Trabajo Parcial
Complejidad de Algoritmos - UPC
2019-2

1. Descripción del trabajo

Algunas industrias, se dedican a recortar figuras en planchas o pliegues de determinado tipo de material; sin embargo, dicha tarea usualmente implica generación de desperdicio de material y/o uso abusivo de maquinaria. En ambos casos, mencionados anteriormente, existe el problema del desperdicio de recursos.

El problema de desperdicio de material puede ser minimizado con un empaquetamiento adecuado; mientras que, el problema del uso abusivo de la maquinaria puede ser solucionado con la minimización de cortes. Las soluciones anteriores, son de distinta naturaleza y aparecen en la literatura bajo diversos nombres, e.g. corte de material, pérdida de recortes, problema de embalaje de contenedores o tiras, problema de carga de contenedores, problemas de anidamiento, problema de la mochila, etc.

Para ambos problemas mencionados, existen algoritmos que representan soluciones exactas y otros que representan soluciones que hacen uso de heurísticas. Finalmente, una empresa debe decidir que problema prefiere atacar, o si desea atacar ambos, y que tipo de solución es mas conveniente usar; para lo cual, debe analizar y discutir las ventajas y desventajas que conlleva cada algoritmo.

Su trabajo consiste en:

1. Brindar algoritmos que solucionen uno o los dos problemas presentados (ya que cada integrante del equipo debe presentar una solución diferente existe la posibilidad de atacar ambos problemas, si asıse desea).
2. Proporcionar una interfaz que permita a un usuario ingresar los datos necesarios y visualizar el resultado de los algoritmos propuestos, o en su defecto la posibilidad de cargar un archivo con datos de entrada y escribir el archivo de datos de salida correspondiente;
3. Hallar e indicar la complejidad de los algoritmos propuestos.
4. Generar archivos de entrada, siguiendo el formato establecido en la siguiente sección, de cantidades de datos diversas.
5. Usando los archivos de entrada generados, analizar y discutir la eficiencia de las soluciones indicadas mediante:
   - porcentaje de desperdicio en el caso de empaquetamiento, y número de cortes en el caso de recortes;
   - tablas de comparción de tiempos de ejecución para cada algoritmo implementado y cada entrada de datos (tiempo de ejecución en función a algoritmo vs entrada);
   - tablas de comparción de uso de memoria para cada algoritmo implementado y cada entrada de datos (memoria consumida en función a algoritmo vs entrada).
6. Emitir conclusiones en función de los datos levantados en el punto anterior.

2. Entrada de datos

El archivo, de datos, de entrada deberá seguir, **extrictamente**, el siguiente formato:

1. Dimensiones de la(s) plancha(s) a utilizar (*Ancho x Alto*)
2. Cantidad de formatos rectangulares (n)
3. Las siguientes n filas presentarán cada una las dimensiones de cada formato y la cantidad de piezas a cortar/encajar con dicho formato (Identificador de formato, Ancho, Alto, Número de Piezas por formato), ningun formato deberá tener dimensiones mayores a las dimensiones indicadas para la(s) plancha(s).

```
720 670
8
A 120 120  1
B 285 130  1
C 200 300  1
D 165 320  1
E 235 470  1
F 200 170  1
G 285 220  1
H 555 200  1
```

Nota: El *Identificador de formato* ha de ser formado por la union de letras mayúsculas. Cada letra representa un formato diferente de pieza. No deben existir dos *Identificadores* indicando un mismo formato. En caso de tener mas de 26 formatos de pieza (*A-Z*) agregar letras al *Identificador de formato* (*AA-ZZ, AAA-ZZZ, etc*).

