Algoritmos Heurísticos en Optimización Combinatoria

Rafael Martí

Departamento de Estadística e Investigación Operativa

Universitat de València

Algoritmos Heurísticos

Introducción

- Problemas Combinatorios
- Calidad de los Algoritmos
- El Problema del Viajante

Procedimientos

- Métodos Constructivos
- Métodos de Búsqueda Local
- Métodos Combinados

Problema de Optimización Combinatoria

- Conjunto base: $E = \{1, 2, ..., n\}$
- Conjunto de soluciones factibles $F \subseteq 2^E$.
- Función objetivo $f: 2^E \rightarrow R$.
- En la versión de minimización buscamos una solución óptima $S^* \in F$, tal que $f(S^*) \leq f(S) \quad \forall S \in F$.

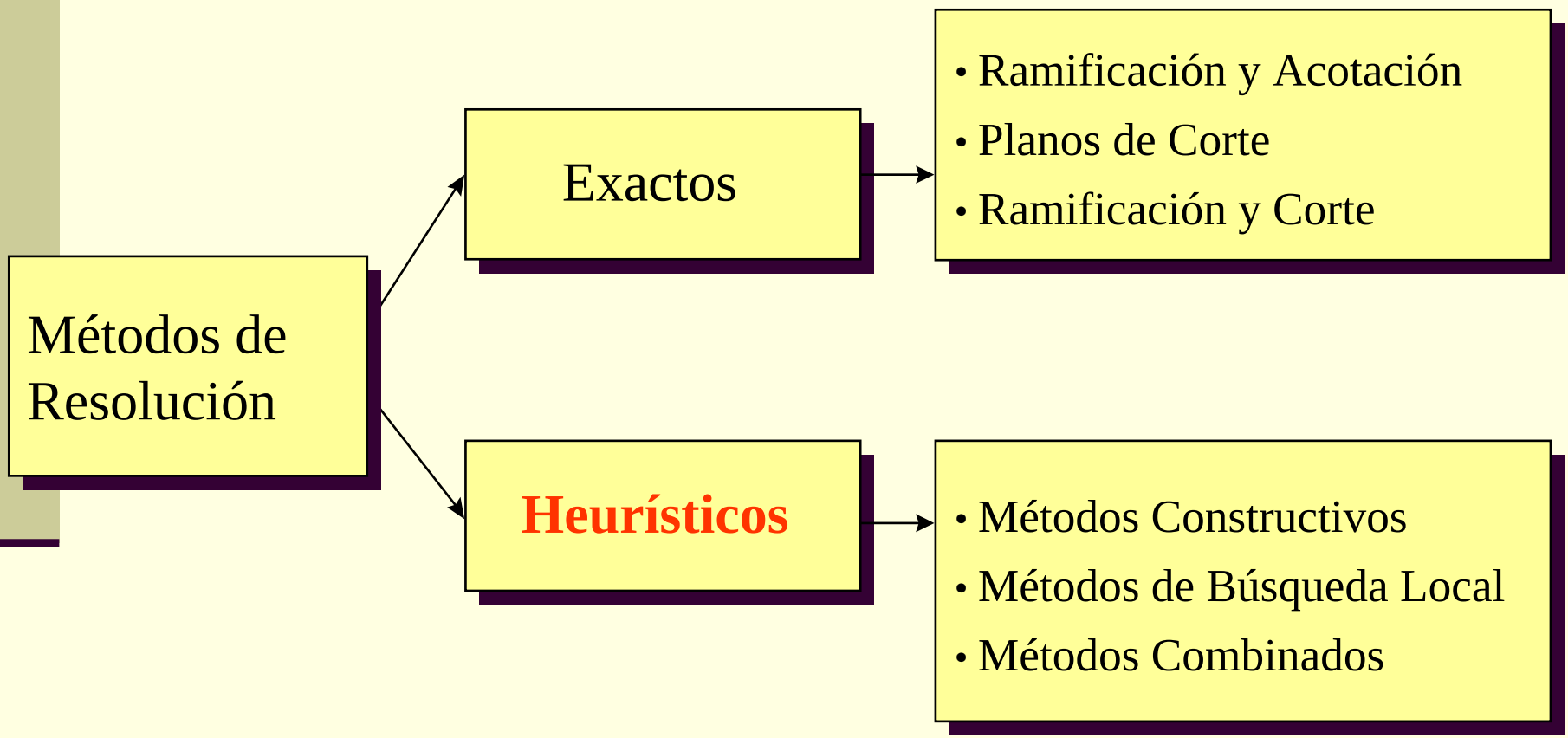
Problemas

- Mochila
- Cubrimiento de Conjuntos
- Empaquetado de Conjuntos
- Partición de Conjuntos
- Viajante
- Asignación Cuadrática
- Asignación Generalizada
- Ordenación Lineal

Resolución de Problemas Combinatorios

- Un **método exacto** proporciona una solución óptima del problema.
- Un **método heurístico** o aproximado proporciona una buena solución del problema no necesariamente óptima.
  - *"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
  - *"Un heurístico es una técnica que busca buenas soluciones con un tiempo de computación razonable sin garantizar la optimalidad"* C.R. Reeves

