UNIVERSIDAD DE GRANADA

/ UGR / decsai

Departamento de Ciencias de la Computación e Inteligencia Artificial

Asignatura: Algorítmica

Grado en Ingeniería Informática

Relación de problemas: Tema 3

1. Introducción

Para elaborar los ejercicios es recomendable utilizar la siguiente bibliografía:

- G. Brassard, P. Bratley, "Fundamentos de Algoritmia", Prentice Hall, 1997
- J.L. Verdegay: Lecciones de Algorítmica. Editorial Técnica AVICAM (2017)

Antes de realizar los ejercicios prácticos, se recomienda al estudiante revisar y comprender las diapositivas estudiadas en clase y complementarlas con la información procedente de la bibliografía básica y recomendada en la guía docente de la asignatura.

En la relación de problemas se presentarán, para cada temática, un conjunto de preguntas sobre conceptos teóricos que el alumno debe dominar antes de abordar los ejercicios prácticos.

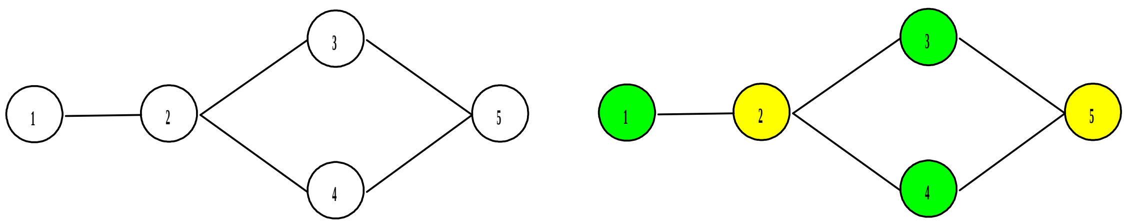
2. Algoritmos voraces (greedy): Conceptos

1. ¿Cuáles son las principales características de los algoritmos voraces (greedy), sus ventajas y sus limitaciones?.
2. Explique qué componentes hay que diseñar para construir un algoritmo greedy. Por último, ponga un problema de ejemplo donde se vea claro el diseño de estas componentes.
3. Explique cuál es el procedimiento general de un algoritmo greedy. Por último, continúe con el ejemplo del ejercicio anterior y escriba el algoritmo greedy que resuelva el problema.
4. Para comprobar la eficacia de un algoritmo greedy, ¿qué se debe hacer? Compruebe la eficacia del algoritmo que ha diseñado en el ejercicio anterior, dando un contraejemplo o demostrando su funcionamiento.
5. En el libro G. Brassard, P. Bratley, "Fundamentos de algoritmia", primeros capítulos, aparece una introducción a los conceptos básicos sobre la inducción matemática. Explique en qué consiste esta técnica y ponga un ejemplo.
6. ¿En qué consiste el problema del Árbol Generador Minimal? Explíquelo y defina aquellos conceptos de grafos que aparezcan en la definición, con ejemplos de cada uno.
7. ¿Qué dos algoritmos óptimos hay con la técnica greedy para resolver el problema del árbol generador minimal? Explique cuál es la idea general de cada uno y las diferencias entre ambos.
8. El algoritmo de Kruskal: Escriba su diseño mediante la técnica greedy, junto con el algoritmo. Ponga un ejemplo, paso a paso, de su funcionamiento.
9. Demuestre, explicando cada paso, que el algoritmo de Kruskal devuelve la solución óptima.
10. El algoritmo de Prim: Escriba su diseño mediante la técnica greedy y detalle el algoritmo. Ponga un ejemplo, paso a paso, de su funcionamiento.
11. Demuestre, explicando cada paso razonadamente, que el algoritmo de Prim devuelve la solución óptima.
12. Explique cómo se diseña el algoritmo de Dijkstra con la técnica greedy y escriba dicho algoritmo. Por último, ponga un ejemplo del funcionamiento del algoritmo en un grafo de no menos de 7 vértices.
13. El algoritmo de Dijkstra siempre proporciona la solución óptima. Demuéstrelo.

