Strip Packaging Problem

Margarita Maguire y Franco N. Merenda

18 de septiembre de 2020

Resumen

En el presente trabajo se analizará y visualizará el problema de optimización de Empaquetado en Tiras (Strip Packaging Problem) para dos de sus versiones. Éste problema de optimización se trata de alinear un conjunto de rectángulos no superpuestos de manera tal de minimizar el material de desecho. "Strip"hace referencia de la tira donde se ubicaran estos últimos. La primer versión es ubicar los rectángulos de acuerdo a sus respectivas coordenadas, no pueden ser rotados, el ancho de la tira (Strip) tiene un máximo y el alto de la misma es infinito. En su segunda versión, mantendrá las mismas restricciones salvo que en ésta los rectángulos se pueden rotar a 90 grados.

1. Introducción

El problema del Empaquetado en Tiras, si bien una de sus principales aplicaciones es disminuir los desechos de material, apareció además para modelar la planificación de trabajos en un tiempo dado. A continuación se dará una descripción formal del problema para las dos versiones a analizar y se mencionarán otros parámetros que se podrían variar para el mismo. Luego se presentarán los algoritmos utilizados para ambas versiones y el estado de arte. Se analizarán los resultados para algunas instancias del problema, luego se discutirán y finalmente de dará una conclusión del trabajo presentado.

2. Descripción del Problema

El problema del empaquetado de tiras es un problema de minimización geométrica bidimensional. Dado un conjunto de rectángulos alineados con el eje y una franja de ancho limitado y altura infinita, se determina un empaque sin superposición de los rectángulos en la franja minimizando su altura.

Algunas de las variantes que también se han estudiado de éste es variar la geometría de los elementos, la dimensión del problema, o el caso de si los artículos rotan o no.

Formalmente, dada una instancia $I = (I, W)$ del problema de embalaje en tiras, que consiste en un strip de ancho $W = 1$ y una altura inifinita así

como un set I de artículos rectangulares. Cada artículo $i \in I$ tiene un ancho $\precsim_i \in (0,1] \cap Q$ y una altura $\eqsim_i \in (0,1] \cap Q$. El empaquetado es un mapeo que se realiza desde la esquina inferior izquierda de un artículo $i \in I$ a una posición $(x_i, y_i) \in ([0, 1 - w_i] \cap Q) \times Q_{\geq 0}$ dentro del strip. Un punto interno de un artículo ubicado $i \in I$ es un punto del conjunto $inn(i) = \{(x,y) \in Q \times Q | x_i < x < x_i + w_i, y_i < y < y_i + h_i\}$. Dos artículos ya ubicados se superponen si comparten algún punto interno. La altura del empaquetado es definida como $max\{y_i + h_i | i \in I\}$. El objetivo es encontrar un empaquetado de los artículos que minimize la altura, en una manera que no se superpongan entre ellos.

Este artículo se centra en resolver el problema con rotación de los rectángulos y sin ella, corroborando que al rotar los rectángulos, las soluciones esperadas deberían ser mejores que sin rotarlos.

3. Propuesta algorítmica

Para resolver el problema propuesto se utilizó un enfoque de algoritmos genéticos. Para el caso del empaquetamiento sin rotaciones, el cromosoma de los individuos fue representado por medio de arreglos, donde cada posición del arreglo representa un gen, los cuales almacenan el rectángulo a ser ubicado en el strip. Estos rectángulos son ubicados en el orden que aparecen en el cromosoma. Un ejemplo de ello, en la tabla 3.1 donde se observa en cada uno de los genes del individuo el rectángulo asociado. Primero ubicará el rectángulo 3, luego el 2 y así siguiendo.

Gen	1	2	3	4	5	6
Individuo	3	2	1	5	4	6

Tabla 3.1 - Representación del individuo

Vale aclarar que tanto el ancho y altura de cada rectángulo están representados de manera análoga. En la tabla 3.2 siguiente se muestra cada rectángulo asociado a sus alturas y anchos, por ejemplo, el rectángulo $rect_1$, tiene 10 unidades de ancho y 5 unidades de altura.

