VC5 – Algoritmos Heurísticos

03MAIR – Algoritmos de optimización

viu Universidad Internacional de Valencia

Preguntas y asuntos pendientes

- Algoritmo por fuerza bruta
- No consultas sobre código python. Si sobre estrategias de diseño de algoritmos
- Preguntas para el trabajo del Seminario(al final)

Agenda

1. Introducción a las metaheurísticas
2. Búsqueda aleatoria
3. Búsqueda basada en trayectorias
4. Métodos basados en trayectorias. Búsqueda Tabú
5. Métodos basados en trayectorias. Recocido Simulado
6. Métodos constructivos. Multiarranque
7. Métodos constructivos. GRASP (greedy randomized adaptative search procedures)
8. Métodos constructivos. Colonia de hormigas
9. Metaheuristicas bioinspiradas.

Introducción a las metaheurísticas

- **Algoritmos aproximados y heurísticos**
  Es posible encontrar soluciones próximas al óptimo en problemas complejos(*) en un tiempo razonable.

- **Destinados a usarlos cuando**:
  - No hay un método exacto o requiere mucho tiempo(o memoria)
  - No es imprescindible la solución optima pero se quiere una buena solución

- ¿Como puedo encontrar soluciones de alta calidad para una función de objetivo dada?

- Terminología
  - Metaheurísticas: Se refiere a técnicas generales (directriz, estrategia,...) (Freed Glober, 1986)
  - Heurístico: Se refiere a un algoritmo concreto.

(*)Problemas no abordables por métodos exactos (búsqueda en arboles, programación dinámica,…) en general debido al tamaño del problema

Introducción a las metaheurísticas

Definición. Metaheurísticas

- Técnicas de algoritmos aproximados, en general iterativos, que exploran el espacio de soluciones con una estrategia “inteligente” o de “sentido común” adaptada al problema.

- **Ventajas**:
  - Técnicas de propósito general
  - En general proporcionan buenos resultados
  - Son “fáciles” de implementar
  - Algunos admiten tratamiento en paralelo

- **Inconvenientes**
  - Son aproximados, no exactos
  - En general no son deterministas(son probabilistas)
  - No existe base teórica(matemática) que los valide.(*)

Introducción a las metaheurísticas

Exactos vs Heurísticos

Exactos

- Se obtiene el óptimo
- Tiempo dependiente de la dimensión
- Problemas “pequeños”

Heurísticos

- Se obtiene soluciones de calidad
- Tiempo “controlable”
- Problemas “grandes”, soluciones “rápidas

