**Universidad Peruana de Ciencias Aplicadas**

**UPC**

“Año de la lucha contra la corrupción e impunidad"”



Complejidad Algorítmica

Problema de Empaquetamiento en 3 Dimensiones

Carrera: Ciencias de la Computación

Profesor: Luis Martin Canaval Sanchez

Alumno: Joaquin Ignacio Flores Palao

Código de Alumno: U201810807

Sección: CC-41

Noviembre de 2019

Índice

Advertencias de uso

Introducción

Objetivo del Estudiante

Estado del Arte: De los algoritmos revisados para el trabajo

Aporte

Diseño de Aplicativo para Pruebas

Muestra tablas comparativas del consumo de recursos de memoria y tiempo.

Trabajos Futuros

Conclusiones

Anexos

Bibliografía

**Advertencias de uso**

* Abrir los tres archivos .py en un mismo nivel (carpeta)
* Todos los módulos usados no necesitan descarga previa
* Cambiar la dirección de destino de los archivos ‘in’ & ‘out’

**Introducción**

Las empresas dedicadas a la logística comúnmente afrontan situaciones de empaquetamiento en 3 dimensiones. Al momento de llenar contenedores, camiones, barco y/o aviones de carga, se desea hacer la menor cantidad de viajes posibles por ende deben asegurarse de enviar la mayor cantidad de elementos en cada transporte que realizan.

El problema, mencionado anteriormente, aparece en la literatura bajo diversos nombres (e.g.: problema de embalaje de contenedores o tiras, problema de carga de contenedores, problemas de anidamiento, problema de la mochila, etc.). Para el problema mencionado anteriormente, existen algoritmos que representan soluciones exactas y otros que representan soluciones que hacen uso de heurísticas.

Las empresas, que afrontan dichas situaciones, deben decidir cómo prefieren atacar el problema, y que tipo de solución es más apropiada. Para esto, deben contratar o consultar personal especializado en procesos, algoritmos y complejidad algorítmica para analizar y discutir las ventajas y desventajas que conlleva cada algoritmo, y decidir qué solución es más apropiada.

**Objetivo del Estudiante**

Aplicar los conocimientos adquiridos de complejidad algorítmica, así como de enfoques y paradigmas de programación, tomando conciencia de la importancia de los algoritmos en la industria como tecnología más que como rutinas simples de procesos.

* Reconocer las responsabilidades profesionales y tomar decisiones informadas sobre prácticas de computación basadas en principios legales y éticos.

**Estado del Arte**

En el problema de empaquetamiento, o bin packing, elementos de diferentes volúmenes deben ser acomodados dentro de un numero finito de contenedores, cada uno con un determinado volumen, de modo que se utilicen la menor cantidad de contenedores. En la teoría de la complejidad computacional, es un problema combinatorio NP-hard. El problema de decisión (decidir si los artículos encajarán en un número específico de contenedores) es NP-complete.

Existen distintas variaciones de este problema, como el empaquetado en 2D, empaquetamiento por pesos, por costo, etc. Tienen muchas aplicaciones, como llenar contenedores, cargar camiones con limitaciones de capacidad de peso, crear copias de seguridad de archivos en medios y mapeo de tecnología en diseño de chip de semiconductor de matriz de puerta programable en campo.

El problema del embalaje del depósito también puede verse como un caso especial del problema del material de corte. Cuando el número de contenedores se limita a 1 y cada elemento se caracteriza por un volumen y un valor, el problema de maximizar el valor de los elementos que pueden caber en el contenedor se conoce como el problema de la mochila.

A pesar del hecho de que el problema de empaquetamiento del contenedor tiene una complejidad computacional NP-difícil, se pueden producir soluciones óptimas para casos muy grandes del problema con algoritmos sofisticados. Además, se han desarrollado muchas heurísticas: por ejemplo, el algoritmo de primer ajuste proporciona una solución rápida, pero a menudo no óptima, que implica colocar cada elemento en el primer contenedor en el que encajará. Requiere tiempo Θ (n log n), donde n es el número de artículos que se deben empacar. El algoritmo se puede hacer mucho más efectivo al ordenar primero la lista de elementos en orden decreciente (a veces conocido como el algoritmo decreciente de primer ajuste), aunque esto todavía no garantiza una solución óptima, y ​​para listas más largas puede aumentar el tiempo de ejecución de El algoritmo. Sin embargo, se sabe que siempre existe al menos un pedido de artículos que permite que el primer ajuste produzca una solución óptima.