14. El problema del viajante de comercio. Escriba su definición formal, junto con los conceptos de grafos asociados al problema. ¿Cuál es la idea general para resolver este problema mediante la técnica greedy? ¿Qué es una heurística? ¿Podría dar una heurística para resolver el problema, razonando su buen funcionamiento?
15. Escriba cómo diseñar el algoritmo greedy para resolver el problema del viajante de comercio, utilizando la heurística de selección, en cada paso, de la arista factible de menor coste. Por último, escriba el algoritmo.
16. ¿Es la solución planteada en el ejercicio anterior óptima? En caso afirmativo, demuéstrelo. En caso negativo, proporcione un contraejemplo en un grafo de su invención (pista: es fácil construir un grafo que sirva para la demostración con n=5 vértices).
17. El problema de la mochila. Describa formalmente en qué consiste el problema de la mochila. ¿Qué tipos de problema de la mochila conoce?
18. Demuestre que la heurística para solucionar con algoritmos greedy el problema de la mochila continuo proporciona la solución óptima.
19. ¿En qué consiste el problema de planificación de tareas que trata de minimizar el tiempo que una tarea espera hasta finalizar su ejecución? Descríbalo formalmente. ¿Hay alguna heurística óptima para este problema? Diga cuál y ponga un ejemplo de su uso.
20. Exponga la demostración de la estrategia óptima para resolver el problema de minimización del tiempo de tareas en el sistema y explíquela con sus palabras.

3. Algoritmos voraces (greedy): Ejercicios prácticos

21. El problema de minimización del tiempo de tareas en el sistema. Diseñe las componentes greedy para resolver el problema con la técnica voraz. Proporcione el pseudo-código de este algoritmo y ponga un ejemplo de su funcionamiento, paso a paso.
22. El problema del coloreo de un grafo consiste en: Dado un grafo G=(V, A), no dirigido, y un conjunto *K* de colores, se pide dibujar todos los vértices del grafo con el mínimo número de colores posible, con la restricción de que no puede haber dos vértices adyacentes (unidos por una arista) que tengan el mismo color. Se pide: Diseñar un algoritmo greedy que resuelva este problema. Como ejemplo, se proporcionan dos figuras con el coloreo de un grafo. A la izquierda, el grafo original. A la derecha, el grafo coloreado con 2 colores:



23. Con la crisis, para minimizar costes, la compañía de autobuses de la ciudad ha decidido rediseñar la ruta por la que pasa un autobús dado de modo que el coste en gasolina por realizar la ruta completa sea mínimo. La ruta en sí tiene un total de 7 paradas y hay que tener en cuenta que, tras viajar por todas las paradas, el autobús vuelve a comenzar el mismo recorrido desde la primera. Podemos suponer que el coste de la gasolina por viajar de una parada a otra viene dado por la siguiente matriz simétrica *M*, y que *M[i][j]* representa el coste de viajar desde la parada de la fila *i* hasta la parada de la columna *j*. Se pide diseñar un algoritmo greedy que devuelva la secuencia de paradas por las que el autobús debe pasar, antes de volver a la primera, de modo que el coste de la gasolina sea mínimo.

$$M=\begin{pmatrix} 0 & 7 & 6 & 9 & 2 & 4 & 8 \\ 7 & 0 & 3 & 6 & 3 & 1 & 8 \\ 6 & 3 & 0 & 4 & 4 & 5 & 3 \\ 9 & 6 & 4 & 0 & 9 & 1 & 9 \\ 2 & 3 & 4 & 9 & 0 & 4 & 2 \\ 4 & 1 & 5 & 1 & 4 & 0 & 5 \\ 8 & 8 & 3 & 9 & 2 & 5 & 0 \end{pmatrix}$$