$rect_i$	1	2	3	4	5	6
w_i	10	5	3	9	13	2
h_i	5	3	2	5	8	1

Tabla 3.2 - Representación de los datos asociados

Para el caso del empaquetamiento con rotaciones, para agregar la información de como se encontraban orientados los rectángulos, (Rotado o No Rotado), se agregó un segundo arreglo asociado a cada individuo, obteniendo así una matriz de dos filas (una para el orden de rectángulos y una para la orientación de los mismos) y n columnas (donde n es la cantidad de rectángulos). Cada gen de ésta nueva fila almacena si el rectángulo que se encuentra en la misma columna de la matriz está rotado o no. En la tabla 3.3 a continuación se observa

lo indicado. El rectángulo 3 no se encuentra rotado, como el 2, 5 y 6, pero el rectángulo 1 y 4 si se encuentran rotados.

Individuo	3	2	1	5	4	6
Rotado	NO	NO	SI	NO	SI	NO

Tabla 3.3 - Representación del individuo - Algoritmo con rotación

Para la solución sin rotaciones, se genera la población inicial, donde cada individuo tiene la longitud igual a la cantidad de rectángulos con las que trabajará el algoritmo. Cada individuo se inicializa con una permutación de los rectángulos. Para el caso de la solución con rotaciones, estos individuos son representados por una matriz, donde en la primera fila se genera una permutación de los rectángulos, y en la segunda fila se inicializan valores al azar de rotaciones, controlando que el ancho no sobrepase el ancho del W del strip.

La función de fitness evalúa la altura ubicando los rectángulos del individuo, en el orden que éste los trae, dentro del strip de ancho limitado, donde el ancho máximo está dado por la instancia a ser ejecutada. Al ser un problema de minimización, mientras menor sea la altura mejor será el individuo.

En cuanto a la selección, se realiza por torneo, eligiendo dos individuos al azar, seleccionando el individuo con mejor fitness. Ambos padres son seleccionados mediante éste método.

Para la reproducción, se realiza con el operador PMX (Partially Mapped Crossover), en las referencias se puede encontrar links para poder tener un mejor entendimiento de este operador. Lo que nos permite este operador es entrecruzar los genes de los padres asegurándonos de que se va a mantener la permutación en el descendiente (que no va a haber rectángulos repetidos).

En la mutación, simplemente se intercambian dos genes al azar en el individuo. Esta operación se realiza para todo individuo en la nueva población (si es que ocurre). Finalmente, la poblacion actual es reemplazada por la nueva población, donde algunos individuos van a ser descendientes de otros viejos individuos, y otros se van a mantener tal y como estaban.

Todos estos operadores son iguales en la solución que se plantea con rotaciones, simplemente se diferencia en su implementación debido a la distinta representación y particularmente, en la función de fitness, considerar que los rectángulos pueden estar rotados.

4. Resultados

Para poder procesar las instancias en ambos algoritmos se utilizó el hardware descrito a continuación, donde la implementación del algoritmo se realizó en lenguaje Python, en su versión 3.8.3.

- CPU: Intel I7-7700HQ - 4 Núcleos @ 2.8 - 3.8 GHz con Hyperthreading
- Memoria Ram: 16GB - DDR4

- Sistema Operativo : Linux PopOS 20.04 (Basado en ubuntu)
- Versión Python: 3.8.3

El tiempo total de ejecución de las 6 instancias en ambos algoritmos, con 20 ejecuciones cada una, donde finalmente se muestran las figuras de los mejores individuos, fue de 602.6 segundos.

Los párametros utilizados fueron para ambos algoritmos son los siguientes:

- Probabilidad de mutación = 0.5 %
- Probabilidad de Crossover (Reproducción) = 65 %
- Tamaño de la población = 50 Individuos
- Número de generaciones = 1000

Se probaron 6 instancias distintas del problema en ambos algoritmos, con 20 ejecuciones de cada uno. Los datos de cada instancia están en la tabla 4.1. Los algoritmos tienen, además, ciertos parámetros que están dados por la instancia a ser ejecutada:

- Ancho máximo del Strip (W)
- Dimensiones de los rectángulos (Ancho y Alto)

Instancias															**Ancho máx.**
SPP9a	Ancho	4	2	6	3	4	2	8	3	10					13
	Altura	15	3	6	12	6	8	2	5	3					
SPP9b	Ancho	4	2	6	3	3	2	8	3	10					13
	Altura	15	3	6	12	8	8	2	5	3					
SPP910	Ancho	5	9	2	3	4	6	4	9	3	2				20
	Altura	11	8	8	6	9	8	7	8	11	11				
SPP11	Ancho	4	7	10	2	6	3	1	4	4	6	4			20
	Altura	13	8	5	8	7	8	12	7	4	12	8			
SPP12	Ancho	3	5	4	10	7	6	8	4	18	2	3	9		20
	Altura	6	8	5	8	4	3	8	5	3	3	10	2		
SPP13	Ancho	7	1	5	9	2	9	5	5	4	3	2	9	2	20
	Altura	5	8	9	3	16	7	9	3	9	7	7	3	16	

