Duodécima Semana

(Semana 12, bah)

Continuamos con Heurísticas....

Función Heurística:

❖ El propósito de una función heurística es el de guiar el proceso de búsqueda en la dirección más provechosa, sugiriendo qué camino se debe seguir primero cuando hay más de uno disponible.

El problema del viajante.

Solución Exacta:

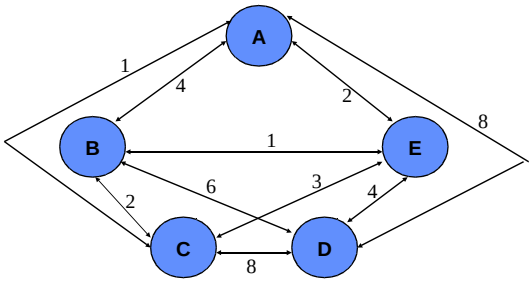
- El problema del viajante de comercio pertenece a la clase de problemas *NP-Hard*. (NP = No-determinístico polinomial).
- Todos los problemas de esta clase son difíciles. Cualquier algoritmo que se conoce para resolver el problema lleva un <u>número exponencial de pasos</u> (Complejidad exponencial = 2^n).
- Hay n! Soluciones para cada viajante (donde n es la cantidad de ciudades)

El problema del viajante.

Solución:

- En 1954 se publicó una solución para 49 ciudades, en 1971 para 64 ciudades, en el 75 para 67, y luego, en los últimos treinta años, hubo saltos cuantitativos más importantes, que permitieron pasar de 318 a 532, luego a 666 y 2392 (en 1987), 7397 (1994), 13.509 (en 1998), y las dos más recientes, 15.112 ciudades en el 2001 y 24.978 en el 2004 cuando se resolvió el problema usando casi 25 mil ciudades o poblaciones en Suecia.

Veamos un ejemplo de un viajante simétrico que parte desde A:



El problema del viajante.

<u>Heurística del vecino más próximo</u>:

❶ Se empieza por un tour parcial trivial que contiene una sola ciudad. (Heurística para elegir el punto de partida).

Mientras queden ciudades sin agregar al tour:

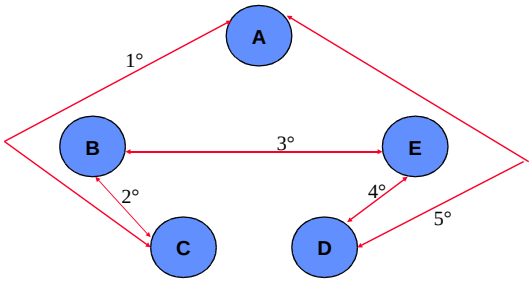
❷ La próxima ciudad elegida es la más cercana a la última del tour siempre que no esté ya incluida en el tour (si hay más de una que sea la más cercana, elegir por orden alfabético).

Fin mientras

❸ Cuando todas las ciudades están en el tour se conecta la última con el origen.

Heurísticas de camino

Heurística del vecino más próximo



CTO = 16

Modelos y Optimización

El problema del viajante.

Heurística del emparchamiento más cercano:

❶ Elegir el eje con menor costo *(si hay más de uno, elegir por orden alfab.).

Mientras queden ciudades sin agregar al tour:

❷ Ir eligiendo de los ejes restantes el que tenga menor costo (*) y nos permita seguir agregando ciudades (el eje se agrega en cualquiera de los extremos).

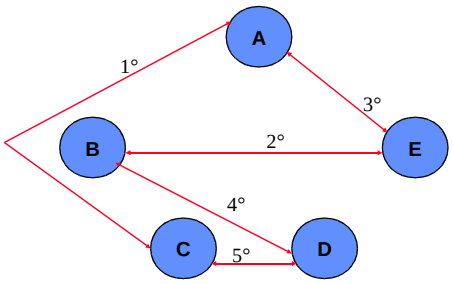
Fin mientras

❸ Se agrega el eje que permita unir las ciudades que hayan quedado desconectadas.

Heurísticas de ejes

Modelos y Optimización

Heurística de emparchamiento más cercano



CTO = 18

Modelos y Optimización

El problema del viajante.

Heurística de Inserción más cercana:

❶ Comenzar con un subtour de una sola ciudad.