Una Clasificación



Calidad del Algoritmo Heurístico

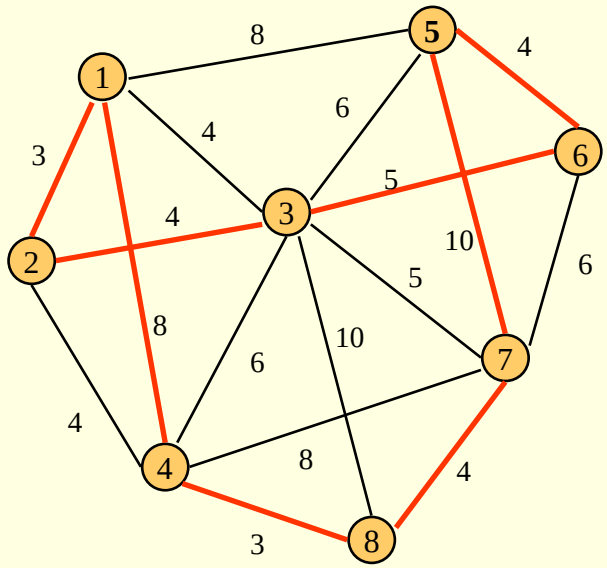
- Un buen algoritmo heurístico debe ser:

Eficiente, Bueno y Robusto

- Los procedimientos para medir la calidad de un algoritmo son:
  - Comparación con la solución óptima
  - Comparación con una cota
  - Comparación con un método exacto truncado
  - Comparación con otros heurísticos
  - Análisis del peor caso

El Problema del Viajante (TSP)

"Un viajante de comercio ha de visitar n ciudades, comenzando y finalizando en su propia ciudad. Conociendo el coste de ir de cada ciudad a otra, determinar el recorrido de coste mínimo."



Formalmente

- Sea un grafo G=(V,A,C) donde:
  - V es el conjunto de vértices
  - A es el de aristas
  - C=(c_{ij}) es la matriz de costes: c_{ij} es el coste (distancia) de la arista (i, j).
- Podemos considerar, sin pérdida de generalidad, G completo.
- Un **tour** (Ciclo Hamiltoniano) es un ciclo que pasa exactamente una vez por cada vértice del grafo.
- El TSP consiste en determinar un tour de coste mínimo.

Elección del Problema

- Es uno de los que mas interés ha suscitado en Investigación Operativa
- Sus soluciones admiten una doble interpretación: mediante grafos y mediante permutaciones.
- Dada su gran dificultad (NP-duro), la gran mayoría de las técnicas de resolución han sido probadas en él.
- Resulta muy intuitivo y con un enunciado muy fácil de comprender.

Algoritmos Heurísticos

Métodos de Resolución

- **Métodos Constructivos**
- **Métodos de Búsqueda Local**
- **Métodos Combinados**

Métodos Constructivos

- Los métodos constructivos son procedimientos iterativos que, **en cada paso añaden un elemento** hasta completar una solución. Usualmente son métodos deterministas y están basados en seleccionar, en cada iteración, el elemento con mejor evaluación.
- Los más destacados para el TSP son:
  - Heurísticos del Vecino más Próximo
  - Heurísticos de Inserción
  - Heurísticos basados en Árboles Generadores
  - Heurísticos basados en Ahorros

Heurístico del "*Vecino más Próximo*"

- Añade en cada paso el vértice más cercano al actual.
- Debido a Rosenkrantz, Stearns y Lewis (1977)

Inicialización

Seleccionar un vértice j al azar.
Hacer $t = j$ *y* $W = V \setminus \{j\}$.

Mientras $(W \neq \emptyset)$

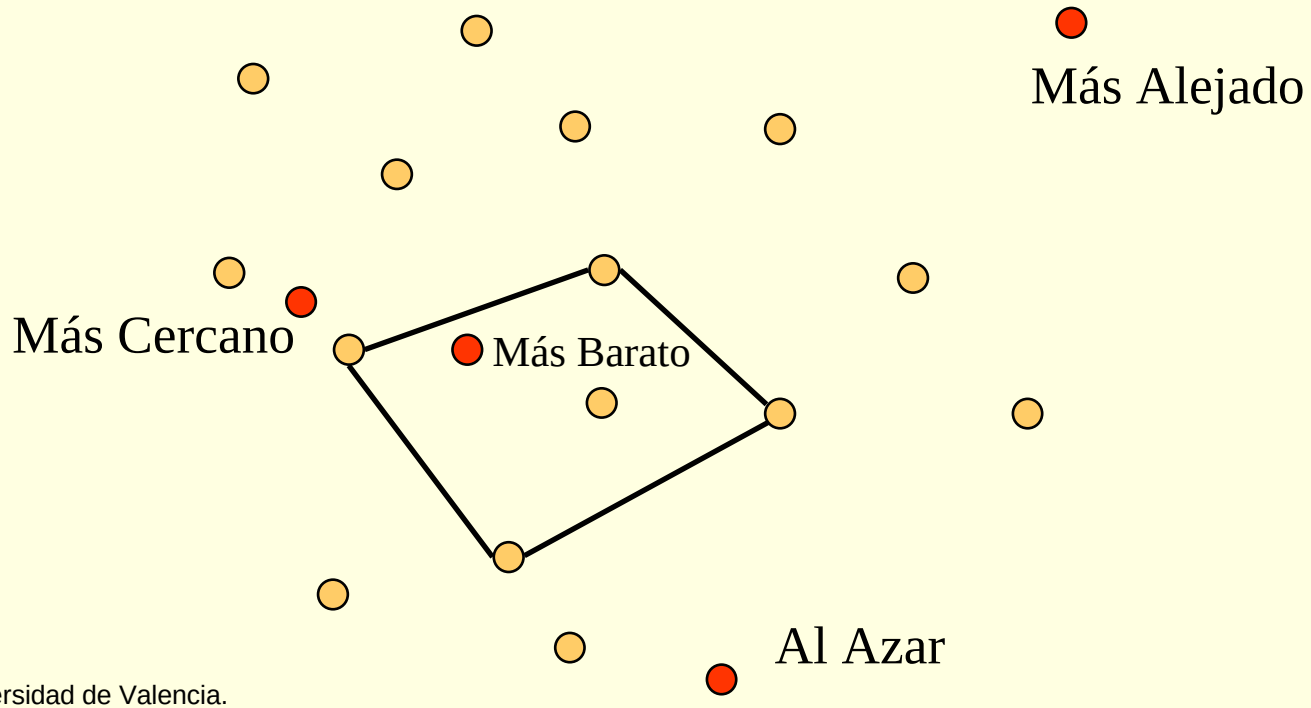
Tomar j *de* $W / c_{tj} = min \{c_{ti} / i \text{ en } W\}$
Conectar t *a* j
Hacer $W = W \setminus \{j\}$ *y* $t = j$.