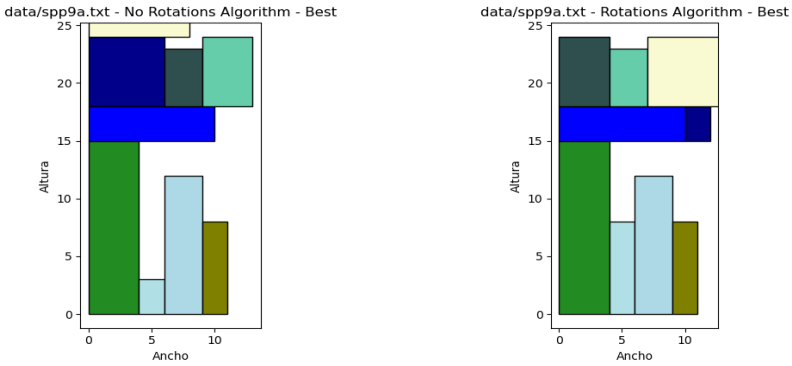
Tabla 4.1 - Instancias del problema

4.1. Resultados SPP9a

Los resultados obtenidos fueron:

	Mejor Fitness	**Mejor individuo**	**Mediana**	**Desviación est.**
AGnr	26	[0, 1, 3, 5, 8, 2, 7, 4, 6]	26	0
AGr	24	[[0, 5, 3, 6, 8, 1, 4, 7, 2] [0, 0, 0, 1, 0, 0, 0, 0, 1]]	26	1,01365

Tabla 4.2 - Resultados spp9a



(a) spp9a sin rotación

(b) sppa9a con rotación

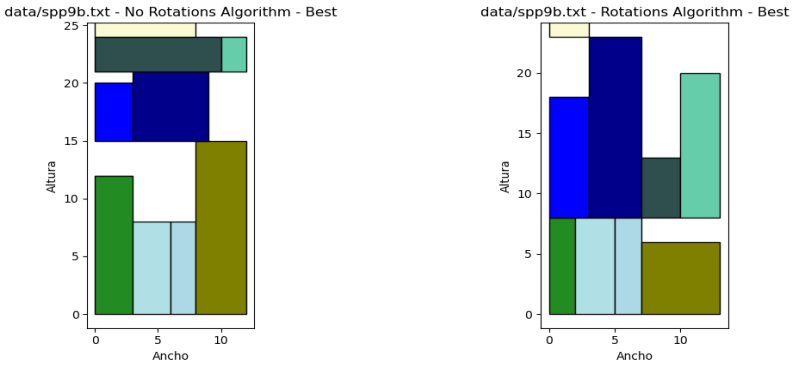
Figura 1: Distribución de los rectángulos en el strip

4.2. Resultados SPP9b

Los resultados obtenidos fueron:

	Mejor Fitness	**Mejor individuo**	**Mediana**	**Desviación est.**
AGnr	26	[3, 4, 5, 0, 7, 2, 8, 1, 6]	26	0
AGr	25	[[6, 4, 5, 2, 8, 0, 7, 3, 1] [1, 0, 0, 1, 1, 0, 0, 0, 1]]	26	0,43301

Tabla 4.3 - Resultados spp9b



(a) spp9b sin rotación (b) sppa9b con rotación

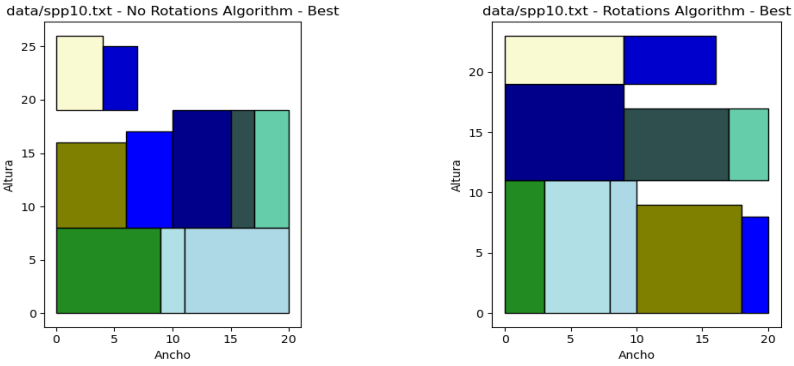
Figura 2: Distribución de los rectángulos en el strip