Dicho problema ha sido “resuelto” utilizando diversos estudios heurísticos, que luego fueron llevados a programas consiguiendo un resultado aproximado con una solución bastante buena. Entre estas soluciones se encuentran los algoritmos de ‘Next-Fit Decreasing Height’ (NFDH), ‘First-Fit Decreasing Height’ (FFDH) y ‘Best-Fit Decreasing Height’ (BFDH); que son algoritmos que no proporcionan el mejor resultado, pero son fáciles de comprender y estudiar; además, aplicando algunas restricciones más ganan precisión, pero pierden velocidad de procesamiento.

**Aporte**

Crear un algoritmo enfocado a un determinado problema y que este lo resuelva no es el final, una vez se obtiene un resultado deseable se debe minimizar el costo que requiere para brindar una solución. Observemos el caso de un arreglo y un heap, ambos pueden ser representados de la misma manera, datos almacenados de forma lineal, pero cuando queremos insertar un valor en un arreglo tenemos que desplazar los n elementos ( n entero positivo) para que el nuevo elemento sea ingresado

**Diseño de Aplicativo para Pruebas**

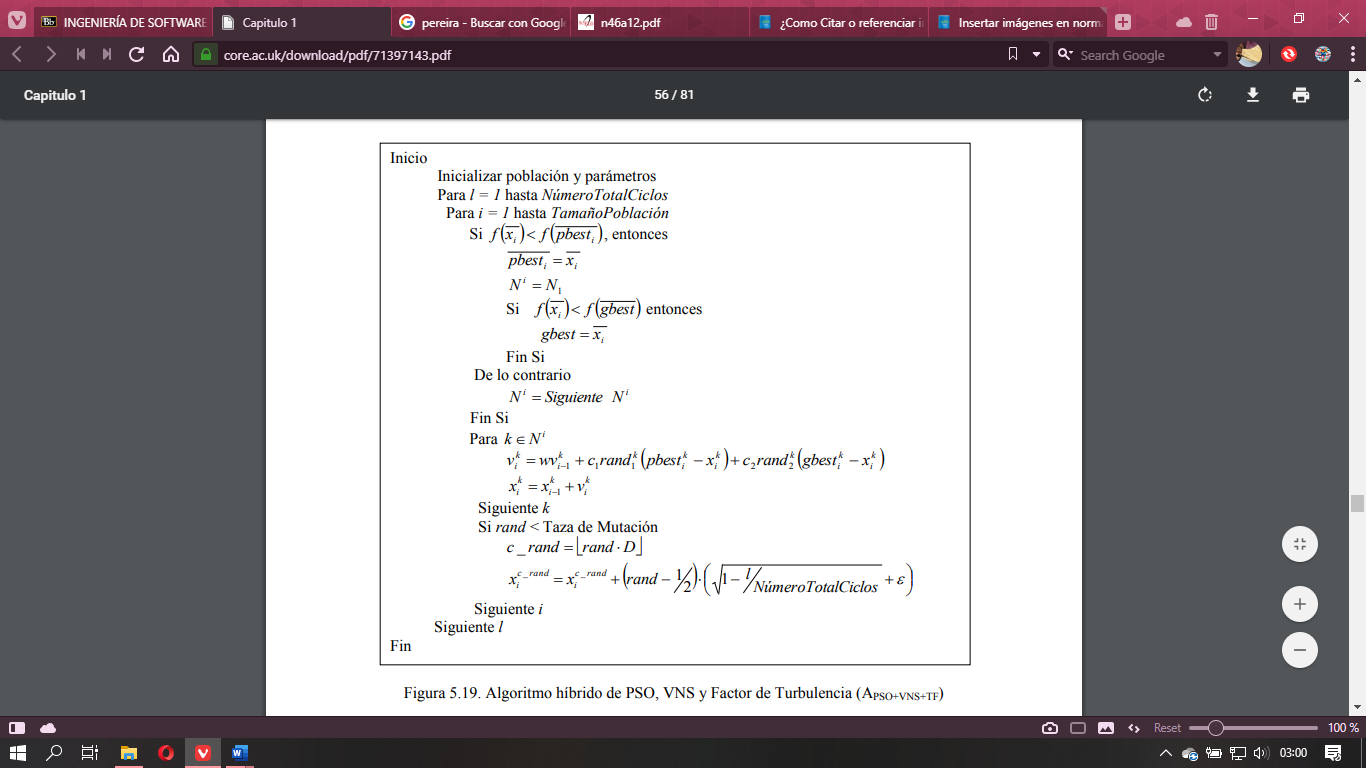


Figura 1: Escobar, L. (2012) Recuperado de: <http://www.scielo.org.co/pdf/rfiua/n46/n46a12.pdf>