Eficiencia del Procedimiento

- **Deficiencias**:
  - Una implementación directa sería poco eficiente, al examinar en cada paso todos los vértices.
  - Miopía del algoritmo
- **Mejoras**:
  - **Subgrafo Candidato**: es un subgrafo del grafo original con los n vértices y únicamente **las aristas consideradas "atractivas"** para aparecer en un tour de bajo coste. (*subgrafo de los k vecinos más cercanos)*
  - En cada paso del algoritmo, se comienza por examinar los vértices del subgrafo candidato.
  - Cuando un vértice que no está en el tour está conectado únicamente a *s* (*s*<*k*) o menos aristas del subgrafo candidato se considera que se está quedando aislado. Por ello se inserta inmediatamente en el tour.

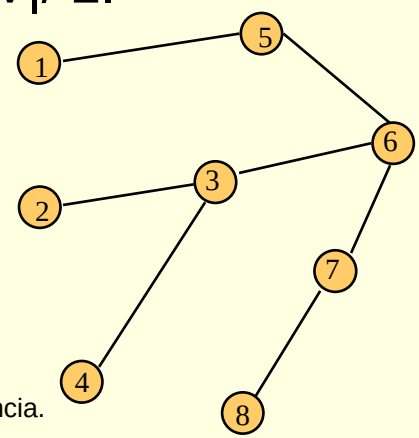
Heurísticos de Inserción

- Extender ciclos que pasan por unos cuantos vértices (*subtours*), insertando un vértice nuevo en cada paso.

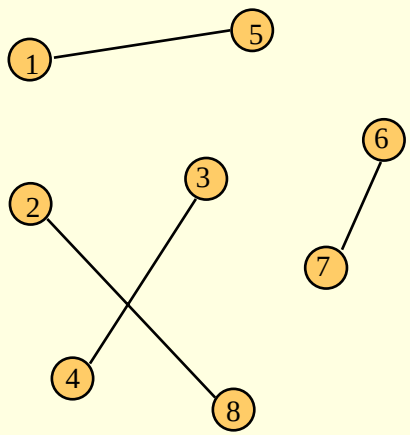


Árboles y Acoplamientos

- Un grafo es **conexo** si todo par de vértices está unido por un camino.
- Un **árbol** es un grafo conexo que no contiene ciclos.
- Un **árbol generador** es un árbol sobre todos los vértices.
- Un **acoplamiento** es un subconjunto M del conjunto de aristas cumpliendo que cada vértice del grafo es a lo sumo incidente con una arista de M.
- Un acoplamiento es **perfecto** si es de cardinalidad máxima e igual a |V|/ 2.