Medidas de calidad

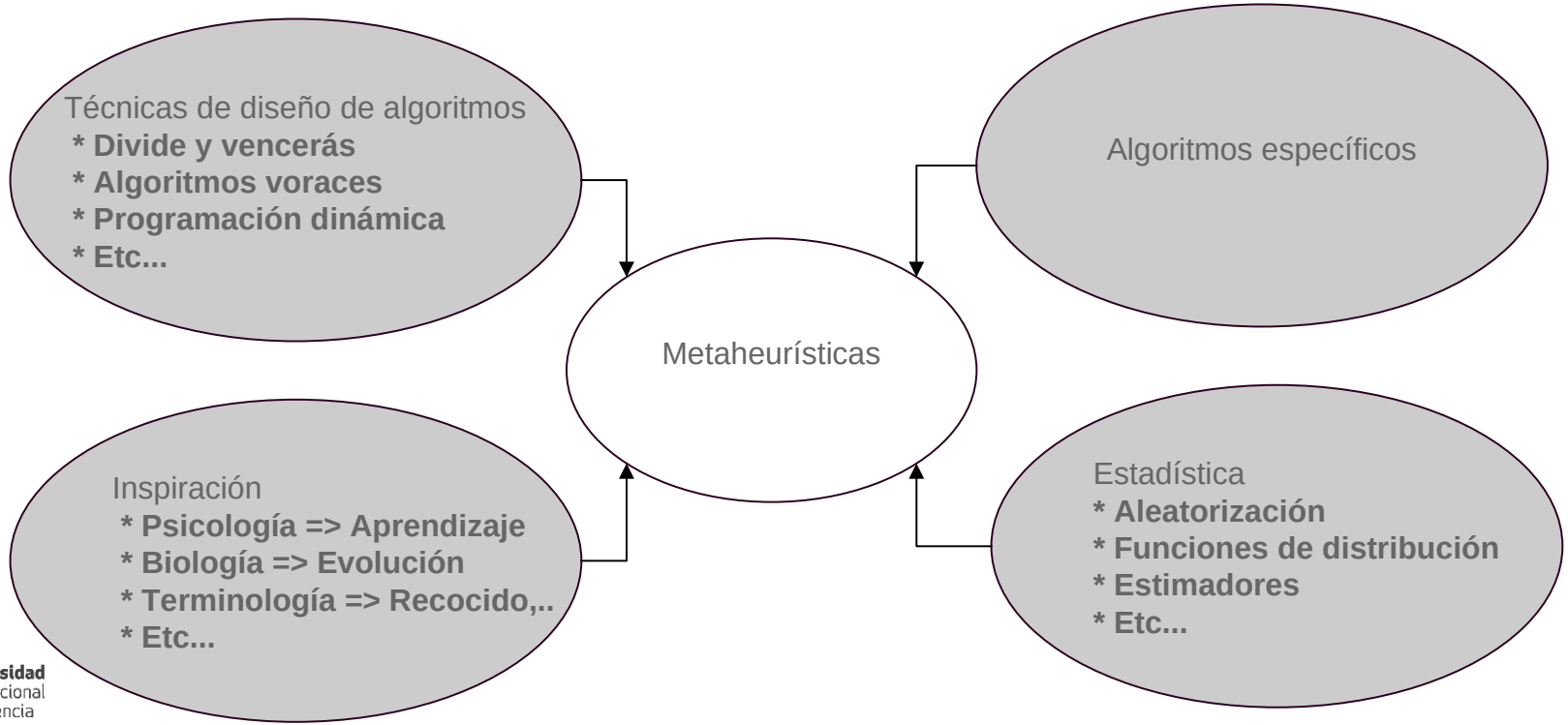
Eficacia: Cuanto “esfuerzo” computacional necesitamos

Bondad: Como de cerca está del óptimo(en promedio)