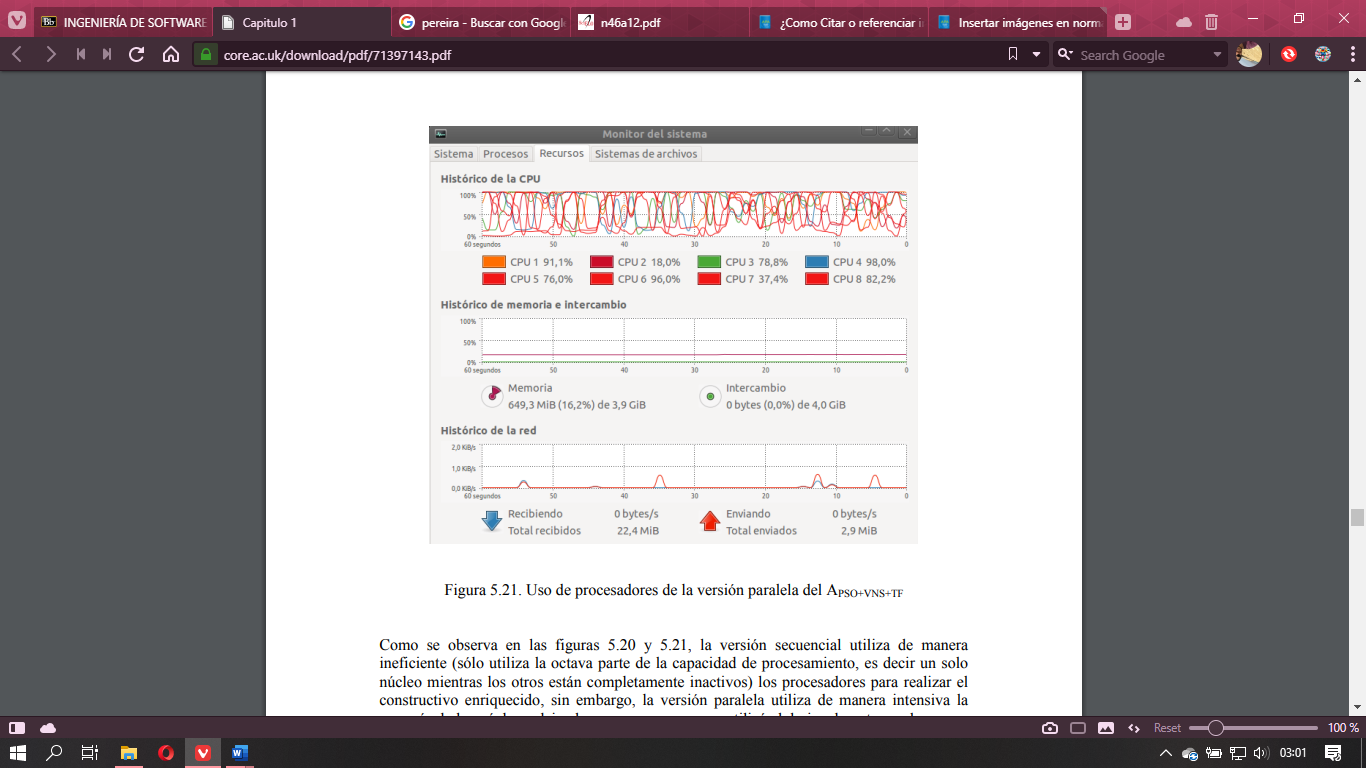
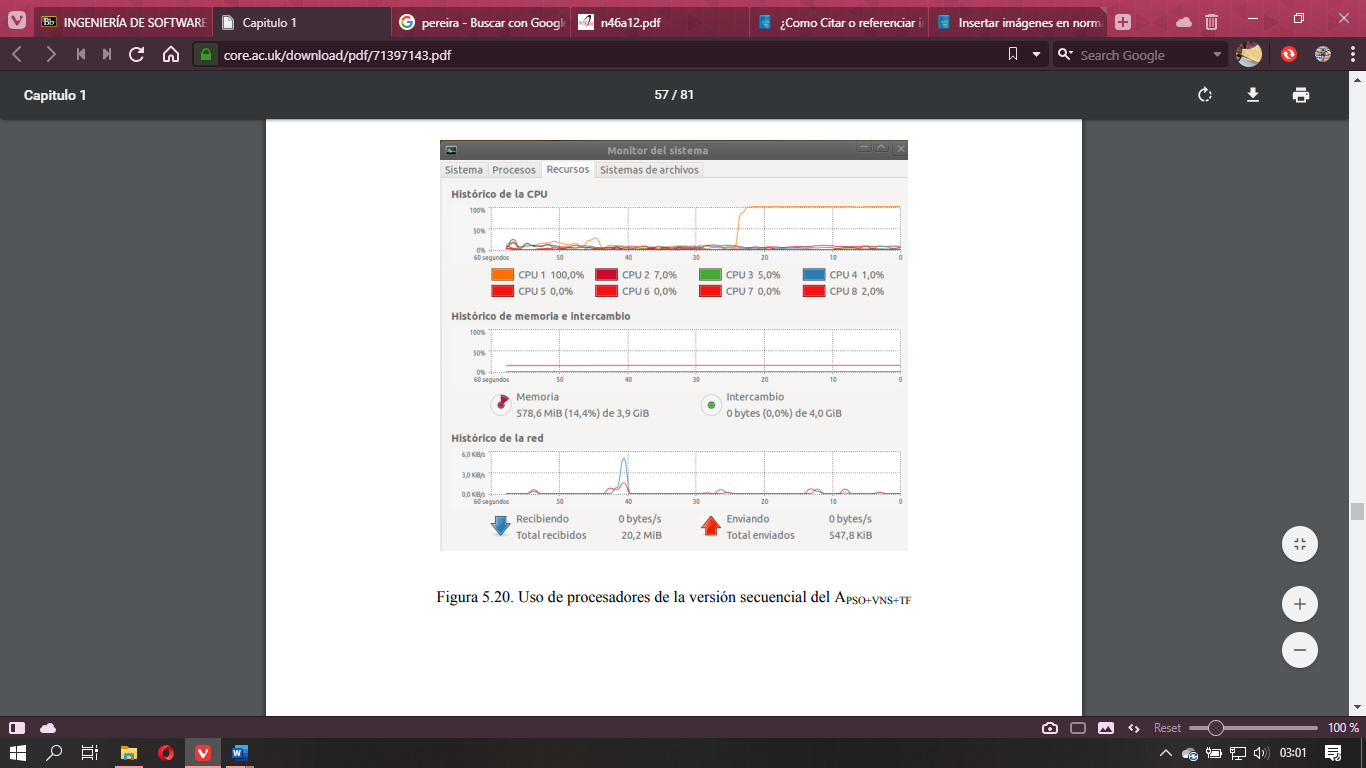


Figura 2 y 3: Escobar, L. (2012) Recuperado de: <http://www.scielo.org.co/pdf/rfiua/n46/n46a12.pdf>