4.3. Resultados SPP10

Los resultados obtenidos fueron:

	Mejor Fitness	**Mejor individuo**	**Mediana**	**Desviación est.**
AGnr	26	[7, 2, 1, 5, 4, 0, 9, 8, 6, 3]	26	0,21794
AGr	23	[[8, 0, 9, 7, 2, 1, 5, 3, 4, 6] [0, 0, 0, 1, 0, 0, 1, 0, 1, 1]]	24	1,1

Tabla 4.4 - Resultados spp10



(a) spp10 sin rotación (b) sppa10 con rotación

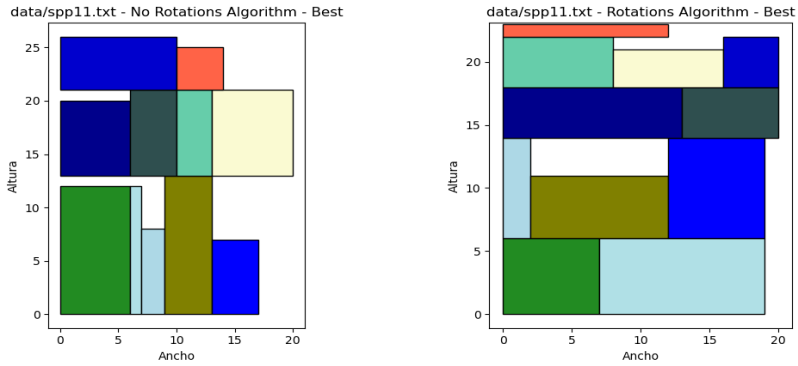
Figura 3: Distribución de los rectángulos en el strip

4.4. Resultados SPP11

Los resultados obtenidos fueron:

	Mejor Fitness	**Mejor individuo**	**Mediana**	**Desviación est.**
AGnr	26	[9, 6, 3, 0, 7, 4, 10, 5, 1, 2, 8]	26	0,43588
AGr	23	[[4, 0, 3, 2, 1, 0, 7, 10, 5, 8, 6] [1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1]]	24	1,32193

Tabla 4.5 - Resultados spp11



(a) spp11 sin rotación (b) sppa11 con rotación

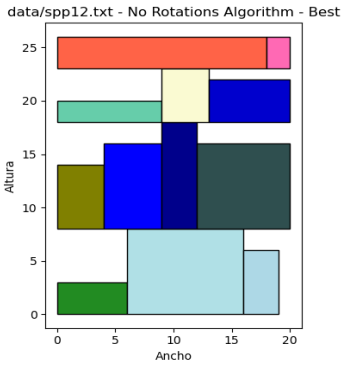
Figura 4: Distribución de los rectángulos en el strip

4.5. Resultados SPP12

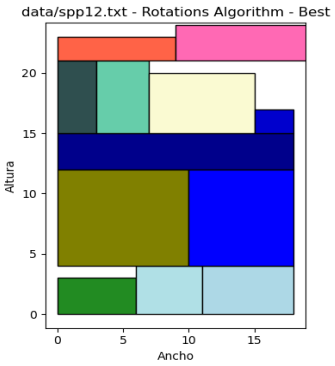
Los resultados obtenidos fueron:

	Mejor Fitness	**Mejor individuo**	**Mediana**	**Desviación est.**
AGnr	26	[5, 3, 0, 7, 1, 10, 6, 11, 2, 4, 8, 9]	26	0,45825
AGr	24	[[0, 2, 4, 3, 6, 8, 5, 7, 1, 9, 11, 10] [1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1]]	25	0,6

Tabla 4.6 - Resultados spp12



(a) spp12 sin rotación



(b) sppa12 con rotación

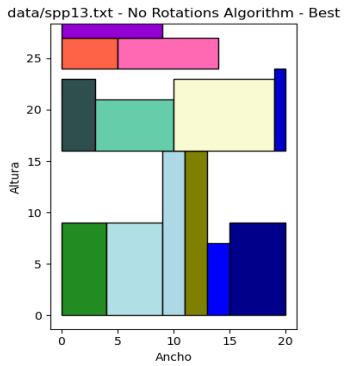
Figura 5: Distribución de los rectángulos en el strip

