Modelos Avanzados de IO

Profesor: Francisco Ballestín

Departamento: Matemáticas para la Economía y Empresa.

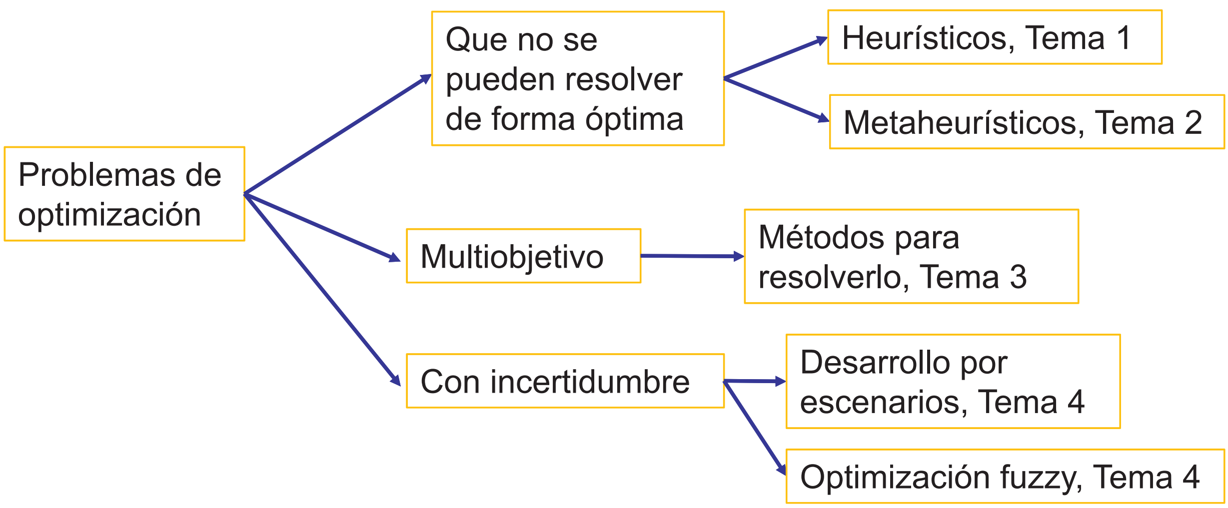
Despacho: 5E04

Horario tutorías: miércoles 9:30-12:30 (online o presencial)

También tutorías a la carta (concertar cita)

Correo: francisco.ballestin@uv.es

Modelos avanzados de IO



Principales elementos a aprender durante el curso
- (Modelizar problema original)
- Incorporar característica (resolución directa, multiobjetivo, incertidumbre)
- Escoger método para su resolución
- Diseñar algoritmo o modelo para su resolución (a mano)
- Seguir/Ejecutar/Resolver algoritmo o modelo (a mano, Lingo, Excel, programar)

Modelos avanzados de IO

Clases y evaluación:

- Clases teóricas
  - explicación de conceptos, con ejemplos. Seguiremos diapositivas. Estarán colgadas en el aula virtual y habrá que imprimirlas y traer a clase.
  - se irán colgando ejercicios para cada tema.
- Clases prácticas (2 tipos)
  - Problemas “a mano”
    - calcular modelo para resolver problema, diseñar algoritmo, seguir algoritmo calculadora
  - Problemas con ordenador
    - Lingo o lingo avanzado para resolver los modelos
    - Excel para seguir los algoritmos diseñados
- Algunas semanas colgaré ejercicios para que los hagáis. Se colgarán las soluciones.

Modelos avanzados de IO

Evaluación:

- **Examen final**: 6 puntos, es obligatorio aprobar este examen (3/6)
  - Ejercicios teórico-prácticos, modelizar, sintaxis indexada, utilización de Excel, diseño de algoritmos
- **Evaluación continua**: 4 puntos
  - Pruebas teórico-prácticas y con ordenador (2 puntos)
  - Trabajo en grupo (1 punto)
  - Parte de la evaluación continua se puede recuperar **(3/4 puntos).**
- El trabajo en grupo no se puede recuperar (1/4 puntos).