Robustez: Como de probable es que no obtenga buenas soluciones

Introducción a las metaheurísticas

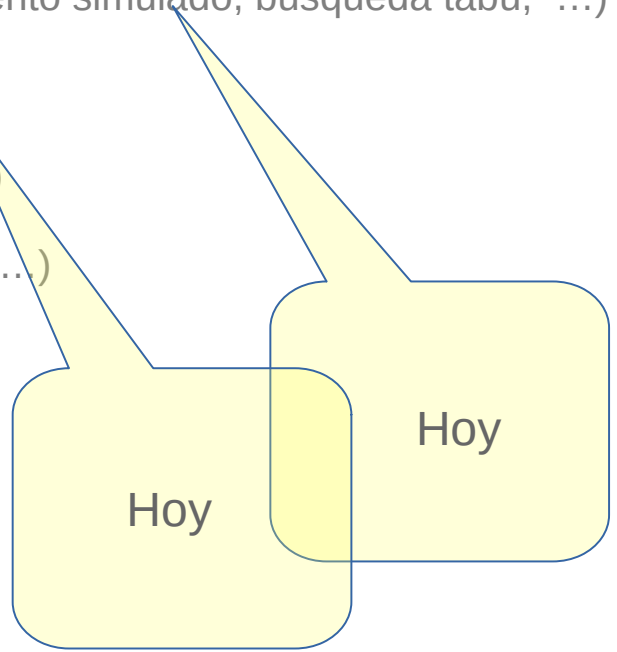
Elementos de las Metaheurísticas



Introducción a las metaheurísticas

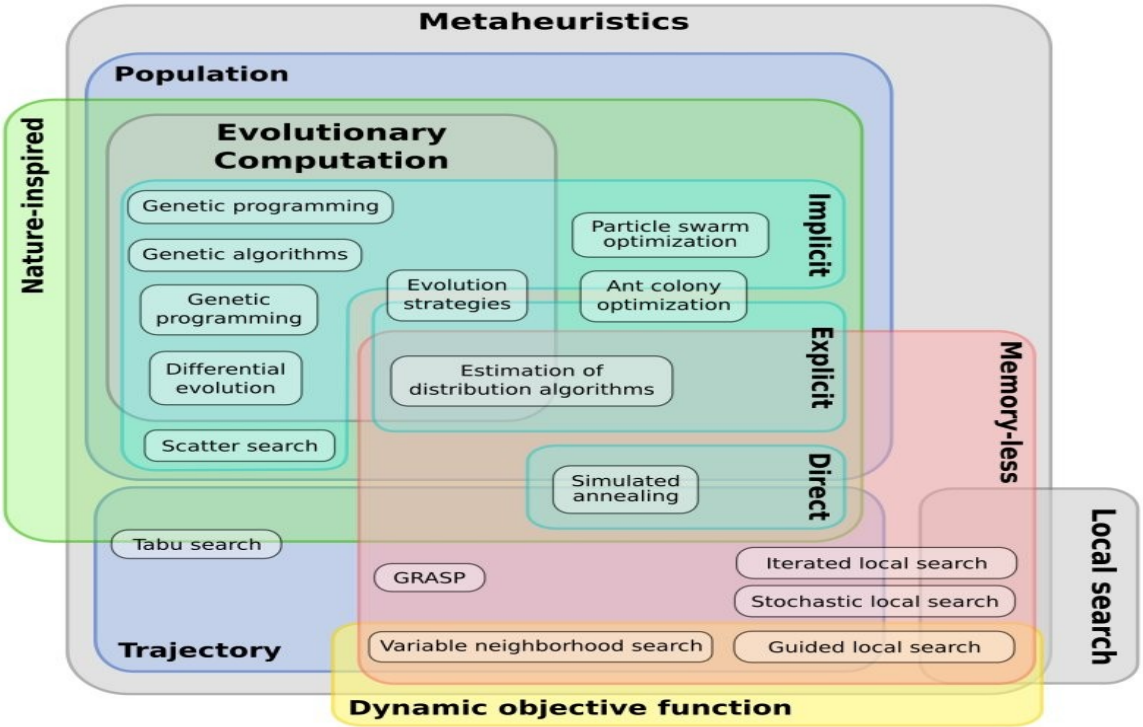
Tipos de Metaheurísticas

- Métodos basados en trayectorias(búsqueda local, enfriamiento simulado, búsqueda tabú, …)
- Métodos constructivos(GRASP, colonia de hormigas, ...)
- Métodos basados en poblaciones(genéticos, evolutivos, …)
- Otros métodos(híbridos, enjambre de partículas, culturales,…)



Introducción a las metaheurísticas

Tipos de Metaheurísticas



Fuente: http://metah.nojhan.net/post/2007/10/12/Classification-of-metaheuristics

Introducción a las metaheurísticas

Características de las Metaheurísticas

- **Intensificación**: Esfuerzo centrado en la búsqueda de la región actual
- **Diversificación**: Esfuerzo centrado en explorar otras regiones

Objetivo: Equilibrio entre **Intensificación** y **Diversificación**

- Identificar rápidamente regiones con soluciones de buena calidad
- No emplear mucho esfuerzo en regiones ya exploradas o poco prometedoras

Metaheurísticas de búsquedas

Búsqueda aleatoria

Definición: Es un proceso por el que se van generando soluciones aleatorias en cada iteración y se devuelve la mejor.

```
Inicio
    GENERA(Solución Inicial)
    Solución Actual ← Solución Inicial;
    Mejor Solución ← Solución Actual;
  Repetir
    GENERA(Solución Actual);
    Si Objetivo(Solución Actual) es mejor que Objetivo(Mejor Solución)
          entonces Mejor Solución ← Solución Actual;
  Hasta (Criterio de parada);
  DEVOLVER (Mejor Solución);
Fin
```

Generación aleatoria

Metaheurísticas de búsquedas

Búsqueda aleatoria

- Parece un método ingenuo pero podemos calcular la probabilidad de obtener la solución óptima en base al n.º de iteraciones.
- Si hay **m soluciones** posibles la probabilidad de encontrar el optimo(si es único) es **1/m**
- Si generamos n soluciones al azar
- Entonces podemos calcular el número de soluciones a generar para tener una probabilidad **p** de que la solución esté en el conjunto generado.