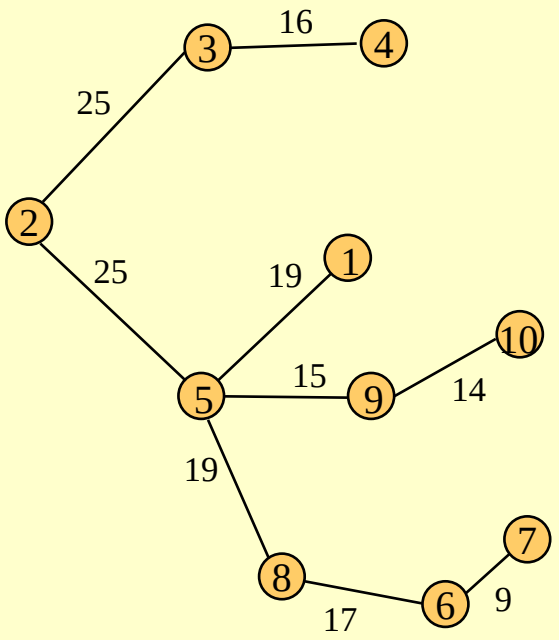


Árbol Generador



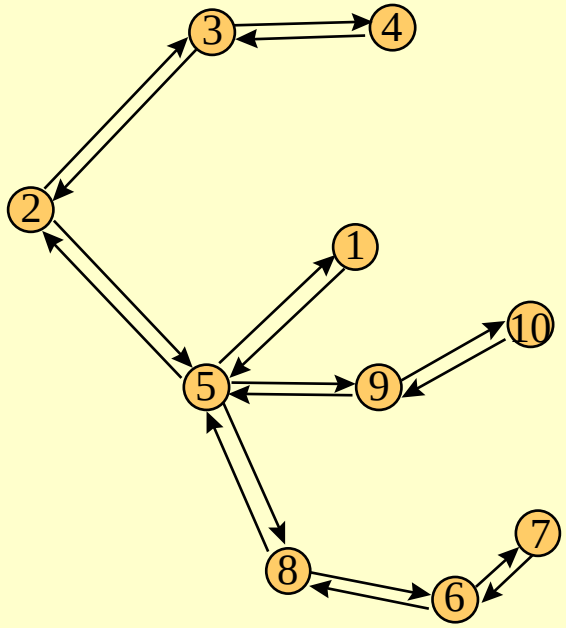
Acoplamiento Perfecto

Heurísticos basados en Árboles Generadores



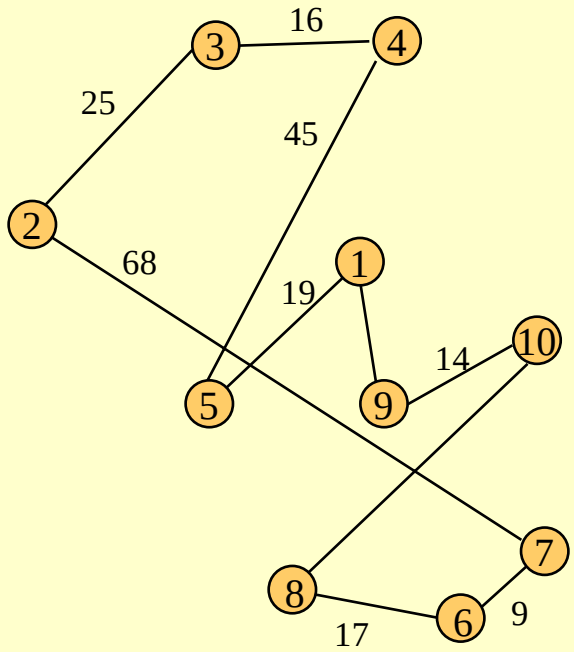
Paso 1

Árbol generador de mínimo peso



Paso 2

Duplicación de Aristas

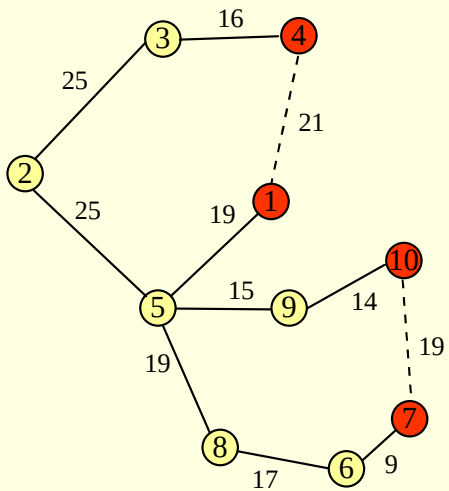


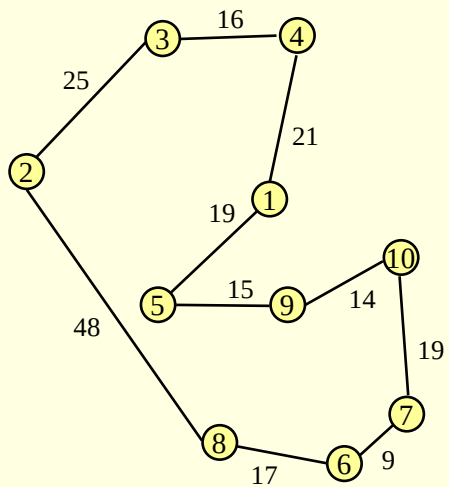
Paso 3

Obtención del Tour

Algoritmo de Christofides

Paso 2: En lugar de duplicar las aristas del árbol se añaden las aristas de un acoplamiento perfecto de mínimo peso sobre los vértices de grado impar del árbol generador.