3. Salida de datos

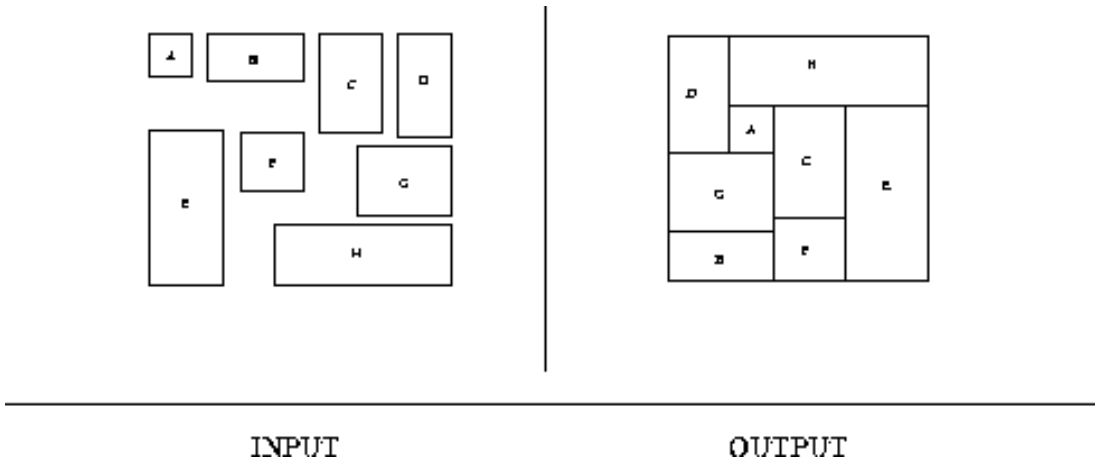
El archivo, de datos, de salida deberá seguir, **extrictamente**, el siguiente formato (guiese del ejemplo):

1. Indicar la cantidad, m, de planchas utilizadas
2. Indicar el porcentaje de desperdicio total y el área desperdiciada.
3. Indicar el número de cortes a realizar
4. A continuación existirán m grupos indicando la disposición de las piezas en la(s) plancha(s) (esta disposición se dará indicando el *Identificador de fomato* de cada pieza concatenado de un número, recordando que puede haber mas de una pieza por formato, seguido de sus las coordenadas x,y en la(s) plancha(s) a partir de donde se ubicará la esquina superior izquierda de tal pieza, finalmente la letra N o G indicando N que la pieza coloca en posición normal y G que la pieza se coloca girada)

```
Planchas: 1 plancha utilizada
Desperdicio: 0 %, Area: 0 metros cuadrados
Cortes: 7
Plancha 1
A1 165 200  N
B1 0 450  N
C1 285 200  N
D1 0 0  N
E1 485 200  N
F1 285 500  N
G1 0 320  N
H1 165 0  N
```

Nota: El hecho de que aparezcan tanto los datos de desperdicio en *porcentaje y area* como de cortes, no significa que su algoritmo deba optimizar ambos resultados. Recuerde que son dos problemas diferentes y sólo se exige que un algoritmo optimice un resultado.

La Figura **??** muestra: a la izquierda, diferentes piezas a encajar en una o mas plancha(s); y, a la derecha, las piezas encajadas de forma a minimizar el área desperdiciada. Así mismo, pueden observarse la cantidad de cortes a realizar.



Cuadro 1: Entrada de Datos y Salida de Datos: piezas encajadas de forma a optimizar el area usada de la(s) plancha(s)

4. Restricciones sobre la implementación del algoritmo

1. Cada grupo es libre de investigar uno o ambos problemas de corte y empaquetamiento 2D. Recalcando que por integrante debe haber una algoritmo diferente solucionando un problema (desperdicio o numero de cortes).
2. Los algoritmos propuestos pueden formar o no parte de los revisados en clase. Sin embargo, entre las propuestas de solución como máximo debe haber una propuesta basada en aproximaciones.
3. La complejidad de cada algoritmo investigado debe ser indicada, así como las ventajas y desventajas que implican y si presentan alguna solución eficiente. Debe implementar
4. Deben haber una solución, de corte y/o de empaquetamiento, por integrante de grupo (máximo tres integrantes).
5. Debe desarrollar una interfaz que permita visualizar el resultado obtenido para un usuario final, y adicionalmente debe permitirse cargar un archivo de entrada y genrear su respectivo archivo de salida (siguiendo **estrictamente** los formatos indicados).
6. Debe analizar e indicar la complejidad de cada algoritmo propuesto.
7. Debe permitir medir los tiempos en que los algoritmos desarrollados resuelven el problema, y la cantidad de memoria consumida.
8. Debe ejecutar un análisis experimental para demostrar cuál es el mejor algoritmo a utilizar para corte y cuál es el mejor algoritmo a utilizar para empaquetamiento.

