1. Introducción

Un problema que se puede presentar, es encontrar mínimos locales y si esto ocurre, se deben de utilizar estrategias para escapar de ellos. Para este trabajo se utilizó la metaheurística de recocido simulado aplicada al problema de Bin Packing en una dimensión, que está basada en un fenómeno físico, y cuyo comportamiento es: explorar mucho en la fase inicial, y después transformar el espacio de búsqueda con un parámetro T. Esta modificación de recocido simulado se llama recocido de pesos, ya que el problema de Bin Packing se basa en guardar elementos con ciertos pesos dentro de contenedores, y su objetivo es minimizar la cantidad de contenedores utilizados. Se aplicó la técnica mostrada en [1], con una modificación que se presentará adelante, los resultados no fueron los esperados, ya que no se logró escapar de los mínimos locales, pese al número de iteraciones y de cambios en los parámetros utilizados.

2. Metodología

La idea que se utiliza es cambiar el objetivo del problema. El objetivo, en lugar de minimizar la cantidad de contenedores, se convierte en maximizar la suma de elementos dentro de cada contenedor. Así, para cada iteración, se utilizará una distorsión que se aplicará a todos los elementos, y esta depende del residuo del contenedor en el que se encuentren.

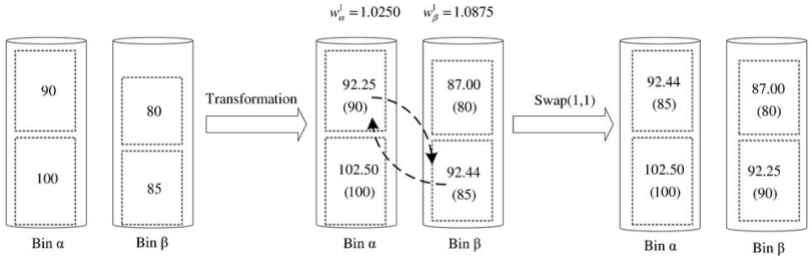

Figura 1: Imagen tomada del artículo mencionado. Detalles de los cálculos se explican en el texto.

Los pasos que sigue este algoritmo son:

1. Empezar con una solución utilizando una heurística constructiva (en este caso se utilizó First-Fit-Decreasing). Esta heurística se basa en llenar los contenedores con los elementos ordenados de mayor a menor, y en cada paso busca el primer contenedor que tenga espacio suficiente para el elemento va a insertar.

2. Determinar un nuevo conjunto de pesos, basados en los residuos de la solución anterior. Se utilizó la fórmula:
$$w_i^T = (1 - Kr_i)^T, \tag{1}$$
donde T es la variable temperatura (que se utiliza en la idea de recocido simulado), K es una constante y $r_i = residuo_i/c$, donde c es la capacidad máxima de cada contenedor, y el residuo es la resta entre la c y la suma de los elementos en el contenedor i.

3. Vaciar contenedores. Para hacer la vecindad, se busca insertar un elemento del contenedor α en el contenedor β, se comparan contenedores de par en par. Esto ocurre si hay espacio libre en el contenedor, y si al insertar no se viola la condición de que cada contenedor tiene una capacidad máxima. Además, se utiliza la condición $\Delta f(1,0) \geq 0$, donde
$$\Delta f(1,0) = (residuo_\alpha - elemento_{\alpha,i})^2 + (residuo_\beta + elemento_{\alpha,i})^2 - residuo_\alpha^2 - residuo_\beta^2. \tag{2}$$
Los autores utilizan esta condición basándose en el artículo [2], así, para poder ingresar elementos en el contenedor β, se deben de cumplir ambas condiciones.

4. Hacer un intercambio entre elementos de los contenedores. Este fue uno de los cambios con respecto al artículo [1]. Esta parte del algoritmo, busca intercambiar elementos entre el contenedor α y β (como se muestra en la Fig. 1). Sin embargo, en el artículo, se busca intercambiar elementos (o al menos eso se entendió) en orden. Y en este caso, se cambiaron elementos utilizando un elemento aleatorio del contenedor α y otro elemento aleatorio del contenedor β. Estos se cambian si no se violaba la capacidad máxima del contenedor, y si se cumplía la condición $\Delta f(1,1) \geq 0$,

$$\begin{aligned}\Delta f(1,1) = (residuo_{\alpha} - elemento_{\alpha,i} + elemento_{\beta,j})^2 + (residuo_{\beta} - elemento_{\beta,j} + elemento_{\alpha,i})^2 \\ - residuo_{\alpha}^2 - residuo_{\beta}^2.\end{aligned} \tag{3}$$

5. Finalemente, si alguno de los contenedores pasaba a no tener ningún elemento, se eliminaba y se volvía a hacer el procedimiento.

