Taller de ordenamiento

Laboratorio
Algoritmos y Estructura de Datos I

DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

1er Cuatrimestre 2019

Menú del día

- Programación, Programación, Programación:

Menú del día

- Programación, Programación, Programación:
  - Cápsula convexa

Menú del día

- Programación, Programación, Programación:
  - Cápsula convexa
  - BPP (Bin Packing Problem)

Menú del día

- Programación, Programación, Programación:
  - Cápsula convexa
  - BPP (Bin Packing Problem)
  - Anagramas

Primer Problema

Convex Hull (wikipedia)

En matemática se define la envolvente convexa, envoltura convexa o cápsula convexa de un conjunto de puntos X de dimensión n como la intersección de todos los conjuntos convexos que contienen a X.

- Vamos a trabajar con n $= 2$.
- La cápsula convexa de un conjunto de puntos en $\mathbb{R}^2$, se puede observar como el conjunto de puntos "de más afuera".

Convex Hull

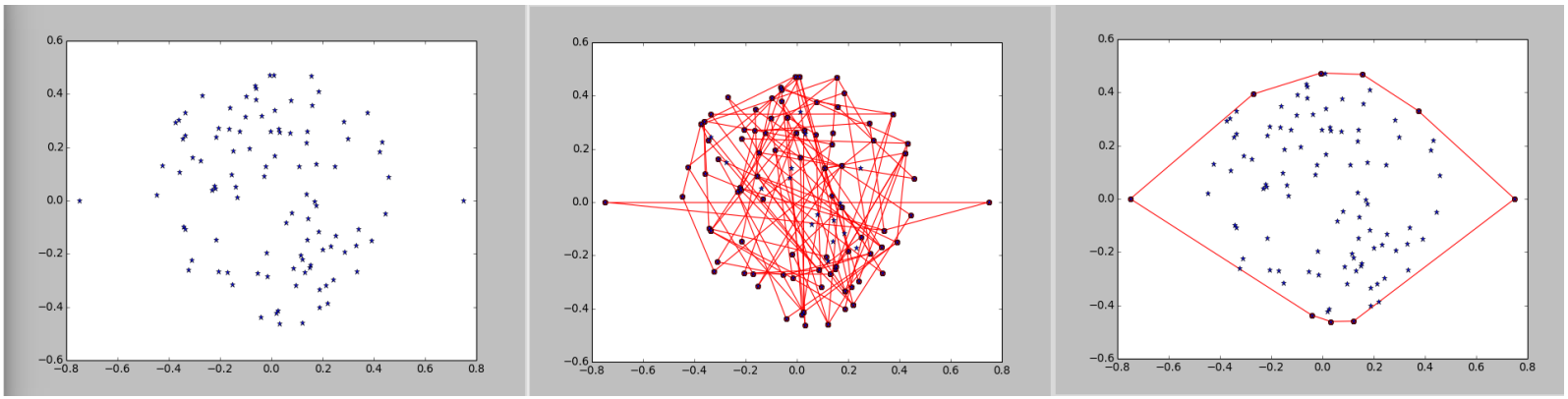
- Para atacar el problema, buscamos ordenar los números de acuerdo a la coordenada x.

Convex Hull

- Para atacar el problema, buscamos ordenar los números de acuerdo a la coordenada x.
- Luego, es más facil buscar un conjunto de puntos que los envuelva a todos.

Convex Hull

- Para atacar el problema, buscamos ordenar los números de acuerdo a la coordenada x.
- Luego, es más facil buscar un conjunto de puntos que los envuelva a todos.



¡Un momento! ¿Cómo ordeno?

Tenemos varias formas de hacerlo:

- Insertion Sort

¡Un momento! ¿Cómo ordeno?

Tenemos varias formas de hacerlo:

- Insertion Sort
- Selection Sort

¡Un momento! ¿Cómo ordeno?

Tenemos varias formas de hacerlo:

- Insertion Sort
- Selection Sort
- Bubble Sort
- Merge Sort
- Heap Sort
- Quick sort
- Bogo Sort
- Timsort
- etc. sort

Repasando lo visto: Insertion y Selection

Idea

- Insertion Sort: *Inserto* ordenado al principio del vector, buscando la posición que le corresponde.
- Selection Sort: *Selecciono* el minimo del vector y lo posiciono donde correspone.