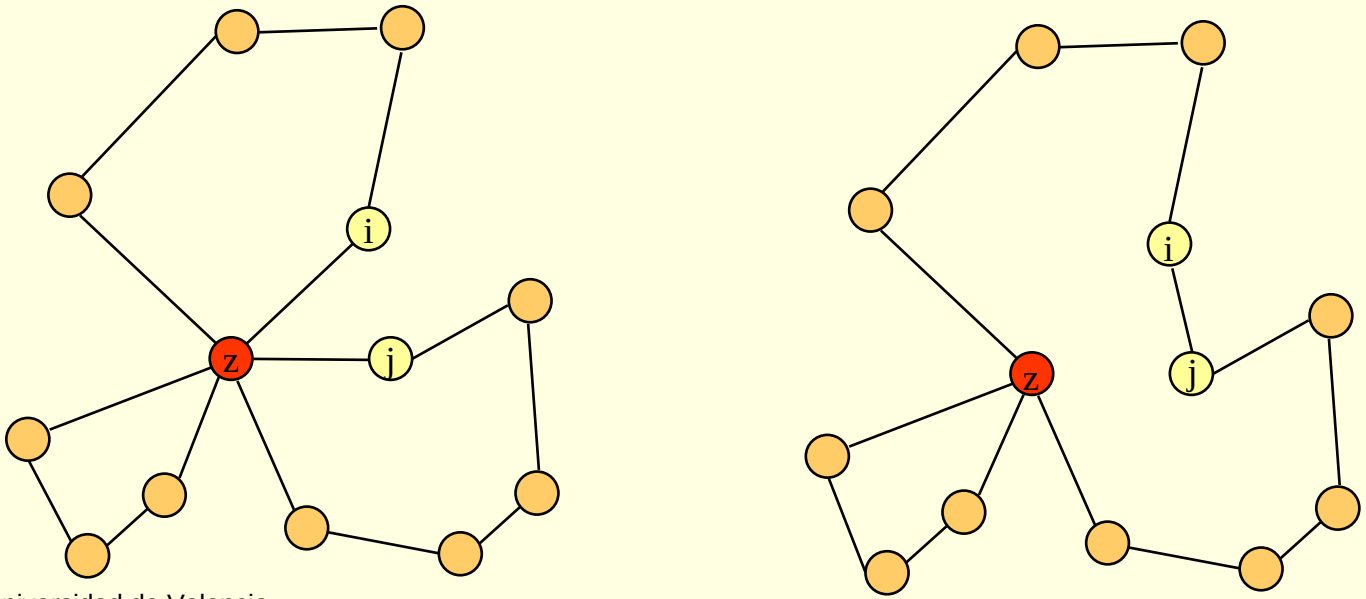


Teorema:

Sobre ejemplos cuya matriz de distancias cumple la desigualdad triangular, el valor de la solución del algoritmo es como mucho 1.5 veces el valor óptimo.

Heurísticos basados en Ahorros

- Combinar sucesivamente subtours hasta obtener un tour. (Clarke y Wright, 1964)
- Los subtours tienen un vértice común llamado **base** (z)



Comparación de los Métodos

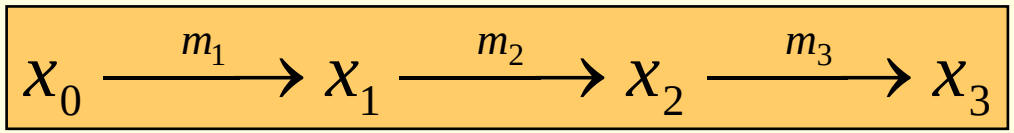
- La librería de dominio público TSPLIB contiene un conjunto de ejemplos con la solución óptima del TSP. (Reinelt , 1991)

Heurístico	**Desviación del Óptimo**	**T. Ejecución** (pr2392)
Vecino más Próximo	18.6%	0.3
Inserción más Alejada	9.9%	35.4
Christofides	19.5%	0.7
Ahorros	9.6%	5.07

Porcentaje de desviación del óptimo: $\frac{c_h - c_{opt}}{c_{opt}} \cdot 100$

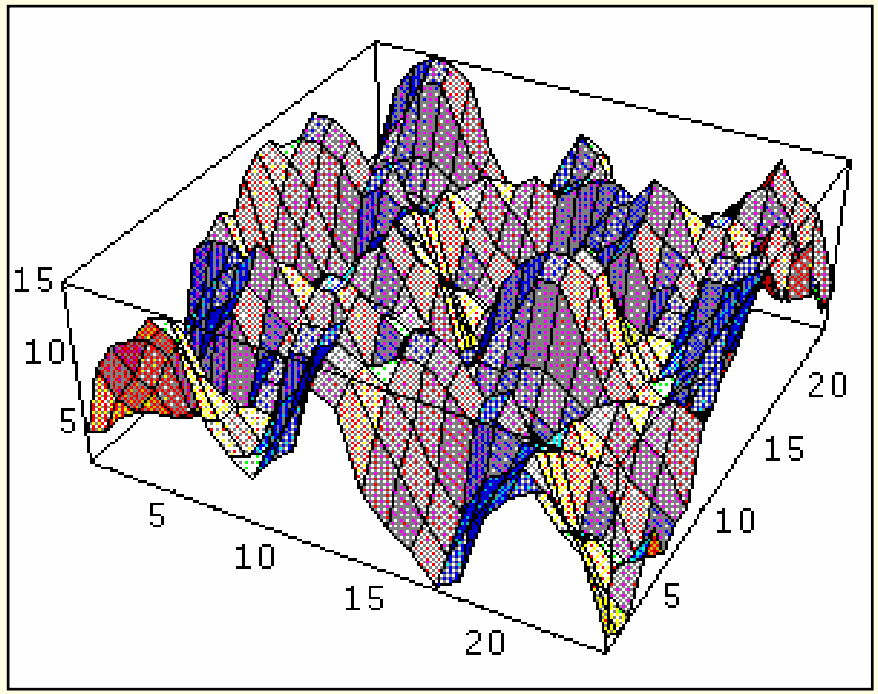
Métodos de Búsqueda Local

- **Definición**: Cada solución *x* tiene un conjunto de soluciones asociadas *N(x)*, que se denomina **entorno** de x.
- **Definición**: Dada una solución *x,* cada solución *x'* de su entorno N(x) puede obtenerse directamente a partir de *x* mediante una operación llamada **movimiento**.
- El método se basa en explorar el entorno de una solución y seleccionar una nueva solución en él (i.e. realizar el movimiento asociado). Desde la nueva solución se explora su entorno y se repite el proceso.