24. La empresa web Bia My-Tseline te ha contratado porque quiere extender su servicio web de trazado de rutas por carretera a dispositivos GPS, para encontrar el mejor recorrido entre ciudades que minimice el kilometraje. Básicamente, el dispositivo GPS tendrá que calcular, dada nuestra posición inicial, cuál es el recorrido óptimo (de menor kilometraje) hasta nuestro destino. Se pide diseñar un algoritmo greedy que resuelva este problema (diseño+procedimiento general del algoritmo). Pon, además, un ejemplo de su funcionamiento.
25. Un buscador de tesoros ha encontrado, en una cueva submarina, sacos llenos de oro, plata y cobre en polvo. Llegar a esa cueva es difícil y seguramente no sabría volver a ella de nuevo, si decidiese salir a la superficie a por herramientas que le permitiesen sacar todo el tesoro. Por tanto, tiene que sacar de la cueva todo el oro, plata y cobre que pueda de una sola vez. Para ello, cuenta con una bolsa del Marcadota que ha recogido en el mar porque algún turista cochino la habría tirado al agua. La bolsa, sin romperse, puede aguantar un peso de 3kg, y tendría que transportar 4 bolsas de 0.5kg de oro en polvo, 7 de plata de 0.8kg, y 10 de cobre de 1.2kg. Sabiendo que el oro cotiza en el mercado a 100 euros el kilo, la plata a 85 euros el kilo, y el cobre a 27 euros el kilo, ¿cuánta cantidad en polvo de oro, plata y cobre debe transportar el buscador de tesoros para maximizar su beneficio? Diseñe un algoritmo greedy que resuelva el problema de forma óptima, y proporcione el procedimiento (en pseudo-código) del algoritmo.
26. Suponga que el marinero no puede abrir las bolsas de oro, plata y cobre para extraer el mineral precioso en polvo y que, por tanto, debe coger los sacos enteros. En este caso, generalice el problema y enúncielo para N materiales preciosos diferentes, del que cada uno hay P_1, P_2, …, P_n bolsas del mismo material (P_i es un número entero, ya que no hay medias bolsas), donde todas las bolsas del mismo mineral precioso tienen el mismo peso W_i, cada mineral precioso tiene un coste en el mercado de B_i euros/kg., y la capacidad máxima de la bolsa del Marcadota es de un material que permite aguantar M kg sin romperse.
27. Proporcione un algoritmo greedy que resuelva el problema general anterior (diseño de las componentes greedy + procedimiento en pseudo-código). Si es óptimo, haga la demostración de su optimalidad. En caso contrario, ponga un contraejemplo en un caso concreto en el que el algoritmo no devuelva la solución óptima.
28. Las paradas del servicio de metro de Madrid se pueden representar mediante un grafo. Este grafo es no dirigido, dado que para toda parada A que conecte directamente con otra parada B podemos coger un transporte que nos lleve también de B a A: El túnel es el mismo. Además sabemos que, para cualquier par de paradas de metro que escojamos, podemos llegar de una a otra aunque sea haciendo trasbordo. Con la llegada de la crisis y los planes de austeridad, se ha reducido considerablemente el presupuesto del servicio de metro, de modo que resulta inviable pagar el mantenimiento de todas las conexiones (túneles) que unen paradas consecutivas. Por ello, es necesario cerrar trayectos entre algunas de estas paradas consecutivas para ahorrar el coste de

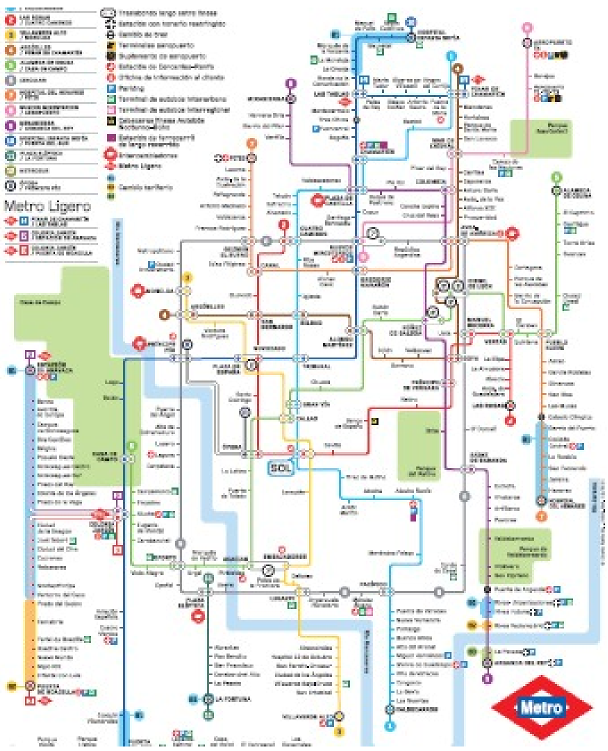
mantener las vías y los túneles del metro. Sin embargo, la comunidad de Madrid no puede permitir que queden “paradas fantasma”; es decir, paradas de las que no lleguen ni salgan vagones, para seguir dando servicio a toda la ciudadanía. Pero sí que quiere reducir el coste de mantenimiento de las vías al mínimo posible de modo que, al igual que antes, se pueda viajar entre cualquier par de paradas consideradas, aunque el tiempo y satisfacción del usuario con el servicio se vea empeorado. Suponga que hay una matriz de *n*n* elementos, con *n*=nº de paradas de metro, que indica las paradas adyacentes que están conectadas entre sí con el coste de mantenimiento del túnel que las une (la matriz contendrá el valor ∞ si no existe túnel entre dos paradas). Se pide: Diseñar un algoritmo greedy que seleccione los túneles afortunados que permitan conectar todas las paradas (en otras palabras, que exista un camino entre cualquier par de paradas) con un coste de mantenimiento mínimo. Nótese que, para que esto ocurra, necesariamente la solución no contendrá ciclos, ya que si los contuviese la solución para viajar entre dos paradas A y B de nodos que forman parte del ciclo no sería óptima: Se podría prescidir de, al menos, un túnel. Escriba el procedimimento greedy que da solución a este problema y, en caso de ser óptimo, demuestre su optimalidad.