Convex Hull

- Disponemos de un código, el cual debemos completar (convexa.cpp)

Convex Hull

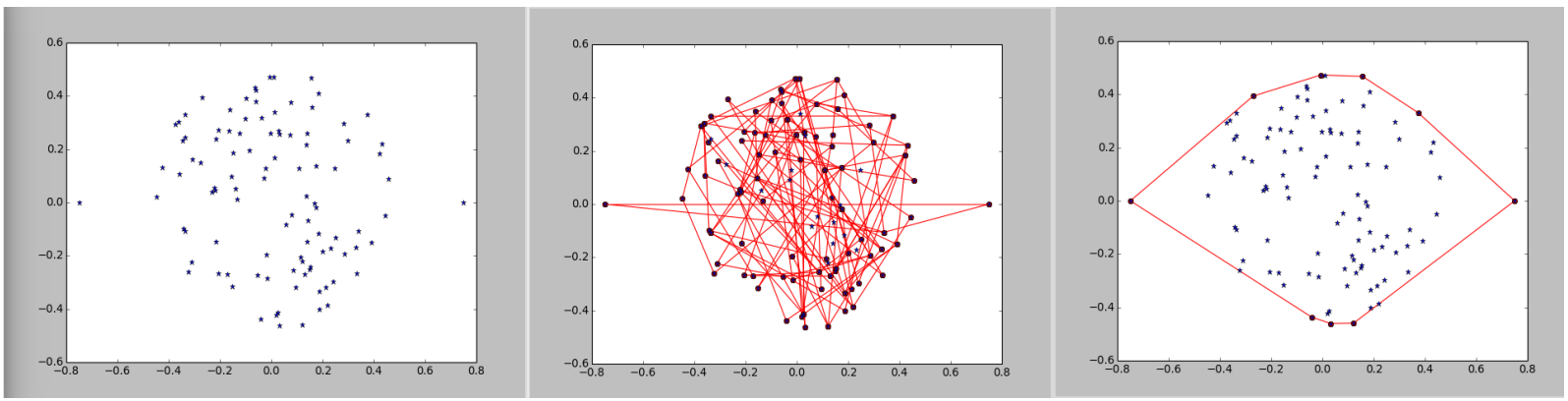
- Disponemos de un código, el cual debemos completar (convexa.cpp)
- El código genera los set de puntos y los ordena.

Convex Hull

- Disponemos de un código, el cual debemos completar (convexa.cpp)
- El código genera los set de puntos y los ordena.
- Tenemos una herramienta para graficar el comportamiento del algoritmo (dibujar.py)
- Para correrlo, se debe ejecutar desde linea de comando » **python dibujar.py**

Convex Hull

- Disponemos de un código, el cual debemos completar (convexa.cpp)
- El código genera los set de puntos y los ordena.
- Tenemos una herramienta para graficar el comportamiento del algoritmo (dibujar.py)
- Para correrlo, se debe ejecutar desde linea de comando » **python dibujar.py**

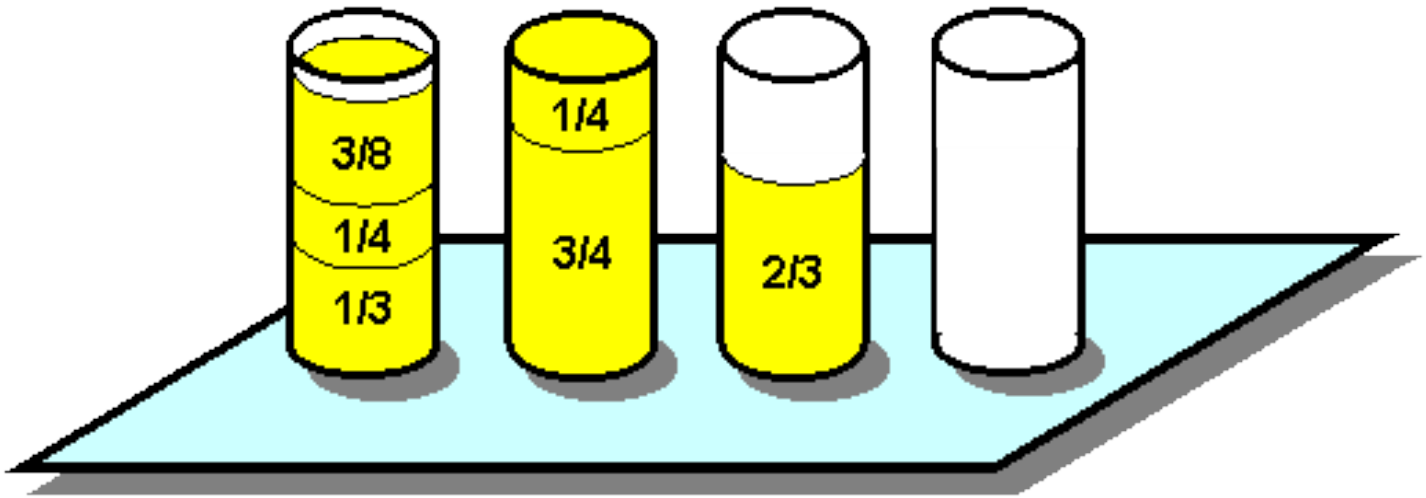


Segundo Problema

Segundo Problema

Bin Packing Problem

Dada una lista de objetos junto a sus volúmenes y una lista de contenedores de dimensión fija, encontrar el mínimo número de contenedores para que todos los objetos sean asignados a algún contenedor.