Modelos avanzados de IO

Tema 1. Algoritmos Heurísticos

1.1.– Introducción. TSP

Introducción

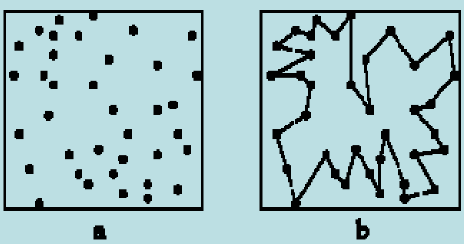
- Los problemas de clase **"P" son "fáciles" de resolver** para los ordenadores; es decir, las soluciones a estos problemas pueden ser calculadas en una cantidad razonable de tiempo (polinomial)
- En los problemas de tipo **"NP", la solución podría ser muy difícil de encontrar** (quizá requeriría miles de millones de años de computación, tiempo no polinomial) pero una vez encontrada, es fácil de comprobar.
- Muchos de los problemas de optimización son del tipo NP
- ***"P versus NP" es uno de los problemas "del Milenio", y se ofrece 1 millón de dólares a cualquier persona que lo resuelva***

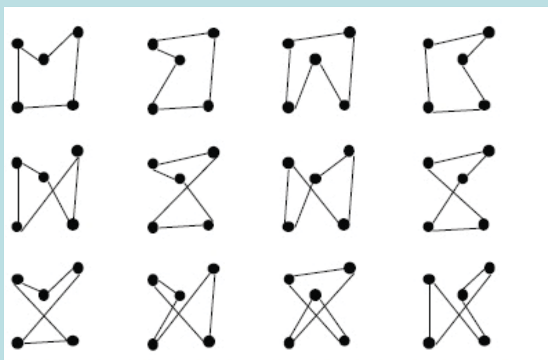
- **Métodos Exactos:** Proporcionan una o todas las soluciones óptimas del problema. Ejemplos: Simplex, B&B

- **Métodos Heurísticos o Aproximados:** Proporcionan buenas soluciones para el problema pero no garantizan que sean óptimas.

El Problema del Viajante (TSP)

Dada una lista de ciudades y las distancias entre cada par de ellas, ¿Cuál es la ruta más corta/más barata posible que visita cada ciudad exactamente una vez y regresa a la ciudad origen?





Hay que considerar (n-1)!/2 rutas posibles

El Problema del Viajante (TSP)

En las décadas de los 50 y de los 60 el problema empezó a ser popular en USA. Gracias a la campaña de publicidad llevada a cabo por "Procter and Gamble" en 1962. La campaña anunciaba un concurso cuyo premio ascendería a 10.000$. Dicha recompensa sería para aquel que ayudara a Toody y Muldoon decidir la ruta más corta entre 33 ciudades de Estados Unidos (Concurso Car 54). Los policías Toody y Muldoon, siempre conduciendo su famoso Car 54, eran personajes de una popular serie americana. Y "su tarea" de visitar las 33 ciudades un claro ejemplo del TSP.

El Problema del Viajante (TSP)

Instancia = caso concreto de un problema, con un tamaño del problema concreto y unos datos concretos. Tenemos que realizar algoritmos que puedan aplicarse a cualquier instancia de un problema.

Dada una lista de 5 ciudades y las distancias entre cada par de ellas se muestra en la tabla, ¿Cuál es la ruta más corta/más barata posible que visita cada ciudad exactamente una vez y regresa a la ciudad origen?

	C1	C2	C3	C4	C5
C1	999	16	63	21	20
C2	57	999	40	46	69
C3	23	11	999	55	53
C4	71	53	58	999	47
C5	27	79	53	35	999

El Problema del Viajante (TSP)

¿POR QUÉ ESTE PROBLEMA?

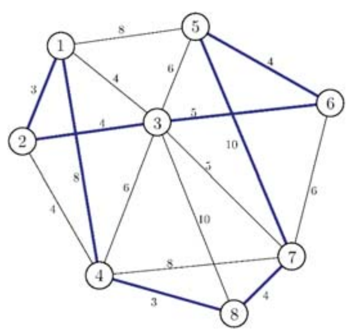
- Resulta muy intuitivo y con un enunciado muy fácil de comprender.
- Es uno de los que más interés ha suscitado en Investigación Operativa.
- Numerosas aplicaciones: impresión de microchips, logística, rutas en almacenes, secuencias de ADN, rutas escolares (buses), reparto de paquetería, producción de circuitos electrónicos (en algunos casos) etc.
- Tiene extensiones y cada una tiene sus aplicaciones reales

El Problema del Viajante (TSP)

Formalmente:

Sea un grafo G=(V,A,C) (completo) donde:
V es el conjunto de vértices/clientes
A es el conjunto de aristas/arcos entre vértices/clientes que representan el camino más corto/barato entre los nodos.
C=(c_{ij}) es la matriz de costes, c_{ij} es el coste (distancia) de la arista (i,j)

Objetivo: Encontrar un ciclo de coste mínimo que pase por todos los vértices exactamente una vez (tour).



Formulación Matemática del TSP

x_{ij} = 1 si después del nodo i se visita el nodo j, 0 otro caso

$$Min \sum_{(i,j)\in A} c_{ij}x_{ij}$$

s.a.