Mientras queden ciudades sin agregar al tour:

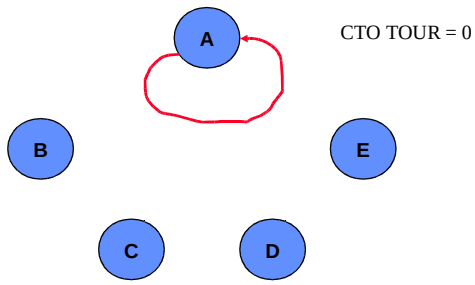
❷ La próxima ciudad a agregar (que aún no haya sido visitada) es la que permita armar un nuevo subtour, con una ciudad más, al menor costo posible. Para cada candidata a agregar se calcula la diferencia de costo entre el tour anterior y el obtenido con el agregado de esta nueva ciudad.

Fin mientras

Heurísticas de tours

Modelos y Optimización

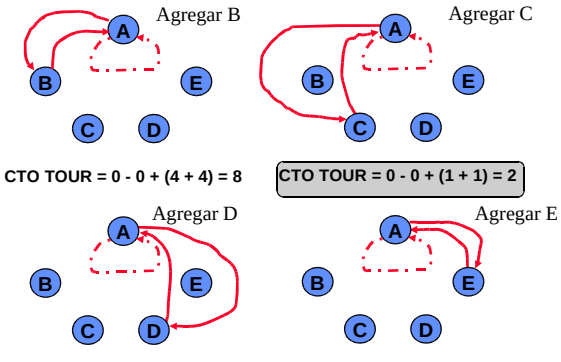
Heurística de inserción más cercana



CTO TOUR = 0

Modelos y Optimización

Heurística de inserción más cercana



CTO TOUR = 0 - 0 + (4 + 4) = 8

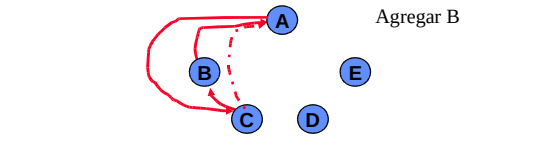
CTO TOUR = 0 - 0 + (1 + 1) = 2

CTO TOUR = 0 - 0 + (8 + 8) = 16

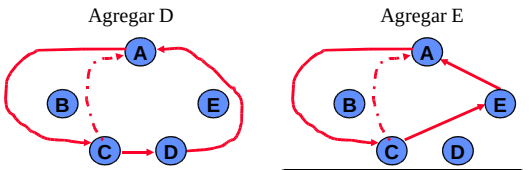
CTO TOUR = 0 - 0 + (2 + 2) = 4

Modelos y Optimización

Heurística de inserción más cercana



CTO TOUR = 2 - 1 + (2 + 4) = 7



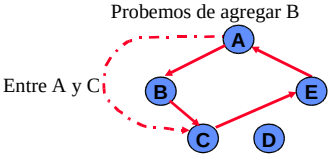
CTO TOUR = 2 - 1 + (8 + 8) = 17

CTO TOUR = 2 - 1 + (3 + 2) = 6

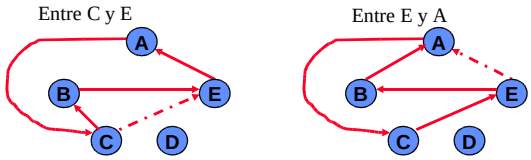
Modelos y Optimización

Heurística de inserción más cercana

Probemos de agregar B



CTO TOUR = 6 - 1 + (4 + 2) = 11



CTO TOUR = 6 - 3 + (1 + 2) = 6

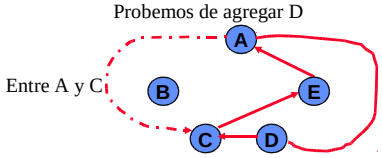
MEJOR OPCION DE B

CTO TOUR = 6 - 2 + (3 + 1) = 8

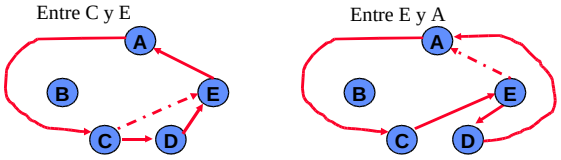
Modelos y Optimización

Heurística de inserción más cercana

Probemos de agregar D



CTO TOUR = 6 - 1 + (8 + 8) = 21



CTO TOUR = 6 - 3 + (8 + 4) = 15

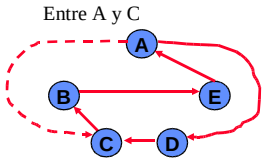
MEJOR DE D, ES PEOR QUE LA MEJOR DE B

CTO TOUR = 6 - 2 + (4 + 8) = 16

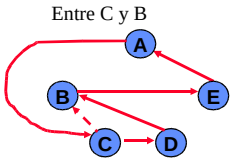
Modelos y Optimización

Heurística de inserción más cercana

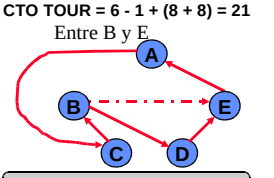
Ahora tenemos que agregar D



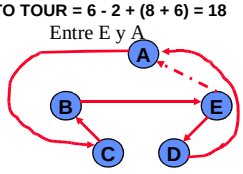
CTO TOUR = 6 - 1 + (8 + 8) = 21



CTO TOUR = 6 - 2 + (8 + 6) = 18



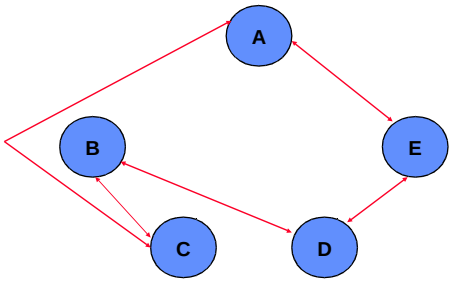
CTO TOUR = 6 - 1 + (6 + 4) = 15



CTO TOUR = 6 - 2 + (4 + 8) = 16

Modelos y Optimización