BPP

- Es un problema muy estudiado, de la clase NP-Hard (algo3)
- Encontrar la solución exacta es computacionalmente difícil

BPP

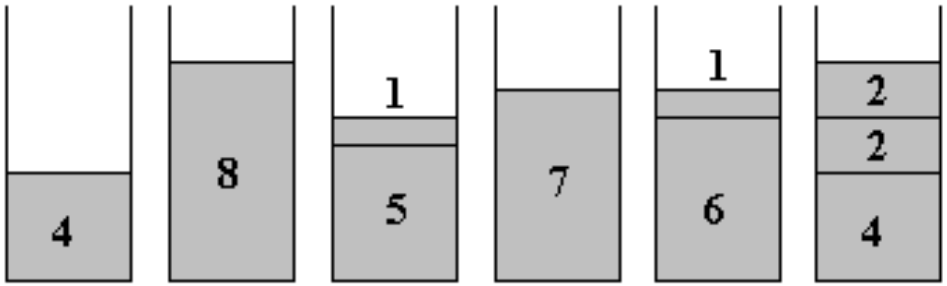
- Es un problema muy estudiado, de la clase NP-Hard (algo3)
- Encontrar la solución exacta es computacionalmente difícil
- Podemos intentar encontrar soluciones aproximadas utilizando heurísticas

BPP

- Es un problema muy estudiado, de la clase NP-Hard (algo3)
- Encontrar la solución exacta es computacionalmente difícil
- Podemos intentar encontrar soluciones aproximadas utilizando heurísticas
  - Next Fit: Ponemos los números de acuerdo al orden en el que vienen.

BPP

- Es un problema muy estudiado, de la clase NP-Hard (algo3)
- Encontrar la solución exacta es computacionalmente difícil
- Podemos intentar encontrar soluciones aproximadas utilizando heurísticas
  - Next Fit: Ponemos los números de acuerdo al orden en el que vienen.
  - Para un conjunto S = {4, 8, 5, 1, 7, 6, 1, 4, 2, 2} sobre contenedores de volumen 10

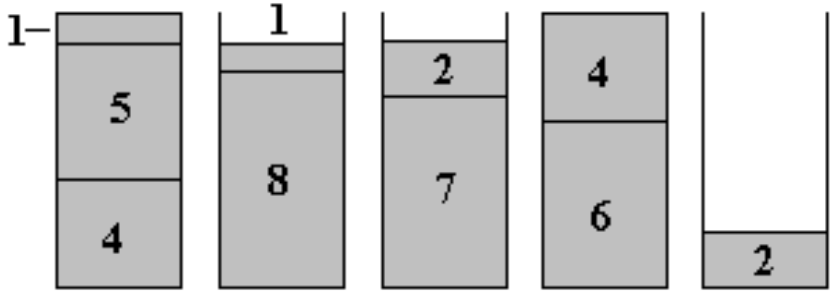


BPP

- Es un problema muy estudiado, de la clase NP-Hard (algo3)
- Encontrar la solución exacta es computacionalmente difícil
- Podemos intentar encontrar soluciones aproximadas utilizando heurísticas
  - Best Fit: Ponemos los números que mejor se ajustan al volumen disponible, según el orden en que vienen.

BPP

- Es un problema muy estudiado, de la clase NP-Hard (algo3)
- Encontrar la solución exacta es computacionalmente difícil
- Podemos intentar encontrar soluciones aproximadas utilizando heurísticas
  - Best Fit: Ponemos los números que mejor se ajustan al volumen disponible, según el orden en que vienen.
  - Para un conjunto S = {4, 8, 5, 1, 7, 6, 1, 4, 2, 2} sobre contenedores de volumen 10

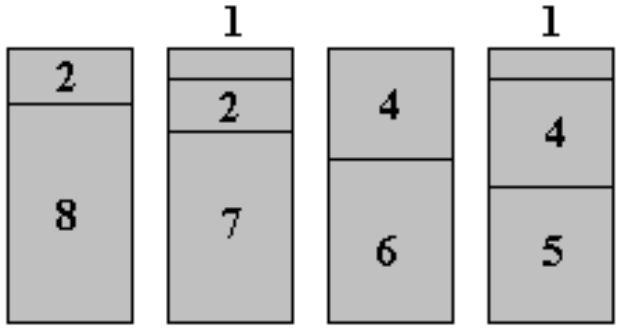


BPP

- Es un problema muy estudiado, de la clase NP-Hard (algo3)
- Encontrar la solución exacta es computacionalmente difícil
- Podemos intentar encontrar soluciones aproximadas utilizando heurísticas
  - Best Fit Ordenado: Ordenamos y ponemos los números que mejor se ajustan al volumen disponible, según el orden en que vienen.