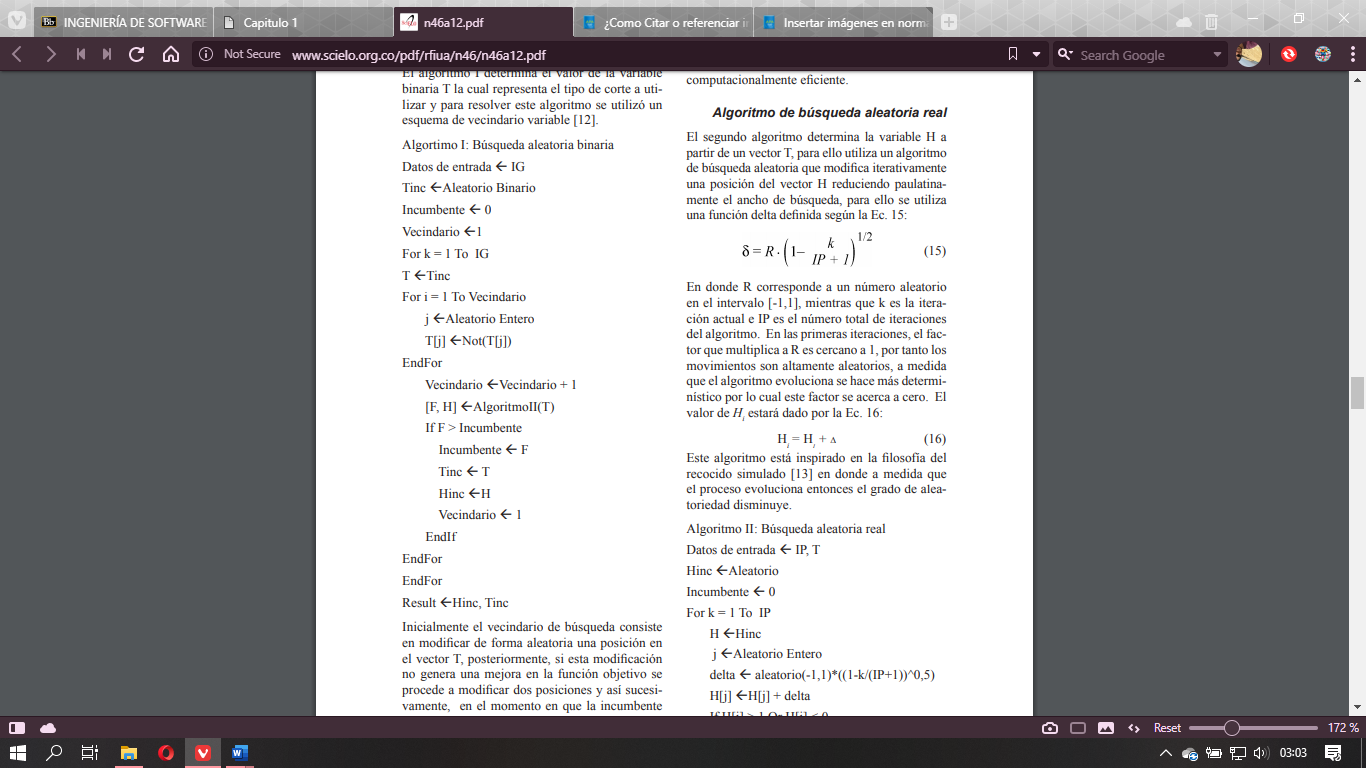


Figura 4: Mirledy, E. (2008) Recuperado de: <https://core.ac.uk/download/pdf/71397143.pdf>