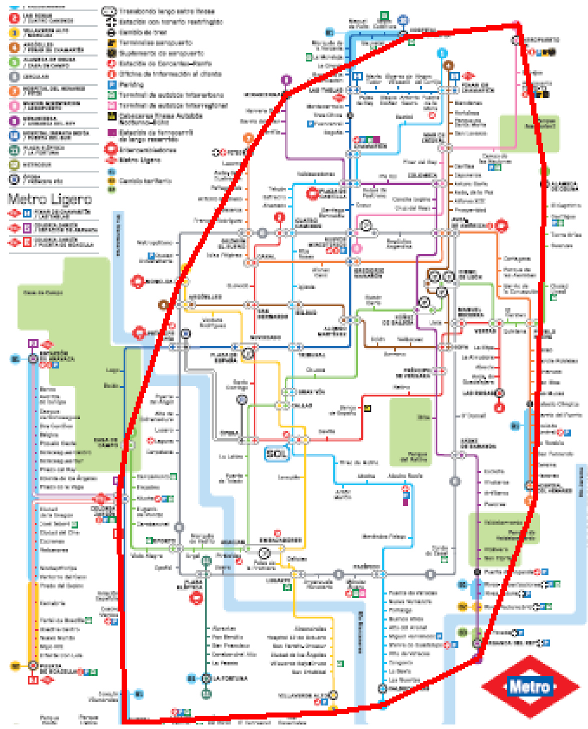
29. Tras un fatídico terremoto, toda la red de carreteras del país se ha visto afectada, de modo que ninguna carretera es transitable entre ninguna ciudad del país y todas ellas necesitan reparación, para que cualquier vehículo pueda viajar de una ciudad a otra. Y ahí está el ejército español, presente como ayuda humanitaria en Haití, con el deber de restablecer una red mínima de carreteras que permita viajar entre cualquier par de ciudades del país para transportar alimentos y artículos de primera necesidad. Sin embargo, su presupuesto es limitado, y sólo pueden restablecer las carreteras de coste mínimo que permitan finalmente hacer que cualquier par de ciudades del país queden conectadas. Como miembro del equipo de inteligencia del ejército, se te pide que digas cuáles son las carreteras que se deban arreglar y cuál será el coste total de las reparaciones, atendiendo a los criterios mencionados: Se debe poder viajar entre cualquier par de ciudades y el coste de reparación de las carreteras tiene que ser mínimo. El perito del servicio de ayuda humanitaria te ha facilitado la información del presupuesto de arreglar cada carretera entre dos ciudades, y te ha proporcionado estos datos en una matriz *M* de tamaño *n*n*, con *n*=nº. de ciudades del país, simétrica, con los elementos de la diagonal principal igual a 0, y donde los valores *M[i][j]= M[j][i]=* coste de reparar la carretera entre la ciudad *i* y la ciudad *j*, si esa carretera ya existía, o el valor ∞ si previamente no había conexión directa entre ambas ciudades. Se pide: Diseñar un algoritmo greedy que resuelva el problema. Escribir, además, el pseudo-código del procedimiento diseñado, y probar su optimalidad. Exponga un caso de ejemplo con 9 ciudades y muestre, paso a paso, cómo actuaría el algoritmo sobre el ejemplo, dando finalmente la solución y su coste total.
30. El Problema de Asignación Cuadrática consiste en asignar un conjunto de *n* localizaciones a *n* instalaciones, atendiendo a que entre cada par de instalaciones habrá un flujo de transporte de elementos y que cada par de localizaciones está a una distancia conocida. Un ejemplo de aplicación real es la construcción de hospitales: Entre cada instalación (UCI, quirófano, consultas, Servicio de Radiología, etc.) y otra instalación habrá un flujo de elementos (pacientes, material, médicos, etc.). Además, entre cada dependencia del edificio donde se construirá el hospital hay una distancia que recorrer. En este caso, hay que asignar el servicio del hospital a las dependencias del edificio tal que se minimice el coste de viajar de una dependencia a otra en función del flujo entre instalaciones. Siendo $\mathbf{F}_{n,n}$ y $\mathbf{D}_{n,n}$ las matrices de flujo entre instalaciones y de distancias entre dependencias, respectivamente, y $\mathbf{p}_n$ un vector con la asignación de instalaciones a dependencias, la función de coste se define como:

$$\min_p \left\{ \sum_{i=1}^{n} \sum_{j=1}^{n} F_{ij} D_{p(i)p(j)} \right\}$$

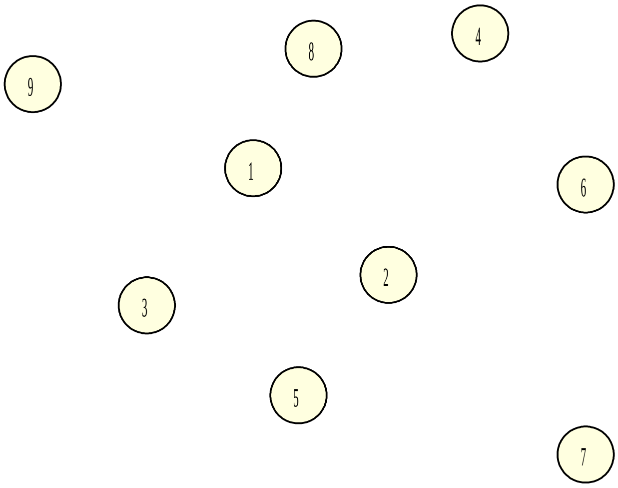
Se pide: Diseñar una solución Branch&Bound para solucionar este problema. Diseñe completamente las componentes del algoritmo y escriba el pseudo-código que solucione el problema. Ponga un ejemplo pequeño (n=4) y explique cómo actúa el algoritmo paso a paso para solucionar el problema.

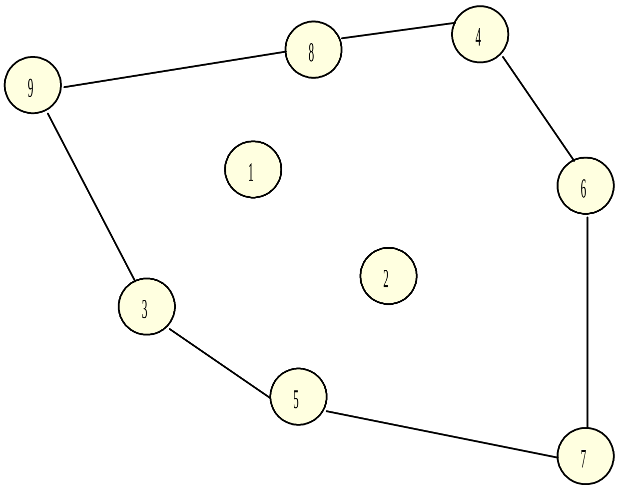
31. Los responsables del metro de Madrid no se aclaran: En ejercicios anteriores dicen que no hay dinero y que hay que recortar gastos en mantenimiento. En este ejercicio, sin embargo, nos plantean que quieren inicial un proyecto para crear un nuevo recorrido circular radial que una todas las paradas de metro más exteriores, de modo que el recorrido encierre interiormente a todas las paradas (salvo aquellas por las que pasa el recorrido) de Madrid, creando un nuevo anillo de metro con el nuevo recorrido circular. Justifican este proyecto por las previsiones de crecimiento de la ciudad que, en el año 2050 y al igual que Londres, necesitará varios recorridos circulares radiales concéntricos. Por hacernos una idea, veamos un extracto del mapa de metro de Madrid (figura de la izquierda):





