

Inteligencia Artificial

Ayudantía C2 - Parte 1

Laura Bermeo

laura.bermeo.12@sansano.usm.cl

Departamento de Informática

Universidad Técnica Federico Santa María

1. Búsqueda Local
2. Técnicas Incompletas
3. Ejercicios

Búsqueda Local

Algoritmos que visitan diferentes regiones del espacio de búsqueda y buscan óptimos locales.

Se busca el mejor óptimo local balanceando dos aspectos:

- **Diversificación** (*explorar*): visitar distintas regiones.
- **Intensificación** (*explotar*): enfocarse en una región y buscar óptimo en este lugar.

Algoritmos Constructivos

A partir de una solución inicialmente vacía, construyen sobre ella asignando valores a cada variable hasta obtener una solución completa.

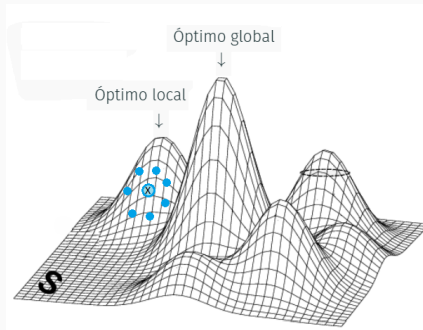
Requieren también la definición de:

- Punto de partida: por dónde se comenzará a construir la solución.
- Función miope: cuál será la siguiente decisión localmente óptima de acuerdo a una función de evaluación.

Búsqueda Local

Algoritmos Reparadores

Comienzan desde una solución ya completa \mathbf{x} y buscan mejorarla a través de **movimientos**.



El conjunto de todas las soluciones \mathbf{x}' posibles resultantes del movimiento aplicado a \mathbf{x} constituyen su **vecindario**.

Definiciones

- Movimiento: Transformación aplicada a una solución, alterando los valores asignados a algunas variables.
 - Bitflip: $0111 \rightarrow 1111$
 - Swap: $0111 \rightarrow 1011$
- Vecindario: Conjunto de soluciones generado a partir del movimiento sobre una solución.
 - $N(x)$ son en algún sentido cercanas a x .
 - Ejemplo con bitflip sobre $x = 1111$:
 $N(x) = [0111, 1011, 1101, 1110]$
- Óptimo local: Una solución $\hat{x} \in S$ es un óptimo local respecto a N si:

$$f(\hat{x}) \geq f(x) \quad \forall x \in N(\hat{x})$$

Definiciones

- Intensificación: Consiste en enfatizar el proceso de búsqueda en las vecindades actuales de la solución candidata, con el objetivo de encontrar el óptimo local de la región.
 - *Explotación* del espacio de búsqueda, obteniendo las mejores soluciones.
- Diversificación: Consiste en estrategias de salto a áreas posiblemente inexploradas del espacio de búsqueda.
 - *Exploración* del espacio, localizando distintas regiones.
- Debe existir un compromiso entre ambos factores.

Técnicas Incompletas

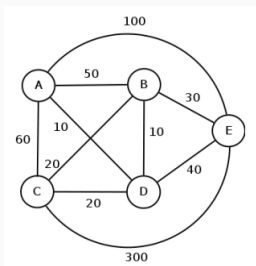
Algoritmo Greedy

Algoritmo constructivo que decide de acuerdo a lo que es localmente óptimo qué agregar a la solución parcial actual.

- Representación
- Función de evaluación
- Función miope
- Punto de partida

Ejemplo Greedy: TSP

- Representación: Arreglo en que la casilla i indica qué ciudad se visitó en esa posición.
- Función de evaluación: Costo asociado al tour
- Función miope: Elegir como siguiente ciudad aquella más cercana que no ha sido visitada
- Punto de partida: Ciudad A



$A - D - B - C - E$ (costo 440)

Hill-Climbing

- A partir de una solución candidata, genera un vecindario y avanza solo si esto significa mejorar la solución actual.
- Dos versiones de HC:
 - Alguna Mejora: Selecciona el primer vecino que mejore la solución actual.
 - Mejor Mejora: Genera el vecindario completo y elige el mejor vecino, siempre y cuando mejore la solución actual.
- Termina su ejecución cuando encuentra la mejor solución candidata de la región.
- Una estrategia para explorar es incluir un número de *restarts*.

Tabu Search

- Algoritmo capaz de escapar de óptimos locales aceptando soluciones de peor calidad.
- A partir de una solución candidata se elige el mejor vecino aunque este empeore la solución actual.
- Se cuenta con una Lista Tabú de tamaño dado para evitar ciclos. Esta puede almacenar por ejemplo soluciones ya visitadas o movimientos aplicados a ciertas variables. Lo que se encuentre en la Lista Tabú no se puede repetir.