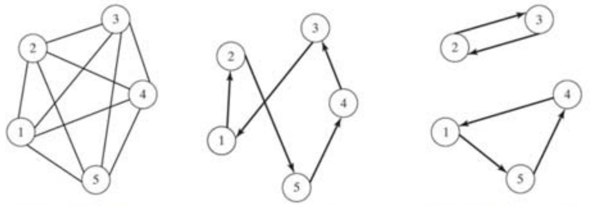
$$\sum_{j=1}^{n} x_{ij} = 1 \qquad \forall i \in V$$

$$\sum_{j=1}^{n} x_{ji} = 1 \qquad \forall i \in V$$

$$\sum_{i\in S, j\in S} x_{ij} \leq |S| - 1 \qquad \forall S \subset V, |S| > 1$$

$$S \text{ cualquier subconjunto}$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in A$$



Problema de 5 ciudades

Solución del recorrido ($x_{12} = x_{25} = x_{54} = x_{43} = x_{31} = 1$)

Solución de subrecorrido ($x_{23} = x_{32} = 1$)($x_{15} = x_{54} = x_{41} = 1$)

- El número de restricciones de eliminación de subtours es exponencial. Para resolver la relajación lineal, se van añadiendo cuando son violadas y se vuelve a resolver el PL.
- Existen formulaciones más eficientes.

Ejemplos de problemas modelizados como un TSP

EJEMPLO TSP1:

La producción diaria en la compañía Rainbow incluye lotes de pintura blanca (B), amarilla (A), roja (R), negra (N) y verde (V). Las instalaciones de producción se deben limpiar entre uno y otro lote. La tabla muestra en minutos los tiempos de limpieza. El objetivo es determinar la secuencia de producción para los lotes de colores que minimice el tiempo de limpieza total.

	Blanca	Amarilla	Negra	Roja	Verde
Blanca	∞	10	17	15	14
Amarilla	20	∞	19	18	15
Negra	50	44	∞	22	25
Roja	45	40	20	∞	26
Verde	30	32	33	35	∞

EJEMPLO TSP2:

Tenemos 4 trabajos que tenemos que realizar en una máquina. En las tablas siguientes se muestra el tiempo de ejecución de los trabajos y el tiempo de preparación de la máquina en función del trabajo que se realice antes. El objetivo es determinar la secuencia de producción para las piezas que minimice el tiempo de ejecución total.

	t1	t2	t3	t4
0	1	2	1	1
t1	∞	3	4	4
t2	1	∞	3	3
t3	2	2	∞	3
t4	2	4	6	∞

1.2.– Introducción a los Métodos Heurísticos

Métodos de resolución para la PLE

Un heurístico es un "procedimiento simple, a menudo basado en el sentido común, que se supone que ofrecerá una buena solución (aunque no necesariamente la óptima) a problemas difíciles, de un modo fácil y rápido". (Zanakins y Evans, 1981)

"Un método heurístico es un procedimiento para resolver un problema matemático bien definido mediante una aproximación intuitiva, en la que la estructura del problema se utiliza de forma inteligente para obtener una buena solución" D. de Werra y otros

- **Algoritmos heurísticos:** No poseen ningún mecanismo que les permita escapar de óptimos locales
- **Algoritmos metaheurísticos:** A diferencia de los heurísticos tratan de huir de óptimos locales orientando la búsqueda en cada momento dependiendo de la evolución del proceso de búsqueda
- Dado un problema, primero hay que intentar resolverlo óptimamente antes de aplicar un heurístico.
- La calidad de un heurístico se mide aplicándolo a muchas instancias del problema y comparando su solución o bien con las soluciones óptimas (si se conocen) o con soluciones de otros heurísticos.

Tipos de Algoritmos Heurísticos que estudiaremos

- **Algoritmo Constructivo (completamente) aleatorio**

  Parte de 0 y obtiene soluciones factibles añadiendo en cada paso añade un elemento. No emplea ninguna característica del problema que favorezca la calidad (valor de la función objetivo) de la solución.

- **Algoritmo Constructivo inteligente**

  Parte de 0 y en cada paso añade un elemento hasta completar una solución factible. El elemento se incluye usando alguna característica "favorable para el problema". Se busca que sean lo más deterministas posible (sin aleatoriedad o casi sin ella).

- **Algoritmo Constructivo Aleatorizado**

  Es como el algoritmo anterior, pero incluyendo azar en las elecciones. Por ejemplo, realizando las elecciones entre varios elementos incluyendo azar. De esta manera se obtienen diferentes soluciones con diferentes aplicaciones del algoritmo.

- **Búsqueda Local:** Comienza con una solución factible inicial, y estudia otras 'cercanas' realizando cambios pequeños en la misma. De entre las estudiadas escoge la mejor y la toma como nueva solución de partida.

