 655–671, Jul. 2004.

Instancia		H^*	n	GA+BLF	SPGAL		MGA	CJ+EA	BF_{BCC}	RB
				H_{min}	H_{prom}	H_{min}	H_{min}	H_{min}	H_{min}	H_{min}
C1	P1	20	16	-	-	-	22	21	21	**20**
	P2	20	17	-	-	-	23	21	21	**20**
	P3	20	16	-	-	-	23	**20**	22	**20**
	e_{prom} C1			4,00	1,70	1,70	13,33	3,33	6,67	0,00
C2	P1	15	25	-	-	-	19	16	16	**15**
	P2	15	25	-	-	-	19	16	16	**15**
	P3	15	25	-	-	-	19	16	16	**15**
	e_{prom} C2			7,00	0,90	**0,00**	26,67	6,67	6,67	0,00
C3	P1	30	28	-	-	-	36	32	32	31
	P2	30	29	-	-	-	34	32	32	**31**
	P3	30	28	-	-	-	36	32	32	31
	e_{prom} C3			5,00	2,20	2,20	17,78	6,67	6,67	3,33
C4	P1	60	49	-	-	-	70	65	63	62
	P2	60	49	-	-	-	72	65	62	62
	P3	60	49	-	-	-	75	64	62	**61**
	e_{prom} C4			3,00	**1,40**	**0,00**	20,56	7,78	3,89	2,78
C5	P1	90	73	-	-	-	117	98	92	92
	P2	90	73	-	-	-	124	99	93	92
	P3	90	73	-	-	-	109	98	**91**	92
	e_{prom} C5			4,00	**0,00**	**0,00**	29,63	9,26	2,22	2,22
C6	P1	120	97	-	-	-	159	134	122	123
	P2	120	97	-	-	-	160	133	122	123
	P3	120	97	-	-	-	160	133	122	123
	e_{prom} C6			4,00	0,7	0,3	33,06	11,11	1,67	2,50
C7	P1	240	196	-	-	-	330	-	**244**	248
	P2	240	197	-	-	-	346	-	243	248
	P3	240	196	-	-	-	352	-	244	248
	e_{prom} C7			5,00	**0,5**	**0,3**	42,78	-	1,53	3,33

Tabla 1: Resultados numéricos con Hopper & Turton [16]