BPP

- Es un problema muy estudiado, de la clase NP-Hard (algo3)
- Encontrar la solución exacta es computacionalmente difícil
- Podemos intentar encontrar soluciones aproximadas utilizando heurísticas
  - Best Fit Ordenado: Ordenamos y ponemos los números que mejor se ajustan al volumen disponible, según el orden en que vienen.
  - Para un conjunto S = {8, 7, 6, 5, 4, 4, 2, 2, 1, 1}



BPP - Programando

- Disponemos de un código, el cual debemos completar (BPP.cpp)

BPP - Programando

- Disponemos de un código, el cual debemos completar (BPP.cpp)
- El programa toma como parámetro un archivo de texto

BPP - Programando

- Disponemos de un código, el cual debemos completar (BPP.cpp)
- El programa toma como parámetro un archivo de texto
  - El formato es #números, volumen del contenedor, seguido de una lista de valores.

BPP - Programando

- Disponemos de un código, el cual debemos completar (BPP.cpp)
- El programa toma como parámetro un archivo de texto
  - El formato es #números, volumen del contenedor, seguido de una lista de valores.
- El programa se encarga de ejecutar Best Fit antes y después de ordenar la lista.

BPP - Programando

- Disponemos de un código, el cual debemos completar (BPP.cpp)
- El programa toma como parámetro un archivo de texto
  - El formato es #números, volumen del contenedor, seguido de una lista de valores.
- El programa se encarga de ejecutar Best Fit antes y después de ordenar la lista.
- Contamos con 4 archivos de volúmenes para ordenar, desde 100 elementos a 1.000.000.

BPP - Programando

- Disponemos de un código, el cual debemos completar (BPP.cpp)
- El programa toma como parámetro un archivo de texto
  - El formato es #números, volumen del contenedor, seguido de una lista de valores.
- El programa se encarga de ejecutar Best Fit antes y después de ordenar la lista.
- Contamos con 4 archivos de volúmenes para ordenar, desde 100 elementos a 1.000.000.
- Calcular el tiempo de cómputo empleado por las implementaciones de ordenamiento para cada valor de entrada. Proponer un orden de complejidad del método. (Usar las funciones vistas en el laboratorio de búsqueda).

Tercer Problema

Tercer Problema

Anagramas

El DC necesita de nuestra ayuda. De tanta gente que vino a la Semana de la Computación se perdió la implementación del stand Anagramas y el departamento sabe que en algoritmos 1 está el futuro de la computación, por eso nos pidieron ayuda.

- Una palabra es anagrama de otra si las dos tienen las mismas letras, con el mismo número de apariciones.

Tercer Problema

Anagramas

El DC necesita de nuestra ayuda. De tanta gente que vino a la Semana de la Computación se perdió la implementación del stand Anagramas y el departamento sabe que en algoritmos 1 está el futuro de la computación, por eso nos pidieron ayuda.

- Una palabra es anagrama de otra si las dos tienen las mismas letras, con el mismo número de apariciones.
- Por ejemplo la palabra "delira" es anagrama de la palabra "lidera".

Anagramas

Anagramas

- Se pide implementar un programa que tome una palabra ingresada por el usuario e imprima por pantalla todas las palabras del idioma que son anagrama de esa.

Anagramas

- Se pide implementar un programa que tome una palabra ingresada por el usuario e imprima por pantalla todas las palabras del idioma que son anagrama de esa.
- Como esta implementación puede tener repercusiones educativas internacionales vamos a hacerlo para el idioma ingles, usando palabras que solo tienen letras minusculas de la "a" a la "z".

Dos implementaciones distintas

Usando ordenamiento

Para ver si p1 y p2 son anagramas basta con ordenar los caracteres de cada una y comparar si el string resultante es el mismo.

Dos implementaciones distintas

Usando números primos

- Asignamos a cada letra del alfabeto un número primo distinto.
- Calculamos los números primeros correspondientes a las letras de la palabra
- Multiplicamos estos números y obtenemos un número que identifica a la palabra.
- Por el teorema fundamental del álgebra a cada combinación de letras se le asigna un número que identifica unívocamente a todos los anagramas.
- Entonces alcanza con comparar si los números obtenidos son iguales.

A programar

- En anagramas.cpp programar cada una de las implementaciones de esAnagrama().
- Para probar: el main toma una palabra en ingles y chequea la cantidad de anagramas que hay en el diccionario.

A Competir!

El primer grupo que tenga todo anagramas terminado (las dos implementaciones) y me diga cual es la cantidad de anagramas en el diccionario de las siguientes palabras se lleva un premio:

- deal
- pyrotic
- else

HINT!: A la hora de correr el diccionario completo, hay una implementacion mas rápida que la otra. Piensen cual usar!