Codificación

- Es el 1er paso antes de cualquier algoritmo/Excel. Es un paso **esencial**.
- Es una representación de una solución, normalmente factible.
- Tiene como base los "elementos principales" del problema.
- Tiene que estar completamente definido cómo obtener una solución del problema a partir de una codificación, y viceversa.
- Las técnicas a aplicar (algoritmos constructivos, búsquedas locales etc.) depende de la representación escogida.
- Ejemplos de codificación
  - Un vector de 0's y 1's
  - Una permutación de números de 1 a n
- En ocasiones son necesarios dos o más vectores para la codificación

1.3.– Métodos Constructivos

¿Qué es un método constructivo?

Los métodos constructivos son procedimientos iterativos que, en cada paso, añaden un elemento hasta completar la solución. La forma de escoger ese elemento determina el algoritmo (completamente aleatorio, inteligente o aleatorizado). Son métodos basados en el elemento esencial del problema. En el caso del algoritmo constructivo inteligente, para escoger el elemento a añadir, se utiliza alguna característica "favorable para el problema".

Elemento esencial en el TSP**: Ciudades.**

Codificación: Permutación (sabiendo que de la última ciudad se vuelve a la 1ª)

Ejemplo – Métodos constructivos inteligentes para el TSP

Algunos métodos constructivos inteligentes desarrollados para el TSP:

- Heurísticos del Vecino más Próximo
- Usar las aristas más pequeñas

Ejemplo - Constructivos para el TSP
Esquema básico

Idea del algoritmo constructivo:

Comienza por un vértice cualquiera y añade en cada paso un vértice no seleccionado al momento.

Algoritmo Constructivo

1. Escoger un vértice aleatoriamente, i. Eliminarlo de V.
2. Comenzar un tour T con el vértice i.
3. Mientras V $\neq\emptyset$
   2.1. Escoger un vértice j $\in$ V.
   2.2. Añadimos j al tour T después del vértice anterior.
   2.3. Eliminamos j de V. Llamamos i a j.
   2.4. Añadimos a la función objetivo el coste c_{ij}.
4. Añadimos a la función objetivo el coste entre el último vértice añadido y el 1º.
5. Proporcionamos el tour y la función objetivo.

Ejemplo - Constructivos para el TSP
Aleatorio, Inteligente, Inteligente aleatorizado

Algoritmo Constructivo (completamente) Aleatorio

2.1. Escoger el vértice j $\in$ V aleatoriamente.

Descripción del algoritmo del vecino más próximo:

Comienza por un vértice cualquiera y añade en cada paso **el nodo más cercano** al último seleccionado. El vértice que añadimos no puede haber sido asignado previamente.

En este caso la característica favorable es que las ciudades estén cerca.

Algoritmo Constructivo Inteligente

2.1. Escoger el vértice j $\in$ V con menor c_{ij}.

Algoritmo Constructivo Inteligente Aleatorizado

2.1.1. Calcular los dos vértices de V con menores c_{ij}.
2.1.2. Escoger aleatoriamente uno de los dos y llamarlo j.

Ejemplo - Constructivos para el TSP
Vecino más próximo

Descripción del algoritmo del vecino más próximo:

Comienza por un vértice cualquiera y añade en cada paso el vértice más cercano al último seleccionado.

En este caso la característica favorable es que las ciudades estén cerca.

	C1	C2	C3	C4	C5
C1	999	16	63	21	20
C2	57	999	40	46	69
C3	23	11	999	55	53
C4	71	53	58	999	47
C5	27	79	53	35	999

Iteración 1: Elegimos un nodo por ejemplo el 2. Solución: (2,…)

Iteración 2: Elegimos el nodo más cercano al último no seleccionado previamente. Solución: (2,3…)

Iteración 3: Elegimos el nodo más cercano al último no seleccionado previamente. Solución: (2,3,1…)

Iteración 4: Elegimos el nodo más cercano al último no seleccionado previamente. Solución: (2,3,1,5…)

Iteración 5: Elegimos el nodo más cercano al último no seleccionado previamente. Solución: (2,3,1,5,4).
Iteración 6: Añadimos el nodo inicial. Solución: (2,3,1,5,4,2).
Evaluación = 171

Ejemplo - Constructivos para el TSP
Usar las aristas más pequeñas

DESCRIPCION:
Buscar las aristas más pequeñas que unan vértices que podemos unir.

Algoritmo común para constructivo aleatorio e inteligente

1. Ordenar las aristas en una lista L.
2. Mientras no se haya creado un tour completo
    - 2.1. Escoger de L la primera arista (i,j).
    - 2.2. Estudiar añadirla al tour creado.
    - 2.3. Si se ha creado un subtour, eliminar la arista de L y volver a 2.1.
    - 2.4. En otro caso, actualizar el tour con la ciudad o ciudades correspondientes, y eliminar la arista de la lista. Eliminar de L también todas las aristas que salgan de i o lleguen a j.