Heurística de inserción más cercana



Modelos y Optimización

El problema del viajante.

Heurística del árbol generador mínimo (para aplicarla se debe cumplir la desigualdad triangular):

- Representar la posición de las ciudades en un plano x y.
- Armar el árbol de costo mínimo. Cada ciudad se conecta a la mas cercana constituyendo una rama, todas las ramas deben estar conectadas formando el árbol.

Heurísticas de árboles generadores

Modelos y Optimización

El problema del viajante.

Heurística del árbol generador mínimo (cont.) :

- Recorrer el árbol en profundidad (un tour que visita todas las ciudades y puede pasar mas de una vez por cada ciudad.
- Repetir el recorrido salteando las ciudades ya visitadas.

Esta Heurística tiene garantía de calidad = 2

El problema del viajante.

Heurística de Barrido:

Se rota una semirecta alrededor del centro del plano y se ponen las ciudades, en el tour, a medida que la semirecta las toca.

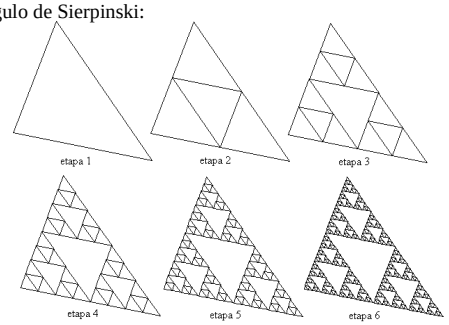
El problema del viajante.

Heurística de Curvas de Sierpinski (L.Platzman y J. Bartholdi´s):

Se genera una curva de Sierpinski (también llamada “triángulo de Sierpinski”) y se arma el tour en el orden en que la curva va tocando las ciudades.

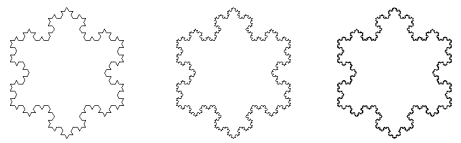
El problema del viajante.

Triángulo de Sierpinski:



El problema del viajante.

Heurística de Curvas de Sierpinski (cont.):



Curva de Koch de orden 3 | Curva de Koch de orden 4 | Curva de Koch de orden 4

Heurísticas de Mejoramiento

K-intercambios:

Dada una solución factible (un tour), tratar de mejorarla cambiando al mismo tiempo k ejes de la solución. (HAY QUE EXPLICITAR COMO SE ELIGEN LOS EJES A CAMBIAR)

Este procedimiento se basa en una propiedad de grafos euclideos: “Si un ciclo Hamiltoniano se cruza a sí mismo, puede ser fácilmente acortado, basta con eliminar las dos aristas que se cruzan y reconectar los dos caminos resultantes mediante aristas que no se corten. El ciclo final es más corto que el inicial.”