Criterio de Selección

- **Greedy**: Seleccionar la solución con **mejor evaluación** de la función objetivo, siempre que sea mejor que la actual.
- El algoritmo se detiene cuando la solución no puede ser mejorada.
- A la solución encontrada se le denomina ***óptimo local*** respecto al entorno definido.
- Miopía del Método



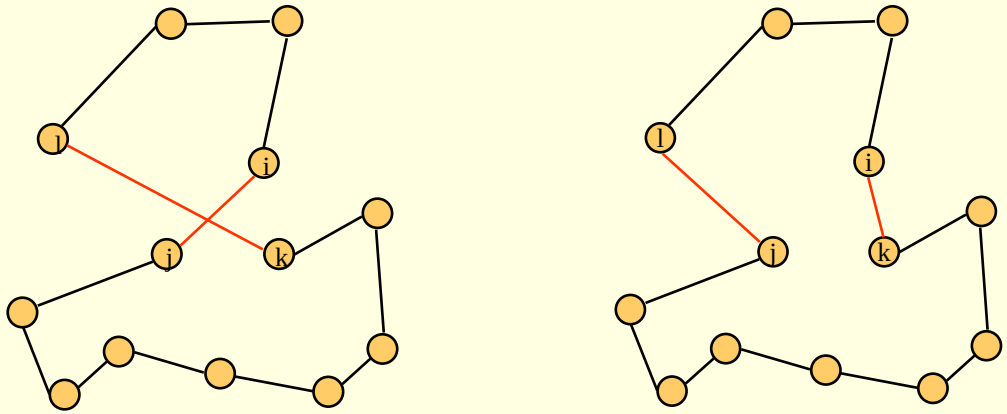
Búsqueda Local en el TSP

- **Definición del Movimiento**
  - 2 intercambio
  - k intercambio
  - Lin y Kernighan

- **Estrategia de Selección**.
  - Greedy
  - Para aumentar la eficiencia se utiliza la estrategia Greedy sobre un subconjunto de candidatos (Subgrafo Candidato).

Procedimiento de 2 Intercambio

- Un **movimiento 2-opt** consiste en eliminar dos aristas y reconectar los dos caminos resultantes, de una manera diferente, para obtener un nuevo ciclo



⌘ Hay una única manera de reconectar los camino formando un tour.

Algoritmo 2-Óptimo

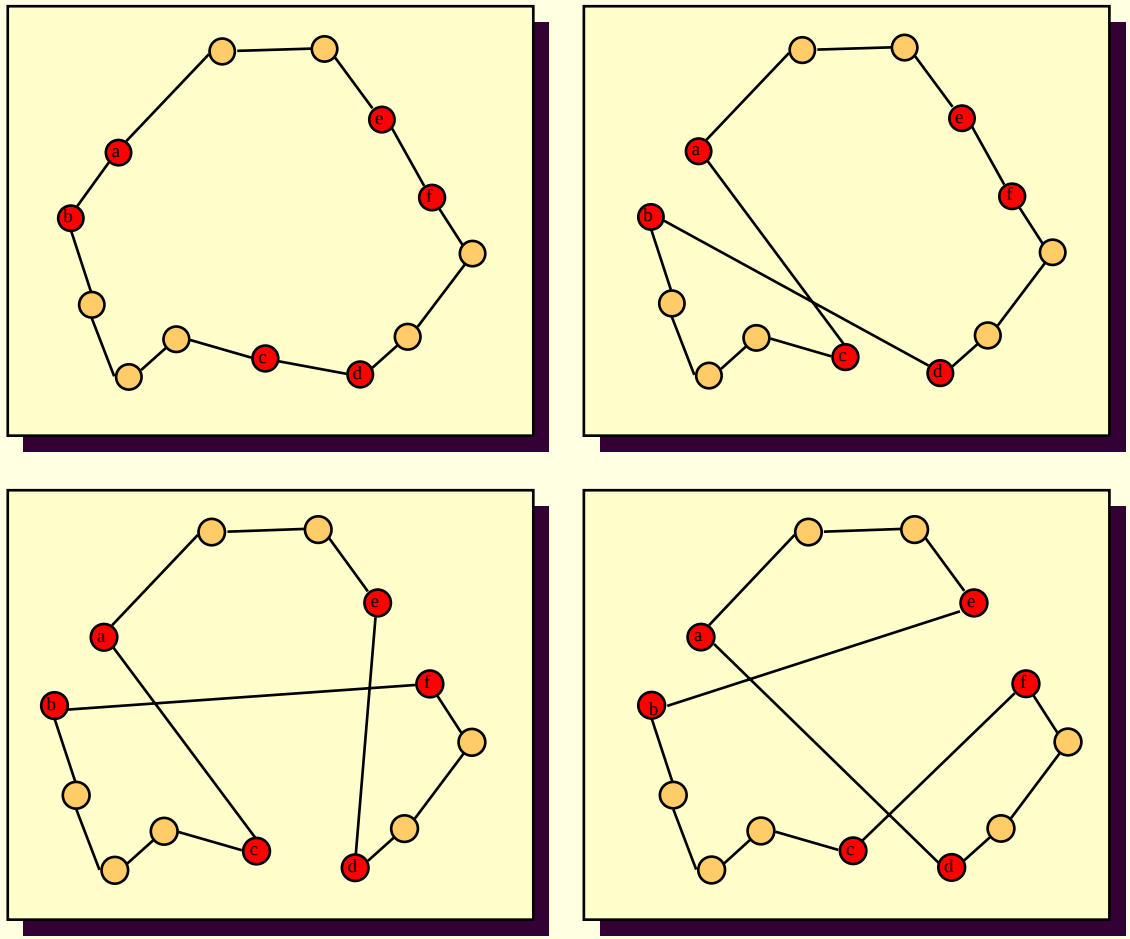
- **Inicialización**
  - Considerar un ciclo Hamiltoniano inicial
  - move = 1
- **Mientras** *(move = 1)*
  - move=0
  - Etiquetar todos los vértices como no explorados.
  - **Mientras** (Queden vértices por explorar)
    - Seleccionar un vértice i no explorado.
    - Examinar todos los **movimientos 2-opt** que incluyan la arista de i a su sucesor en el ciclo.
    - Si alguno de los movimientos examinados reduce la longitud del ciclo, realizar el mejor de todos y hacer move = 1. En otro caso etiquetar i como explorado.