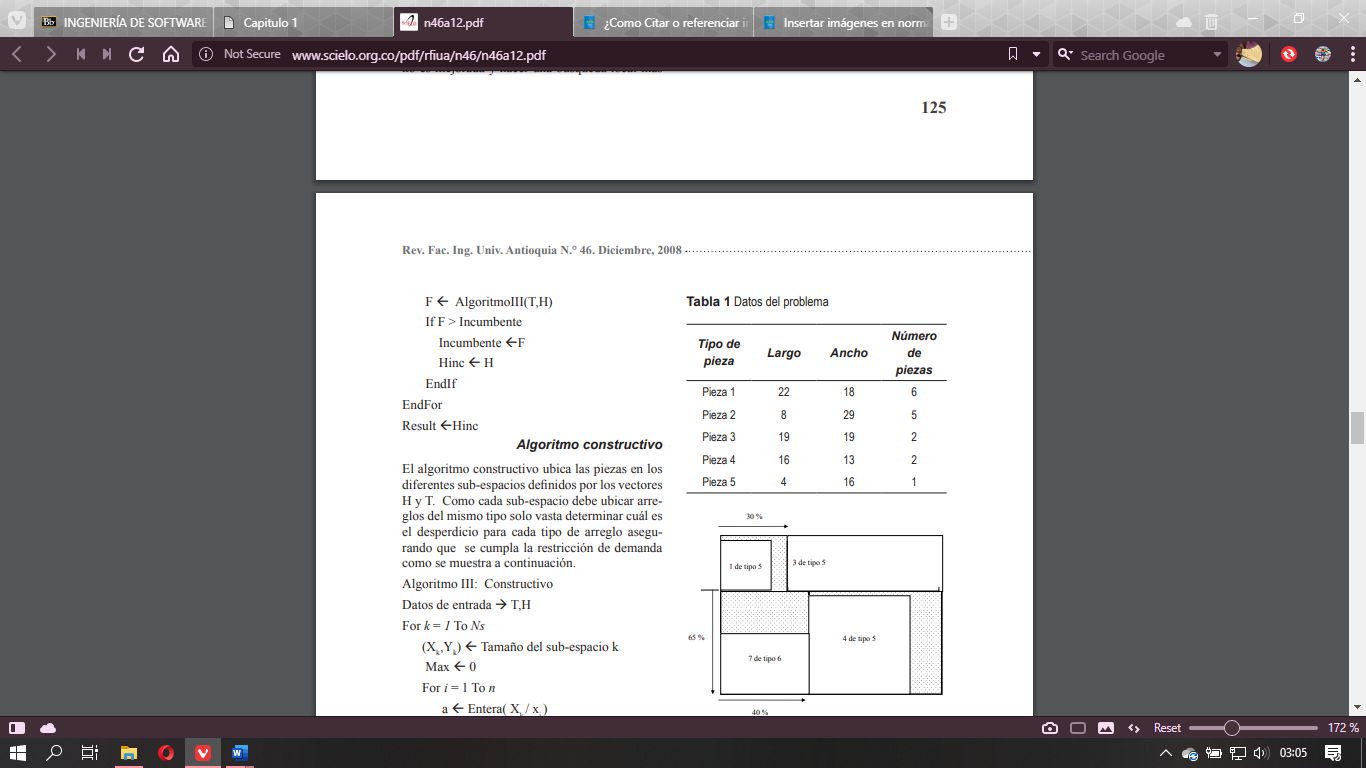