Heurísticas de Mejoramiento

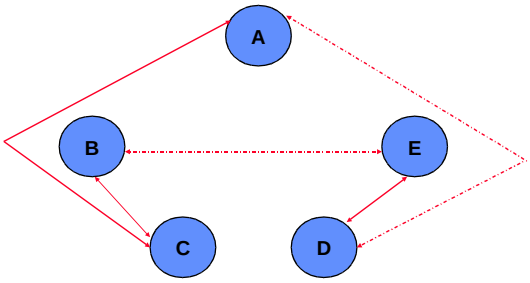
K-intercambios:

Técnicas de optimización local:

① Sea S la solución actual.
② Sea S' la solución obtenida al hacer un k-intercambio.
③ Si S' es mejor que S, definir S = S'.
④ Sino, determinar si existen otros k-intercambios que aún no fueron examinados. Si hay, ir a 1, sino FIN.

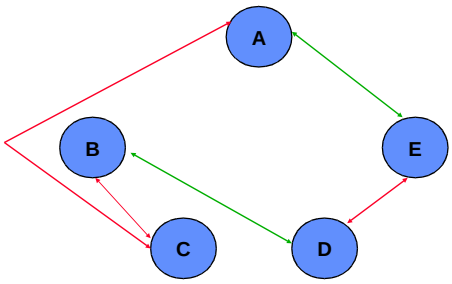
Las más usadas son 2 intercambios (como 2-Opt)

Heurística de mejoramiento de 2-intercambio



Costo total = 16

Heurística de mejoramiento de 2-intercambio



Costo total = 15

Heurísticas de Mejoramiento

Algoritmo de Lin y Kernighan:

Considerar un tour inicial
marca = 1
Mientras marca = 1
  marca = 0
  Etiquetar todos los nodos como no explorados
  Mientras queden nodos sin explorar:
    Seleccionar un nodo i no explorado aún
    Examinar todos los movimientos (2-opt, inserción) que incluyan la arista de i a su sucesor en el tour
    Si alguno de los movimientos reduce la longitud del tour, realizar el mejor de todos y hacer marca = 1
    Sino, etiquetar el nodo i como explorado

Heurísticas de Mejoramiento

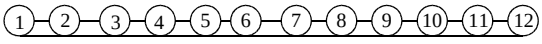
Algoritmo de Lin y Kernighan (cont.):

- Dado el gran número de combinaciones posibles para escoger los movimientos, es evidente que una implementación eficiente del algoritmo tendrá que restringir el conjunto de movimientos a examinar en cada paso.
- El algoritmo de Lin y Kernighan propone un movimiento compuesto, en donde cada una de las partes consta de un movimiento que no mejora necesariamente pero el movimiento compuesto sí es de mejora. De esta forma es como si se realizaran varios movimientos simples consecutivos en donde algunos empeoran y otros mejoran el valor de la solución, pero no se pierde el control sobre el proceso de búsqueda ya que el movimiento completo mejora.

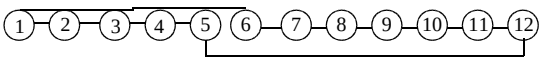
Heurísticas de Mejoramiento

Algoritmo de Lin y Kernighan (cont.):

Consideramos el tour inicial dado por el orden natural de los vértices y lo representamos así:



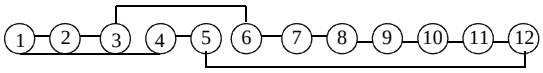
Paso 1: Realiza un *movimiento 2-opt* reemplazando las aristas (12,1) y (5,6) por (12,5) y (1,6).



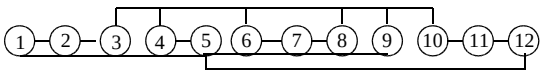
Heurísticas de Mejoramiento

Algoritmo de Lin y Kernighan (cont.):

Paso 2: Realiza un *movimiento 2-opt* reemplazando las aristas (6,1) y (3,4) por (6,3) y (1,4).



Paso 3: Realiza un *movimiento de inserción,* insertando el vértice 9 entre el 1 y el 4.



Modelos y Optimización