Mejoras

- En una **implementación eficiente** se considera una lista de vértices candidatos a examinar.
  - Cada vez que se examina un vértice i, éste se coloca al final de la lista y los vértices involucrados en el movimiento se insertan en primer lugar.
- Para **reducir el tiempo de computación**, podemos:
  - Exigir que al menos una de las dos aristas añadidas en cada movimiento, para formar la nueva solución, pertenezca al ***subgrafo candidato*.**
  - Interrumpirlo antes de finalizar, dado que en las primeras iteraciones la función objetivo decrece substancialmente, mientras que en las últimas apenas se modifica.

Procedimiento de 3 Intercambio

- Al eliminar 3 aristas hay 8 maneras de reconectar los 3 caminos resultantes.
- A diferencia de los 2-opt, el reconstruir el ciclo aquí es muy costoso.
- El examinar todas las posibilidades representa un esfuerzo en computación enorme.
- Aplicar un búsqueda restringida: Subgrafo Candidato.

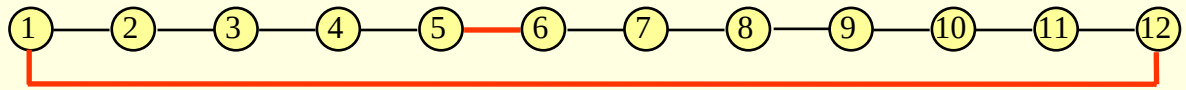


Algoritmo de Lin y Kernighan

- Realiza un movimiento compuesto.
- Cada uno de los movimientos simples puede mejorar o empeorar el valor de la función objetivo.
- **El movimiento global es de mejora**, por lo que no se pierde el control sobre el proceso de búsqueda.
- El mezclar diferentes movimientos altera la estructura de la solución haciendo que la **convergencia al óptimo local** sea lenta e introduce una componente de diversificación.

Un ejemplo: (2-opt, 2-opt, inserción)

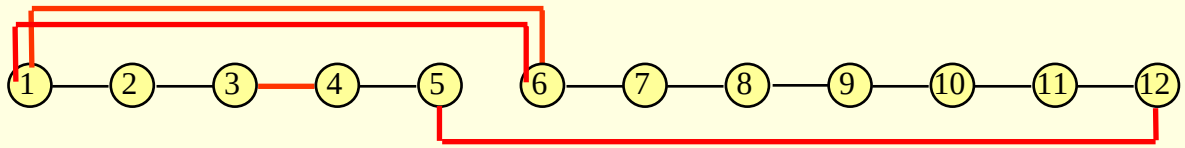
- **Ciclo Inicial**



- **2-opt cambio**

(12,1), (5,6) por

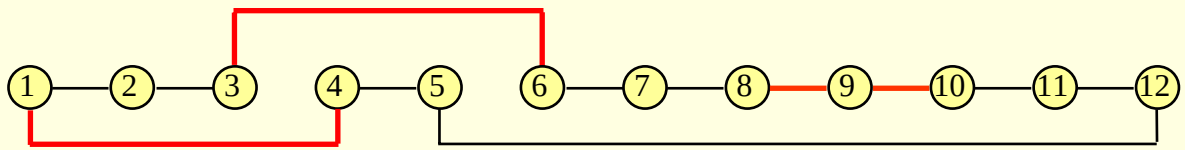
(12,5), (1,6)



- **2-opt cambio**

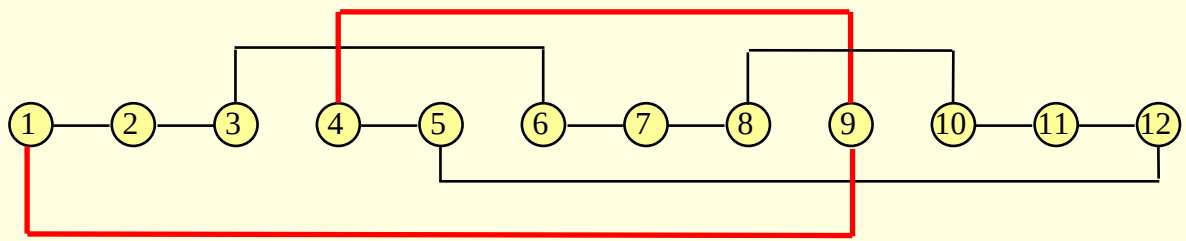
(1,6), (3,4) por

(6,3), (1,4)