5. Experimentación

- Debe generar datasets que representen retos interesantes para el problema de cortar y/o empaquetar, aidiconalmente procurar tener también la solución óptima.
- Debe mostrar un análisis para indicar a partir de que cantidad de instancias las algoritmos presentadas se vuelven inviables.
- Debe discutir la performance de los algoritmos utilizados en cuestión de tiempo y espacio (representadas en cifras y también utilizando la notación Big O).

6. Evaluación

Ver la hoja anexa: Rubrica de Calificación

7. Fecha de Presentación

Última sesión de la semana 7.

Anexo

Item	Sobresaliente	Satisfactorio	Deficiente
Generación de los archivos de entrada	Implementa funciones y/o clases para generar diferentes casos para el algoritmo.	Utiliza archivos de entrada generados manualmente.	No implementado
	(2 puntos)	(1 punto)	(0 puntos)
Informe	Elabora una introducción presentando el problema y explicando la motivación para el desarrollo del proyecto.	Elabora una introducción simple, describe el problema de manera elemental o no describe la motivación.	No implementado
	(1 punto)	(0.5 puntos)	(0 puntos)
	Define objetivos del proyecto de manera consistente con el problema planteado.	Define objetivos simples, poco claros o inconsistentes con el problema.	No implementado
	(1 punto)	(0.5 puntos)	(0 puntos)
	Elabora el marco teórico del informe explicando detalladamente todas las estrategias usadas para dar solución al problema.	Elabora un marco teórico haciendo uso únicamente de materiald e clase.	No implementado
	(1 punto)	(0.5 puntos)	(0 puntos)
	Análisis de la complejidad algorítmica de cada una de las estrategias usadas para solucionar el problema.	Analiza la complejidad solo de algunas estrategias o de manera superficial.	No implementado
	(2 puntos)	(1 punto)	(0 puntos)
	Propone conclusiones que discuten las ventajas y desventajas de los algoritmos analizados, la performance respecto al hardware utilizado, proponen mejoras y guardan coherencia con los objetivos planteados.	Propone conclusiones simples o que no guardan coherencia con los objetivos.	No implementado
	(2 puntos)	(0.5 punto)	(0 puntos)
Implementación	Implementa soluciones para el problema de empaquetamiento y para el problema de cortes.	Implementa soluciones unicamente para empaquetamiento o para cortes.	No implementado
	(1 punto)	(0.5 puntos)	(0 puntos)
	Implementa una interfaz de usuario que permita la interacción adecuada con la aplicación y la visualización adecuada de los resultados.	Implementan una interfaz de usuario inadecuada o visualización de resultados insuficiente.	No implementado
	(2 puntos)	(1 puntos)	(0 puntos)
	Implementan y explican correctamente las estrategias de solución del problema. Cuenta con por lo menos una estrategia por integrante de grupo.	Implementan insuficientemente las estrategias de solución o no explican satisfactoriamente el funcionamiento .	No implementado
	(4 puntos)	(1 punto)	(0 puntos)
	Presenta adecuadamente y a tiempo los entregables y presentación del proyecto.	Presenta a destiempo los entregables del proyecto .	No implementado
	(1 punto)	(0.5 puntos)	(0 puntos)
Experimentación	Experimenta con varios tamaños de los datasets (mínimo 4 experimentos). Documenta la performance, en términos de tiempo de ejecución y memoria, para cada entrada de datos y cada algoritmo..	Experimenta en pocos tamaños de los datasets (menos de 4 experimentos). Documenta la performance, en términos de tiempo de ejecución o memoria, para cada entrada de datos y cada algoritmo	No experimenta.
	(2 punto)	(0.5 puntos)	(0 puntos)
	Experimenta con un dataset suficientemente grande y con piezas relevantes. Su propuesta muestra ser escalable (i.e. funciona con data de gran tamaño) aún sin encontrar una solución en tiempo corto, se realiza la computación.	Experimenta con un dataset suficientemente grande y con piezas relevantes. Su propuesta no muestra ser escalable (i.e. funciona con data de gran tamaño) aún sin encontrar una solución en tiempo corto, se realiza la computación	No experimenta.
	(1 punto)	(0.5 puntos)	(0 puntos)