3. El tour obtenido es la solución propuesta.

Constructivo aleatorio: 1. Ordenarlas aleatoriamente.
Constructivo inteligente: 1. Ordenarla de menor a mayor coste.
Ejemplo:
L = { (3,2) [11], (1,2) [16], (1,5) [20], (1,4) [21], (3,1) [23], …}
Añadiríamos (3,2), (1,2) NO, (1,5), (1,4) NO, (3,1) NO, (5,1) NO, (5,4), (2,3) NO, (2,4) NO, …, Se obtiene la permutación (1 5 4 3 2 1), eval = 181

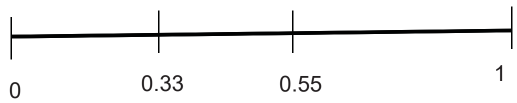
Mejoras: Introducir el azar en el procedimiento

Se pueden calcular muchas soluciones y así obtener mejores resultados, a costa de más tiempo computacional. Son apropiados para metaheurísticos, donde se necesitan varias soluciones distintas de cierta calidad.

- **Selección al Azar Restringida:** Por ejemplo de entre los 3 mejores elegir uno aleatoriamente.
- **Selección Probabilística (método de la ruleta, roulette-Wheel):**

Más probabilidad el mayor, calculamos la suma

valor	probabilidad
30	30/90 =0.33
20	20/90 = 0.22
40	40/90 = 0.45



Se elige un número al azar en el intervalo [0,1]. Por ejemplo 0.35. Como cae en el segundo trozo: elegimos el segundo dato.

Más probabilidad el menor, calculamos el máximo +1.

valor	Valor Opuesto	Probabilidad
20	41-20=21	21/33 = 0.64
30	41-30=11	11/33 = 0.33
40	41-40=1	1/33 = 0.03

Se elige un número al azar en el intervalo [0,1]. Por ejemplo 0.4. Como cae en el tercer trozo: elegimos el primer dato.

Constructivos inteligente aleatorizado para el TSP.

Algoritmo Constructivo Inteligente Aleatorizado

2.1. Escoger el vértice j $\in$ V con el método de la ruleta, usando el coste (a menor coste mayor probabilidad)

EJEMPLO:

Resolver con la ayuda del Excel el ejemplo proporcionado con los siguientes métodos y representando las soluciones como permutaciones:

- El algoritmo constructivo aleatorio (calcular varias soluciones)
- El algoritmo constructivo del vecino más próximo.
- El algoritmo constructivo aleatorizado.

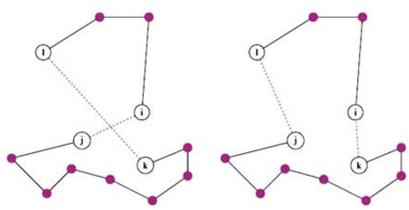
Comparar los métodos.

1.4.– Búsquedas Locales

¿Qué es una búsqueda local?

- Un proceso iterativo que empieza en una solución y la mejora realizando modificaciones locales.
- Básicamente empieza con una solución inicial y busca en su vecindad una solución mejor. Si la encuentra, reemplaza la solución por la nueva y continua el proceso, hasta que no es capaz de mejorar la solución actual.
- La **vecindad** o **conjunto de soluciones vecinas** son todas las soluciones factibles del problema que sean “vecinas” de la solución actual S. La definición de qué es una **solución vecina** depende del problema.
- **Movimiento**: Se denomina de esta forma al cambio que se efectúa en la solución para transformarla en otra.

$$S_1 \xrightarrow{M1} S_2 \xrightarrow{M2} S_3 \xrightarrow{M3} S_4 \xrightarrow{M4} S_5$$

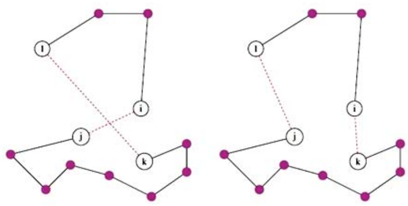


¿Qué es una búsqueda local?

Hay varias formas principales de realizar esta búsqueda en el vecindario, veremos dos:

- **Búsqueda de la mejor solución (exhaustiva)**: **<u>Se evalúan todas las soluciones del vecindario</u>**. Si ninguna tiene un coste estrictamente mejor que la solución actual, el algoritmo se detiene. En caso contrario, el algoritmo continúa aplicándose sobre la mejor solución encontrada hasta el momento.

- **Búsqueda de una solución mejor (no exhaustiva)**: Se evalúan según algún orden definido las soluciones del vecindario. **<u>Cuando se encuentre una solución que tenga un coste mejor que la solución actual el algoritmo para y no evalúa el resto de soluciones vecinas</u>**. Si no se encuentra una con un coste estrictamente mejor que la solución actual, el algoritmo se detiene. En caso contrario, el algoritmo continúa aplicándose sobre la mejor solución encontrada hasta el momento.