Resulta sencillo, de forma manual y gráfica, trazar dicho recorrido cuando el número de paradas es relativamente reducido, como es este caso. Basta con reconocer cuáles son las paradas de metro más exteriores tales que, trazando un perímetro entre ellas, ninguna otra parada quede en el exterior del perímetro trazado. (véase el dibujo de la derecha). Sin embargo, la empresa para la que trabajamos no quiere que resolvamos este problema concreto para el caso del metro de Madrid, sino que quiere una solución general que pueda ser aplicada a cualquier servicio de metro de cualquier ciudad del mundo. El problema de encontrar el recorrido circular radial en el servicio de metro de Madrid se puede generalizar a otro problema matemático, conocido como **“Dado un grafo G=(V,A), donde cada vértice v en V viene expresado por sus coordenadas cartesianas en el plano 2-D v=(x,y),** ***encontrar la envolvente convexa del grafo*****”**. La envolvente convexa del grafo se define como el polígono convexo mínimo que encierra a todos los puntos del mismo en su interior. Por ejemplo, supuesto que conocemos las coordenadas (x,y) de cada vértice del grafo de la siguiente figura (a la izquierda), su envolvente convexa se representa en la figura de la derecha (nótese que, para este problema, basta con saber el conjunto de puntos que definen los vértices del grafo, las aristas del mismo no son necesarias en ningún momento para resolver el problema y, por este motivo, no han sido dibujadas).

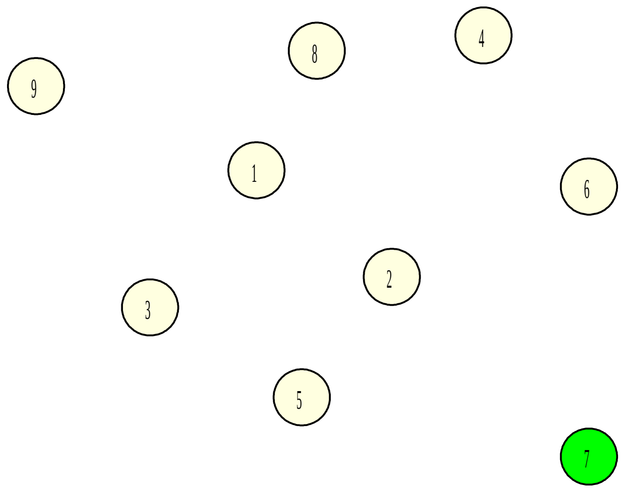




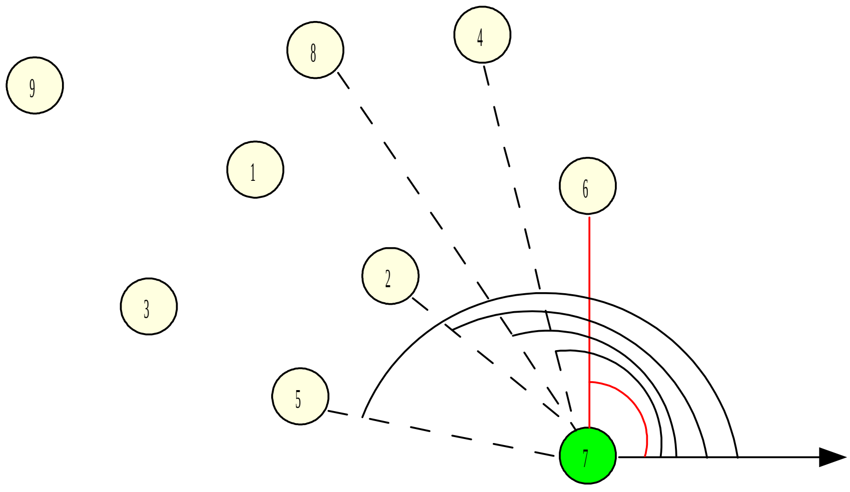
Existe una solución greedy para calcular la envolvente convexa de un conjunto de puntos en el plano, conocido como ***gift wrapping***. La idea básica de este algoritmo consiste en trazar una línea recta entre 2 puntos y, si todos los demás puntos quedan a un único lado (digamos el izquierdo) de la recta, entonces dicha recta forma parte del perímetro del polígono a buscar y los puntos usados para trazarla forman parte de la solución al problema. El esquema básico del algoritmo consiste en:

2. Solución inicial $\mathbf{s_0}$= Seleccionar el punto de menor valor en la ordenada (coordenada *y*) ya que este punto, por ser el más externo en la coordenada *y*, pertenecerá forzosamente a la envolvente convexa.
3. Encontrar el siguiente punto $\mathbf{s_1}$, tal que el ángulo que forma la recta $\mathbf{s_0 s_1}$ con el origen de coordenadas x sea mínimo, y añadir $\mathbf{s_1}$ a la solución.
4. Encontrar el siguiente punto $\mathbf{s_2}$, tal que el ángulo que forma la recta $\mathbf{s_1 s_2}$ con la recta anterior $\mathbf{s_0 s_1}$ sea mínimo, y añadir $\mathbf{s_2}$ a la solución.
5. Seguir así hasta que se selecciona un punto $\mathbf{s_i}$ tal que $\mathbf{s_i}$=$\mathbf{s_0}$. Fin del algoritmo.

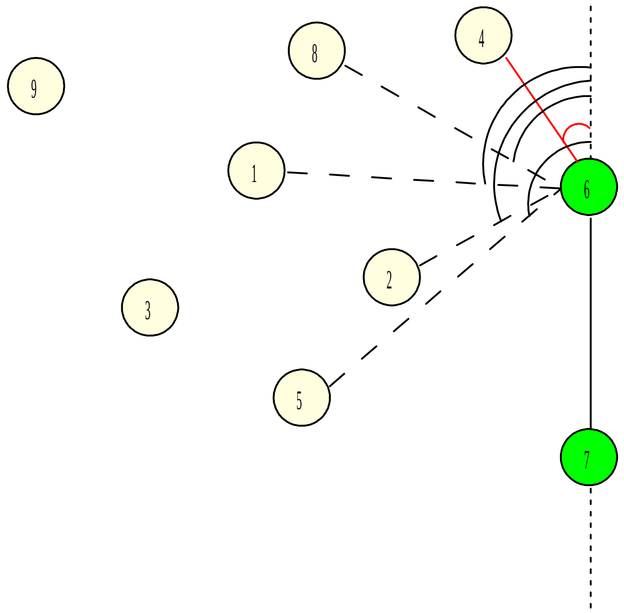
Las siguientes figuras muestran un ejemplo de los 3 primeros pasos del algoritmo:



Paso 1: Selección del nodo inicial.



Paso 2: Selección del nodo cuya recta entre 7 y ese nodo tenga ángulo mínimo con respecto al origen.



Paso 3: Selección del nodo cuya recta de 6 a ese nodo tenga ángulo mínimo con la recta de 7 a 6.

Para realizar este ejercicio, se pide: Plantear el algoritmo gift-wrapping con un diseño de algoritmo greedy. Definir clara y justificadamente cada una de las componentes de este diseño, y proporcionar un procedimiento en pseudo-código, partiendo del procedimiento general greedy, que resuelva este problema. Demuestra, mediante inducción matemática, que el algoritmo siempre selecciona los vértices del grafo más externos, y demuestra también que el algoritmo para (finaliza), y que el resultado que devuelve es la envolvente convexa del grafo. Indica cuál es la eficiencia de este algoritmo diseñado. Por último, expón un caso de ejemplo (inventado por ti) y resuelve el problema como lo resolvería el algoritmo diseñado, explicando cada paso detalladamente.

32. El alcalde de la ciudad pretende financiar un proyecto para implantar un carril bici. Sin embargo, debido a los efectos de la crisis, no hay presupuesto suficiente para implantar el carril en todas las calles de la ciudad. Sin embargo, sí que resulta imprescindible, de cara a obtener ayudas europeas del Plan para la Mejora de la Calidad de Vida aprobado por la Comisión Europea, que desde un punto de interés turístico se pueda llegar a cualquier otro por el carril bici y sin necesidad de viajar fuera del mismo. Por este motivo, se requiere la ayuda de un Ingeniero Informático que diseñe qué calles deberán ser asfaltadas con el carril bici de modo que se resuelva el problema planteado con el mínimo impacto económico. Se pide diseñar un algoritmo Greedy que resuelva este problema. ¿Es óptimo el algoritmo? Demuéstrelo o proporcione un contraejemplo. Por último, explique paso a paso el funcionamiento del algoritmo para el siguiente caso, donde la matriz indica el coste de implantar el carril bici entre un punto turístico de origen (fila) y otro destino (columna).