El último cambio con respecto al algoritmo del artículo, es que este utiliza cuatro cambios en el algoritmo, y en este trabajo sólo se utilizaron dos, el punto 3 y el punto 4.

3. Resultados

No se logró escapar de los mínimos locales, si se logró vaciar algunos contenedores, sin embargo, ninguno se vació por completo. Por lo tanto, la solución obtenida siguió siendo la misma que la heurística constructiva. Se utilizaron tres instancias con distinta cantidad de elementos y con distintos parámetros. Para la instancia con 120 elementos, con K = 0.005 y T = 1, con un cambio a cada iteración de T = T*0.95, se muestran algunos cambios:

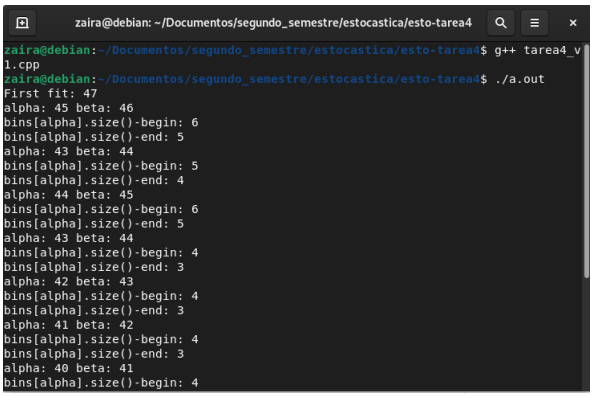

Figura 2: Algunos cambios en los contenedores con la instancia de 120 elementos y capacidad máxima de cada contenedor c = 150.

Para la instancia con 500 elementos, se utilizaron los mismos parámetros y se encontró:

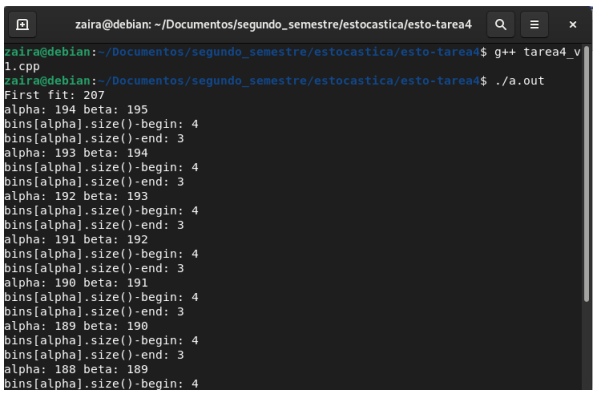

Figura 3: Instancia con 500 elementos, y capacidad máxima c = 150.

Estas instancias se probaron con distintos parámetros de K y T, los utilizados en los resultados son los mismos que el artículo. También se aumentó el número de iteraciones, y aún así, lo máximo que se llegó fue a obtener contenedores con dos elementos, no se logró obtener contenedores que al vaciarse llegaran a un elemento o al esperado, a cero elementos. Los resultados de la instancia que no se muestran, fue porque se dejó con un

millón de iteraciones, y el proceso se detuvo, aunque esta instancia tiene menos elementos que la instancia 3, la instancia que lo detuvo tiene 250 elementos.

4. Conclusiones

Aunque en clase se vio la técnica de recocido simulado, una de las dificultades fue pasar esta idea al problema de Bin Packing. Además de encontrar artículos cuyo objetivo inicial fuera el mismo, ya que se encontraron artículos que cambiaban el objetivo a distribuir de forma equitativa los elementos con la idea de recocido simulado, en estos, no había una capacidad máxima, esta era la que se buscaba. Otra de las dificultades fue entender el término de distorsión de pesos, y que el objetivo tenía que cambiar a maximizar la suma de los pesos. Se decidió cambiar el paso 4 porque se pensaba que haciendo los cambios de forma aleatoria, podía mejorar los resultados, ya que no había ningún parámetro aleatorio en toda la metaheurística. Los resultados obtenidos no fueron satisfactorios, aunque se aumentó la cantidad de iteraciones, y se cambiaban los parámetros para ver mejoras.

Referencias

[1] Kok-Hua Loh, Bruce Golden, and Edward Wasil. Solving the one-dimensional bin packing problem with a weight annealing heuristic. *Computers & Operations Research*, 35(7):2283–2291, 2008.

[2] Krzysztof Fleszar and Khalil S Hindi. New heuristics for one-dimensional bin-packing. *Computers & operations research*, 29(7):821–839, 2002.