¿Qué es una búsqueda local?

- La base de muchos métodos heurísticos usados en problemas de optimización.
- Método determinístico y sin memoria.
- El algoritmo se detiene cuando la solución no puede ser mejorada. La solución encontrada se denomina **óptimo local** respecto al entorno definido.
- Principal ventaja: Encuentra soluciones muy rápidamente.
- Principal desventaja: queda atrapada fácilmente en óptimos locales.

Ejemplo para el TSP:
Vamos a ver el método de Inserción.
Sin embargo, la mayoría de las búsquedas locales para el TSP se basan en movimientos donde se eliminan arcos del tour actual y se reconecta el tour usando nuevos arcos.
- 2-Opt
- 3-Opt

Ejemplo – Búsqueda Local para el TSP Inserción

MOVIMIENTO: Seleccionar un nodo del tour y probar a ponerlo en cualquier otro lugar

El problema tiene n nodos, la codificación es una permutación de 1 a n.
Estudiamos dos versiones.
Versión Completa

Algoritmo
1. Iniciar mejora = 1. La solución actual la llamamos S.
2. Mientras mejora =1 hacer
    2.1. mejora = 0
    2.2. Desde i = 1 hasta n, hacer
        2.2.1 Consideramos el nodo de la posición i-ésima. Lo llamamos J
        2.2.2 Probamos a insertar este nodo en el resto de posiciones y nos quedamos con la mejor solución, S´(i).
    2.3. Consideramos S’ como la mejor solución de las S’(i).
    2.4. Si f(S´) < f(S)
        2.4.1. S = S’. (es decir, realizamos el movimiento correspondiente).
        2.4.2. mejora = 1.

Ejemplo – Búsqueda Local para el TSP Inserción

MOVIMIENTO: Seleccionar un nodo del tour y probar a ponerlo en cualquier otro lugar

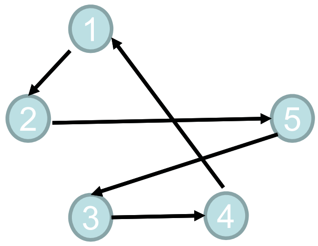
El problema tiene n nodos, la codificación es una permutación de 1 a n, V(i) es el nodo en la posición i-ésima.
Estudiamos dos versiones.
Versión Simplificada

Algoritmo
1. Iniciar mejora = 1. La solución actual la llamamos S.
2. Mientras mejora =1 hacer
    2.1. mejora = 0
    2.2. Escoger aleatoriamente una posición **j** entre 1 y n
    2.3. Desde i = 1 hasta n, hacer
        2.3.1 Insertamos el nodo V(j) en la posición i (salvo si i = j).
        2.3.2 Llamamos S´ a la nueva solución y calculamos f(S’).
    2.4. Si f(S´) < f(S)
        2.4.1. S = S’. (es decir, realizamos el movimiento correspondiente).
        2.4.2. mejora = 1.

Ejemplo – Búsqueda Local para el TSP Inserción

MOVIMIENTO: Seleccionar un nodo del tour y probar a ponerlo en cualquier otro lugar

Iteración 1
Solución Actual: (1,2,5,3,4,1) C=264
Vértice (elegido al azar) para aplicar BL: 1
Vecindario del 1: (2,1,5, 3,4,2) C=238
(2,5,1, 3,4,2) C=267
(2,5,3,1,4,2) C=219



*En una **búsqueda local exhaustiva** generaríamos todo el vecindario y nos quedaríamos con la mejor solución (la de C=219). En una **búsqueda no exhaustiva** iríamos generando y nos quedaríamos con la primera mejor que encontráramos (la de C=238 en este caso).*

Iteración 2
Solución Actual: (2,5,3,1,4,2) C=219
Vértice (elegido al azar) para aplicar BL: 3
Vecindario del 3: (3,2,5,1,4,3) C=186 Mejora la solución por tanto actualizamos la mejor solución y seguimos.

Ejemplo – Búsqueda Local para el TSP Inserción

Iteración 3
Solución Actual: (3,2,5,1,4,3) C=186
Vértice (elegido al azar) para aplicar BL: 2
Vecindario del 2: (2,3,5,1,4,2) C=194
(3,5,2,1,4,3) C=268
(3,5,1,2,4,3) C=200
Como no mejoramos pasamos a estudiar la vecindad de otro vértice.
Vértice (elegido al azar) para aplicar BL: 4
Vecindario del 4: (3,4,2,5,1,3) C=267
(3,2,4,5,1,3) C=194
(3,2,5,4,1,3) C=200
Como no mejoramos pasamos a estudiar la vecindad de otro vértice.
Vértice (elegido al azar) para aplicar BL: 5
Vecindario del 5: …………………
.