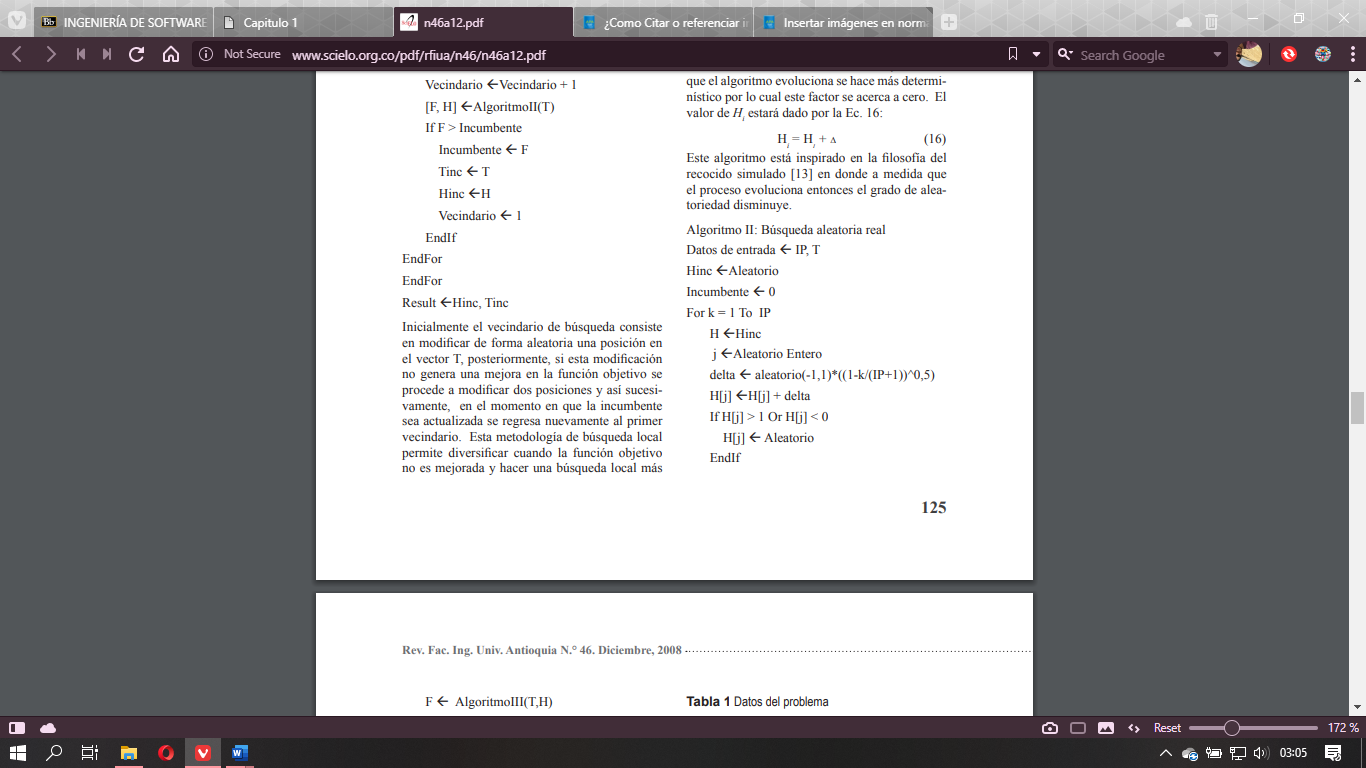