- **Inserción**

Insertar el 9 entre el 1 y el 4



Implementaciones

- **Algoritmo 1**
  - Utilizar el **subgrafo candidato** con **6** vecinos más cercanos para cada vértice.
  - Movimientos de hasta 15 movimientos simples 2-opt o inserción.
  - Para el primer movimiento simple únicamente se examinan 3 candidatos.
- **Algoritmo 2**
  - Utilizar el **subgrafo candidato** con 8 vecinos más cercanos para cada vértice.
  - Movimientos de hasta 15 movimientos simples 2-opt o inserción.
  - Para los 3 primeros movimiento simples únicamente se examinan 2 candidatos.

Comparación de los Métodos

Heurístico	**Desviación del Óptimo**	**T. Ejecución** (pr2392)
2-óptimo	8.3 %	0.25
3-óptimo	3.8 %	85.1
Lin y Kernighan 1	1.9 %	27.7
Lin y Kernighan 2	1.5 %	74.3

Métodos Combinados

Método Constructivo + Método de Búsqueda Local

- Variantes y Mejoras

  Introducir el azar en el procedimiento:

  En los métodos de inserción se obtiene mejores resultados en promedio al elegir al azar el vértice a insertar, que al tomar el elemento más cercano (11% frente a 20%)

  - Selección al Azar Restringida
  - Selección Probabilística

Construcción: Selección al Azar Restringida

- Sustituir una elección determinista por una elección al azar de entre un conjunto de buenos candidatos.
  - En cada paso del procedimiento, se evalúan todos los elementos que pueden ser añadidos y **se selecciona un subconjunto con los mejores**. La elección se realiza al azar sobre ese subconjunto de buenos candidatos.
- Estrategias para calcular el subconjunto:
  - Incluir los k mejores elementos.
  - Incluir los elementos cuya evaluación esté por encima de un umbral.
  - Incluir los elementos cuya evaluación sea próxima a la del mejor.

Selección Probabilística

- Considerar las evaluaciones como pesos y utilizar un método probabilístico para seleccionar un elemento.

Elemento	Evaluación	Intervalo
v_1	e_1	$[0,\ e_1[$
v_2	e_2	$[e_1,\ e_1 + e_2[$
…	…	…
v_k	e_k	$\left[\sum_{i=1}^{K-1} e_i, \sum_{i=1}^{k} e_i\right]$

Se genera un número *a* al azar entre 0 y $\sum_{i=1}^{k} e_i$, y se selecciona el elemento correspondiente al intervalo que contiene a *a*.

Algoritmo Combinado "Aleatorizado"

- ***Inicialización*** (i = 0)
  - Obtener una solución con el algoritmo constructivo.
    (Sea c* el coste)
- ***Mientras*** *(i <* MAX_ITER*)*
  - Obtener una solución x(i) con el **algoritmo constructivo aleatorizado**.
  - Aplicar el algoritmo de búsqueda local a x(i).
    - Sea x*(i) la solución obtenida y S*(i) su valor.
  - Si (S*(i) mejora a c*)
    - Hacer c* = S*(i) y almacenar x*(i)
  - i = i +1

Los resultados son de mejor calidad que los obtenidos por el heurístico de Lin y Kernighan (alrededor de un 0.5 %) aunque a expensas de emplear tiempos de computación bastante mayores (del orden de algunos minutos).

Conclusiones

- Los Problemas Combinatorios se pueden resolver mediante:
  - **Métodos Exactos**: Proporcionan el óptimo pero suelen ser costosos.
  - **Métodos Heurísticos** : Son rápidos pero no garantizan el óptimo.
- Los **Métodos Constructivos** son heurísticos que proporcionan rápidamente una solución relativamente buena.
- Los **Métodos de Búsqueda Local** son heurísticos iterativos que mejoran una solución dada.
- Ambos métodos pueden **Combinarse** y mejorarse aleatorizando las selecciones.

Al diseñar un Método Heurístico hemos de estudiar

- Calidad: Estudio Teórico (Christofides) o Empírico (TSPLIB)
- Eficiencia: Detalles de implementación, Agilizar los cálculos (Subgrafo candidato)

Referencias

- Jünger, M., Reinelt, G. y Rinaldi, G. (1995), “The Traveling Salesman Problem", En: Ball, M.O., Magnanti, T.L., Monma, C.L. y Nemhauser, G.L. (eds.), *Handbook in Operations Research and Management Science*, Vol. 7, Network Models, pág 225--330. North-Holland, Amsterdam.
- Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G. y Shmoys, D.B. (eds.) (1985), *The Traveling Salesman Problem. A Guided Tour to Combinatorial Optimization*, John Wiley and Sons, Chichester.
- Reinelt, G. (1991), “TSPLIB - A Traveling Salesman Problem Library”, *ORSA Journal on Computing* 3, 376-384.
- Reeves, C.R. (1995), *Modern Heuristic Techniques for Combinatorial Problems*, McGraw-Hill, UK.
- Feo, T. and Resende, M.G.C. (1995), “Greedy Randomized Adaptive Search Procedures”, *Journal of Global Optimization*, 2, 1-27.