Simulated Annealing

- Algoritmo capaz de escapar de óptimos locales aceptando soluciones de peor calidad bajo ciertos criterios.
- La decisión para aceptar soluciones de peor calidad está dado por la Temperatura (T) y la variación de calidad con respecto a la solución anterior (Δ_{eval}).

Simulated Annealing

- SA comienza con una solución inicial. En cada iteración, se aplica un movimiento a la solución actual s , generando una nueva solución s^* .
- Si s^* es de mejor calidad, reemplaza a s y continúa la búsqueda.
- En caso contrario, se calcula $\Delta_{eval} = f(s^*) - f(s)$ (caso minimización) y una probabilidad $P = e^{-\frac{\Delta_{eval}}{T}}$. Si $P > random$, s^* se acepta.
- La temperatura suele ir bajando cada cierto período; algunas versiones del algoritmo permiten recalentamiento.
- Cuando $T \rightarrow 0$, SA tiene un comportamiento parecido a HC Primera Mejora.

Ejercicios

Ejercicio 1

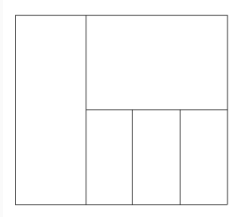
Sea el siguiente modelo:

$$\begin{array}{ll}\max & 5x_1 + 10x_2 + 30x_3 + 25x_4 \\ \text{sujeto a} & x_1 + 3x_2 + 2x_3 + 8x_4 < 10\end{array}$$

- i) Defina representación, movimiento y función de evaluación.
- ii) Resuelva usando Hill Climbing Mejor Mejora. Considere como solución inicial la mochila vacía.

Ejercicio 2

Considere el problema de coloreo de 3 colores para el siguiente mapa:



- i) Defina representación, movimiento y función de evaluación.
- ii) Resuelva usando Tabu Search Primera Mejora. Considere un máximo de 5 iteraciones, tamaño de lista 2 y como solución inicial todas las regiones coloreadas del mismo color.

Ejercicio 3

Sea el siguiente modelo:

$$\begin{array}{ll}\max & 18x_1 + 25x_2 + 11x_3 + 14x_4 \\ \text{sujeto a} & 2x_1 + 2x_2 + x_3 + x_4 \leq 3\end{array}$$

- i) Defina representación, movimiento y función de evaluación.
- ii) Resuelva usando Simulated Annealing. Considere un máximo de 5 iteraciones, temperatura = 10, y como solución inicial la mochila vacía.

Use números aleatorios: 0.15 - 0.83 - 0.33 - 0.41 - 0.09

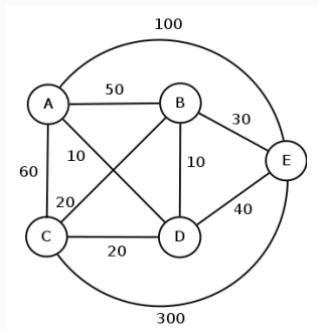
Ejercicio 4

Considere el problema de las n reinas, con $n = 5$.

- i) Defina representación, movimiento y función de evaluación.
- ii) Realice 4 iteraciones de Tabu Search. Considere el tamaño de lista 2.
- iii) ¿Cambia la ejecución al usar un tamaño de lista 5?

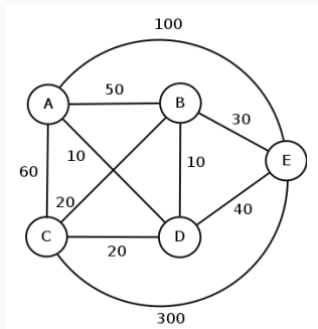
Ejemplo corto 1: Greedy

- Representación: X_i es la i -ésima ciudad visitada, $i = (1..5)$.
- Función miope: Agregar la ciudad más cercana no visitada
- Función de evaluación: Largo del tour
- Punto de partida: A



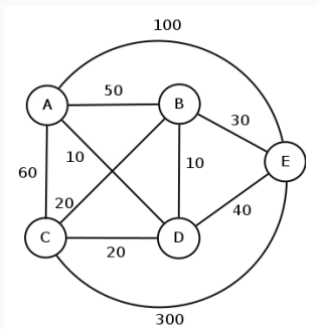
Ejemplo corto 2: Hill Climbing

- Solución inicial aleatoria: A-E-D-B-C
- Costo: 230
- Movimiento: Intercambio entre ciudades contiguas del tour



Ejemplo corto 3: Tabu Search

- Solución inicial aleatoria: A-D-C-E-B
- Costo: 410
- Largo de la lista: 2
- Movimiento: Intercambio entre ciudades contiguas del tour



Ejemplo corto 4: Simulated Annealing

- Solución inicial aleatoria: A-C-D-E-B
- Costo: 200
- $T = 100$
- $Rnd_1 = 0.5$
- Movimiento: Intercambio entre ciudades contiguas del tour