Figura 5: Mirledy, E. (2008) Recuperado de: <https://core.ac.uk/download/pdf/71397143.pdf>

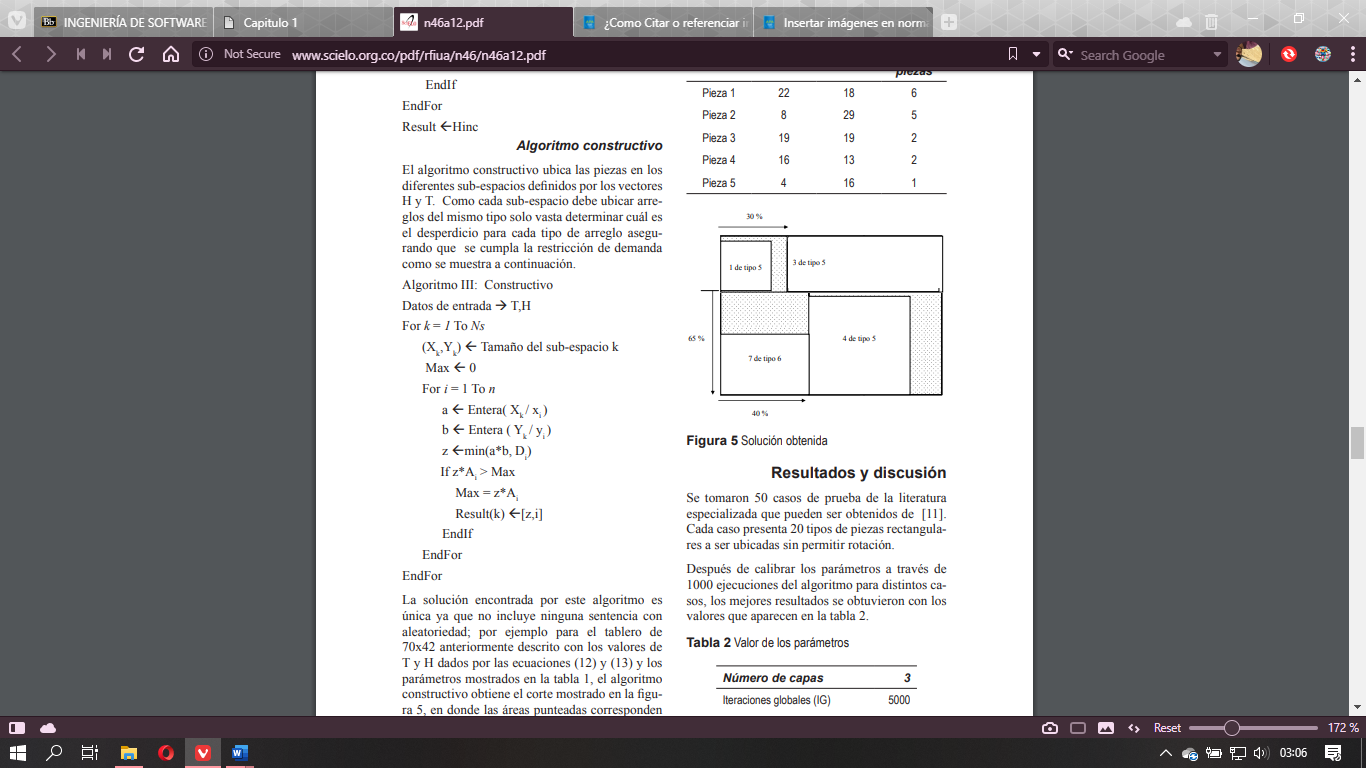


Figura 6: Mirledy, E. (2008) Recuperado de: <https://core.ac.uk/download/pdf/71397143.pdf>

**Trabajos Futuro**

Tomando en cuenta que el promedio de desperdicio resultante de las pruebas hechas oscila entre el 40% y el 55% podemos afirmar que el algoritmo es ‘no recomendable para uso industrial’, dado que, suponiendo que se hace uso de este maximizar el volumen útil de un contenedor no se maximizaría nada; suponiendo que se utiliza para medir el tiempo/energía que se utiliza para ubicar las cajas, caemos en el mismo resultado, no se maximiza el uso de tiempo/energía.

Una proyección propia es sentenciar significativamente mejor las restricciones de intersección para que el algoritmo pueda brindar un resultado más favorable al ahorro.

**Conclusiones**

* La solución brindada para la problemática abarcada podría responder de una mejor manera si se planteara mejor el escenario base y cómo se va modificando de un estado a otro.
* La buena técnica para programar no sirve de nada sin un estudio previo y consciente del problema a abarcar.

**Bibliografía**

Martello, S., Lodi, A. (2001). *Recent advances on two-dimensional bin packing problems.* Italia, Bologna: University of Bologna. Recuperado de: <https://paginas.fe.up.pt/~mac/ensino/docs/OR/HowToSolveIt/RecentAdvancesonTwo-DimensionalBinPackingProblems.pdf> [28 de octubre de 2019]

Mirledy, E., Garcés, A., Ruiz, H. (2008). *Solución al problema de empaquetamiento bidimensional usando un algoritmo híbrido constructivo de búsqueda en vecindad variable y recocido simulado.* Colombia, Pereira: Universidad Tecnológica de Pereira. Recuperado de: <http://www.scielo.org.co/pdf/rfiua/n46/n46a12.pdf> [5 de noviembre de 2019]

Escobar, L. (2012). *SOLUCIÓN DEL PROBLEMA DE EMPAQUETAMIENTO ÓPTIMO USANDO TÉCNICAS METAHEURÍSTICAS DE OPTIMIZACIÓN SIMULTÁNEAS A TRAVÉS DE PROCESAMIENTO PARALELO.* Colombia, Pereira: Universidad Tecnológica de Pereira. Recuperado de: <https://core.ac.uk/download/pdf/71397143.pdf> [5 de noviembre de 2019]