	PT1	PT2	PT3	PT4	PT5	PT6
PT1		70000	40000	10000	300	110000
PT2	70000		4500	12000	170000	123456

PT3	40000	4500		650123	43410	22300
PT4	10000	12000	650123		53000	23000
PT5	300	170000	43410	53000		123000
PT6	110000	123456	22300	23000	123000	

33. El vecino Pepito es aficionado a descargar series de TV desde la web de diferentes cadenas de televisión. Es domingo por la tarde y se ha quedado sin espacio en el disco duro, pero ha encontrado un DVD con capacidad para almacenar 4.4GB. Debemos ayudarle a seleccionar qué ficheros debe guardar en el DVD de modo que pueda borrar estos ficheros y liberar el máximo espacio en disco. Diseñe un algoritmo greedy para resolver este problema. ¿Es óptimo? Demuéstrelo o proporcione un contraejemplo. Por último, indique paso a paso cómo se ejecutaría el algoritmo si el contenido de la carpeta de descargas del vecino Pepito contiene los siguientes ficheros:

Lost 5x12.avi	500MB
The Walking dead 3x16VOSE.avi	450MB
Revolution2012 1x01.avi	355MB
Prison Break 3x13.avi	560MB
House 2x01 HD.avi	1.1GB
SarahConnorChronicles 1x01Pilot VOSE.avi	800 MB
Lost 5x13 HD.avi	650MB
Lost 5x14 HD.avi	150MB
The Walking dead 2x11VOSE.avi	550MB
The Walking dead 3x08VOSE.avi	335MB
Elementary 1x01.avi	250MB
Homeland.1x03VOSE.avi	576MB

34. Desarrolle un algoritmo de tipo Greedy para factorizar un entero en números primos. Exponga un ejemplo de su funcionamiento para el número 252.

35. Un autobús realiza una ruta determinada entre dos ciudades. Con el tanque de gasolina lleno, el autobús puede recorrer n kilómetros sin parar. El conductor dispone de un mapa de carreteras que le indica las gasolineras que hay en su ruta. Para minimizar el tiempo empleado en recorrer su ruta, el conductor desea pararse a repostar el menor número posible de veces. Diseñe un algoritmo greedy que determine en qué gasolineras tiene que repostar. Indique si el algoritmo es óptimo o no, realizando una demostración de optimalidad o proporcionando un contraejemplo que lo refute.

36. Se tiene un buque mercante cuya capacidad de carga es de k toneladas y un conjunto de contenedores c1, ..., cn cuyos pesos respectivos son p1, ..., pn (expresados también en toneladas). Teniendo en cuenta que la capacidad del buque es menor que la suma total de los pesos de los contenedores:
    a. Diseñe un algoritmo que maximice el número de contenedores cargados.
    b. Diseñe un algoritmo que intente maximizar el número de toneladas cargadas.

37. Tenemos que completar un conjunto de n tareas con plazos límite. Cada una de las tareas consume la misma cantidad de tiempo (una unidad) y, en un instante determinado, podemos realizar únicamente una tarea. La tarea i tiene como plazo límite di y produce un beneficio gi (gi > 0) sólo si la tarea se realiza en un instante de tiempo t ≤ di.
    a. Diseñe un algoritmo greedy que nos permita seleccionar el conjunto de tareas que nos asegure el mayor beneficio posible.
    b. Aplique su algoritmo a la siguiente instancia del problema:

Tarea (i)	1	2	3	4
Beneficio (gi)	50	10	15	30
Plazo límite (di)	2	1	2	1

(Brassard & Bratley, 1997, sección 6.6.2: “Planificación con plazo fijo”)

38. Deseamos optimizar el uso de las aulas de un centro educativo. Dados un conjunto de aulas y un conjunto de clases con un horario preestablecido (la clase i empieza a la hora si y termina a la hora fi), diseñe un algoritmo greedy que minimice el número de aulas necesario para impartir toda la docencia del centro.