Ejemplo – Búsqueda Local para el TSP
Comparación de Métodos

Heurístico	Desviación del óptimo	Tiempo de ejecución (segundos) Para el pr2392
Vecino más próximo	24.2%	15.3
Vecino más próximo con subgrafo candidato (k=10, s=4)	18.6%	0.3
Inserción más cercana (k=3)	20%	
Inserción más lejana (k=3)	9.9%	35.4
Inserción más barata (k=3)	16.8%	
Inserción aleatoria (k=3)	11.1%	
2-Opt	8.3%	0.25
3-Opt	3.8%	85.1
Lin y Kernighan	1.9%	27.7 V1 / 74.3 V2

Jünger, M., Reinelt, G., Giovanni, R. (1995)
30 instancias de la TSPLIB con tamaños entre 105 a 2392 y de los que se conoce la solución óptima
Porcentaje de desviación del óptimo = $\frac{c_H - c_{opt}}{c_{opt}} \times 100$

Ejemplo – Búsqueda Local para el TSP
Visualización Búsquedas Locales

Búsqueda local 2-Opt para el TSP:
https://www.youtube.com/watch?v=UGGPZnAUjPU

Búsqueda local 3-Opt para el TSP:
https://www.youtube.com/watch?v=fByHMYjx1Gg

Diversos procesos de búsqueda para el TSP:
https://www.youtube.com/watch?v=q6fPk0--eHY
https://www.youtube.com/watch?v=SC5CX8drAtU

1.5.– Aplicaciones

Para todos los problemas que se introducen en esta sección:

1. Calcula una codificación que represente soluciones.
2. Crea un algoritmo aleatorio que construya soluciones factibles.
   - Descríbelo.
   - Obtén una solución con él.
3. Crea un algoritmo constructivo.
   - Descríbelo.
   - Obtén una solución con él.
4. Crea un algoritmo constructivo aleatorizado.
   - Descríbelo.
   - Obtén dos solución con él.
5. Crea una búsqueda local.
   - Descríbela.
   - Aplícala sobre la solución obtenida con el constructivo aleatorio.

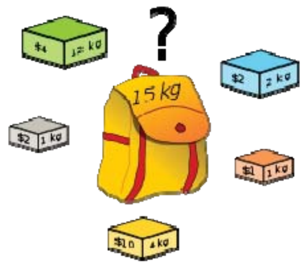
6. Modeliza el problema matemáticamente y resuélvelo

Nota: Utiliza Excel para ayudarte con los cálculos de los apartados 2-5 si es posible.

Ejemplo 1 – Problema de la Mochila (Knapsack problem)

N objetos a transportar en una mochila.
Cada objeto j tiene un volumen a_j y un beneficio c_j.
La mochila tiene una capacidad b.
Se trata de determinar los objetos a llevar en la mochila de forma que el beneficio sea máximo.

$$\max c_1 x_1 + c_2 x_2 + \ldots + c_n x_n$$
$$s.a.\ a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \leq b$$
$$x_i \in \{0,1\} \qquad i=1,\ldots n$$



Ejemplo 1 – Problema de la Mochila (Knapsack problem)

Aplicaciones:
- Situaciones reales donde es necesario acomodar artículos de diferentes dimensiones en un espacio reducido:
  - Uso de contenedores en las aduanas, donde se requiere enviar ítems de diferentes pesos, tamaños y valores de beneficio.
  - Almacenaje de contenedores.
  - Abastecer vehículos de transporte y entrega de productos de diferentes tamaños que deben ser colocados en múltiples compartimentos de igual o diferente tamaño.
- Selección de proyectos.
- En la solución de problemáticas donde es necesario detectar patrones de corte.
- En situaciones donde se problemas de distribución de carga (física, eléctrica, etc.).

Ejemplo 1 – Problema de la Mochila (Knapsack problem)

Consideremos una empresa que puede realizar 7 proyectos, cada uno de los cuales con una estimación del beneficio, el coste y los requerimientos de personal. La información se resume en la tabla mostrada. Por ejemplo, el Proyecto 1 requiere de 120 profesionales para ser realizado, con una inversión inicial de 15 millones de dólares y aportará un beneficio de 2 millones de dólares.

Asumiremos que la empresa dispone de 130 profesionales y un presupuesto para inversión de 30 millones de dólares. Los proyectos 3 y 6 son excluyentes, es decir, sólo uno de los 2 puede ejecutarse.