4.6. Resultados SPP13

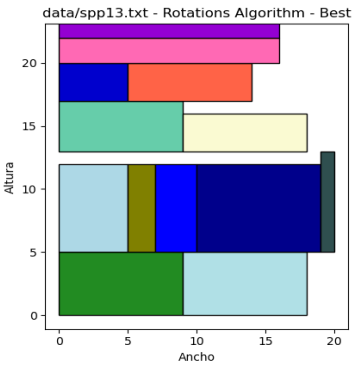
Los resultados obtenidos fueron:

	Mejor Fitness	**Mejor individuo**	**Mediana**	**Desviación est.**
AGnr	30	[8, 6, 4, 12, 10, 2, 9, 0, 5, 1, 7, 11, 3]	31	0,45825
AGr	24	[[6, 2, 0, 10, 9, 5, 1, 8, 11, 7, 3, 4, 12] [1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1]]	25	0,90967

Tabla 4.7 - Resultados spp13



(a) spp13 sin rotación



(b) sppa13 con rotación

Figura 6: Distribución de los rectángulos en el strip

5. Discusión

Como se observa en las secciones de la 4.1 a la 4.6, los resultados confirman que al permitir rotaciones, se puede conseguir mejores resultados, debido a que permite más opciones a como ubicar los artículos. Además reafirma lo que la intución podría predecir en cuanto a permitir o no rotaciones en este tipo de problemas.

Si se observa en cada uno de los resultados de la tablas de dichas secciones, se ve como el algoritmo con rotaciones, siempre consigue un mejor individuo, para todas las instancias planteadas del problema.

Si bien en el artículo se habla de unidades y no de alguna unidad de medida en particular, si se extrapola a situaciones de la realidad, donde las unidades fueran centímetros, o metros, podríamos estar hablando de una reducción de desechos muy significativa, lo cual impactaría directamente en la economía de quien esté intentando solventar este problema. Las implicaciones de este tipo de soluciones pueden afectar de manera simple y efectiva como se ha visto en los resultados.

Los costos computacionales para tener una primera aproximación, como se puede observar, no son tan costosos comparado al problema que se intenta resolver. Inclusive se podría con un mejor hardware, intentar con mayor cantidad de generaciones para comprobar y explorar un mayor número de posibles soluciones.

6. Conclusiones

Para concluir el trabajo, se puede afirmar que a través de este tipo de soluciones, se pueden lograr resultados con bajos costos computacionales de un gran impacto. Hay que tener en cuenta que si bien es una simplificación de un problema real, las implicaciones ya mencionadas son significativas.

Además ambos algoritmos tienen la particularidad de ser modulares por lo que se podría en un futuro probar con diferentes tipos de crossover, otros métodos de selección y ver que resultados se pueden conseguir, para poder lograr relacionarlos con este artículo. También probarlo con diferentes probabilidades de crossover y mutación, para ver como afectan a los individuos finales obtenidos.

Para futuros trabajos se va agregar una interfaz gráfica para permitir la reproducción de los experimentos con mayor facilidad y permitir que la parametrización sea de una manera intuitiva y práctica. A su vez, generar un mayor número de experimentos con las mismas instancias de problemas pero diferentes parametrizaciones en cuanto a la probabilidad de mutación, reproducción, tamaño de la población y el número máximo de generaciones, para poder, a partir de todos hacer una inferencia de que sucede en cuanto a las soluciones, que variabilidad se encuentran en los resultados y que implicancias tiene cada párametro al momento de aumentarlo o disminuirlo.

Finalmente, una posibilidad es implementar ambos algoritmos aprovechando el paralelismo, haciendo uso de tecnologías de hoy en auge, para acelerar el

procesamiento de la población.

Agradecimientos

Un agradecimiento especial a Lila MM por permitir concluir el trabajo.

Referencias

[El-Ghazali Talbi, 2009] Metaheuristics - From Design to Implementation. Wiley 2009, ISBN 978-0-470-27858-1, pp. I-XXIX, 1-593

[Chuan-Kang Tinga, Chien-Hao Sub, Chung-Nan Leeb, 2009] Multi-parent extension of partially mapped crossover for combinatorial optimization problems - https://www.cs.ccu.edu.tw/ ckting/pubs/eswa2010.pdf

[Gokturk Uçoluk 2001] Genetic Algorithm Solution of the TSPAvoiding Special Crossover and Mutation - http://user.ceng.metu.edu.tr/ ucoluk/research/publications/tspnew.pdf