Proyecto	Personal	Coste	Beneficio
1	120	15.000.000	2.000.000
2	25	5.000.000	1.000.000
3	15	6.000.000	900.000
4	60	10.000.000	1.200.000
5	8	3.200.000	500.000
6	12	4.000.000	1.000.000
7	20	12.000.000	2.500.000

Ejemplo 2 - Asignación

La encargada de recursos humanos de una gran empresa debe escoger 5 trabajadores de entre 10 trabajadores para cubrir 5 puesto nuevos que va a crear la empresa. Tras realizarles un test de conocimientos obtiene una calificación que le indica el rendimiento de cada trabajador en cada puesto de trabajo. Los resultados del test son los que se muestran en la tabla (10: rendimiento alto, 1: rendimiento bajo). La encargada debe decidir qué trabajador asigna a cada puesto para maximizar el rendimiento total.

	Puesto 1	Puesto 2	Puesto 3	Puesto 4	Puesto 5
Trabajador 1	9	9	9	9	1
Trabajador 2	7	8	2	1	8
Trabajador 3	2	2	4	9	9
Trabajador 4	4	8	2	5	8
Trabajador 5	5	9	8	2	1
Trabajador 6	1	2	3	3	6
Trabajador 7	7	3	4	9	9
Trabajador 8	2	4	8	8	8
Trabajador 9	9	10	4	9	10
Trabajador 10	7	6	4	3	2

Ejemplo 3 – Problema de Localización

El sistema de salud del estado de Florida quiere construir una serie de clínicas de primeros auxilios, de modo que den servicio a todo el estado. Para ello han dividido el estado en 8 regiones y consideran 6 posibles localizaciones para su construcción. La siguiente tabla muestra en el encabezamiento el nombre de las 6 localidades, y hay una fila por cada región del estado. Las cruces indican que una clínica en la localidad daría servicio a la correspondiente región. La tabla también muestra el coste, en miles de dólares, de construir una clínica en cada localidad.

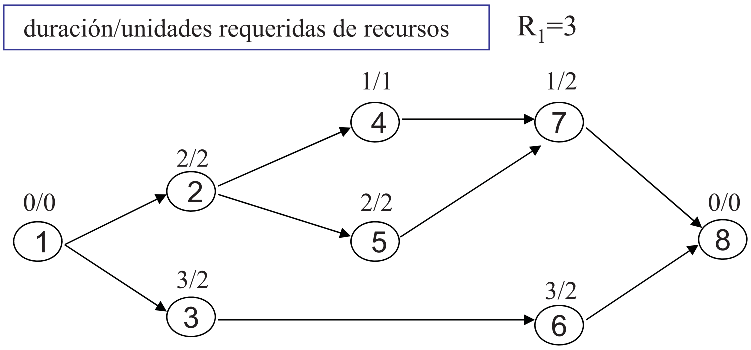
Región	Sanford	Altamonte	Apopka	Casselberry	Maitland	Goldenrod
1	X		X			X
2	X	X		X	X	
3		X		X		X
4			X		X	
5	X	X				X
6			X		X	
7				X	X	
8		X	X			X
Coste	450	650	550	500	525	700

Ejemplo 4 – Asignación de recursos

Una empresa tiene que realizar 10 tareas que consumen recursos de 6 tipos. En la tabla se muestra las cantidad de recurso que consume cada tarea, la disponibilidad de cada recurso (última fila) y el beneficio que se obtendría si se realizara la tarea. Se trata de decidir qué tareas deben ejecutarse para maximizar el beneficio.

	1	2	3	4	5	6	Beneficio
1	2	-	4	1	-	1	2
2	2	2	3	-	-	-	4
3	3	-	2	4	-	-	6
4	6	-	6	6	6	6	7
5	3	-	3	3	6	7	4
6	2	-	-	2	-	1	3
7	3	3	3	3	5	-	6
8	-	-	-	-	-	-	2
9	-	-	-	5	5	5	10
10	-	-	-	5	1	3	2
	10	20	30	25	18	10	

Ejemplo 5 – Secuenciación de proyectos



Supongamos un proyecto formado por 8 actividades (la primera y la última indican principio y fin del proyecto). Cada una de las actividades requiere para su realización de un número de operarios. Se dispone de 3 operarios durante todo momento en la ejecución del proyecto.

Bibliografía

- Jünger, M., Reinelt, G., Giovanni, R. (1995), The traveling salesman problem, Chapter 4, Handbooks in Operations Research and Management Science, Volume 7, Pages 225–330, doi:10.1016/S0927-0507(05)80121-5
- Martí, R. (2003). Procedimientos metaheurísticos en optimización combinatoria. Matematiques, vol. 1, No 1, p. 3-62.
- Taha, H. A. Investigación de operaciones. Pearson, 9ª edición , 2012.
- Hillier, F. S., Lieberman, G.J., Investigación de operaciones, 10ª edición, 2015.
- Search Methologies, Introductory Tutorials in Optimization and Decision Support Techniques, Burke & Kendall 2014, Springer