$$n > \frac{\log(1-p)}{\log\left(1-\frac{1}{m}\right)}$$

Metaheurísticas de búsquedas

Búsqueda aleatoria

- Valores del n.º de iteraciones (n) necesarios para asegurar una probabilidad p para diferentes tamaños del espacio de soluciones (m)

p(probabilidad)	m(soluciones)	n(iteraciones)
0,9	1000	2.301
0,8	1000	1.609
0,7	1000	1.203
0,6	1000	916
0,5	1000	693
0,9	2000	4.604
0,8	2000	3.218
0,7	2000	2.407
0,6	2000	1.832
0,5	2000	1.386

p(probabilidad)	m(soluciones)	n(iteraciones)
0,9	1000	2.301
0,8	1000	1.609
0,7	1000	1.203
0,6	1000	916
0,5	1000	693
0,9	2000	4.604
0,8	2000	3.218
0,7	2000	2.407
0,6	2000	1.832
0,5	2000	1.386

$$n > \frac{\log(1-p)}{\log\left(1-\frac{1}{m}\right)}$$

Metaheurísticas de búsquedas

Búsqueda Aleatoria

Ejemplo: TSP – Agente viajero

- **Esquema de representación**: Permutaciones de n ciudades {1,2,….n}
- **Función objetivo**: Distancia

$$Min\ C(S) = \sum_{i=1}^{n-1} (D[S[i], S[i+1]]) + D[S[n], S[1]]$$

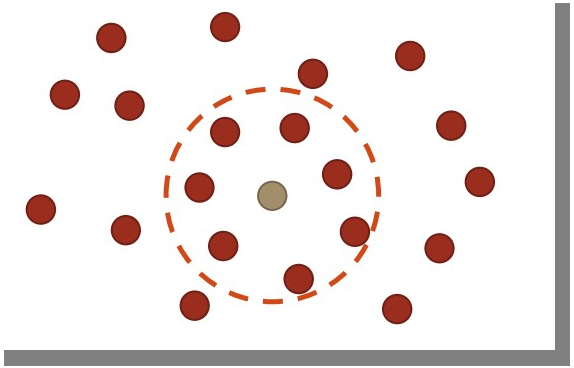
- [3, 4, 5, 1, 6, 7, 8, 2] Distancia: 1832
- [4, 3, 5, 2, 6, 7, 1, 8] Distancia: 1283
- [2, 1, 4, 5, 6, 3, 7, 8] Distancia: 1109
- [4, 5, 6, 3, 7, 1, 8, 2] Distancia: 2001
- ....

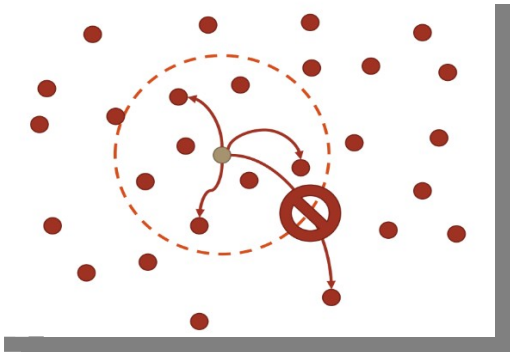
Metaheurísticas de búsquedas basadas en trayectorias

Búsqueda local

Definición: Vecindad

Se refiere a todas las soluciones incluidas en un entorno de la solución de referencia que está delimitado por un operador generador de soluciones(también llamado movimiento).

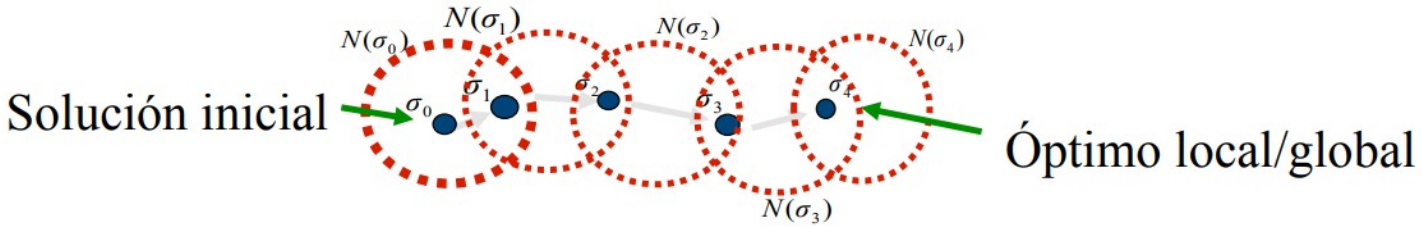




Metaheurísticas de búsquedas basadas en trayectorias

Búsqueda local

Definición: Es un proceso por el que dada una solución inicial, se va seleccionando iterativamente una nueva solución en el entorno local componiendo una trayectoria.(muy orientada a la intensificación y escaso componente diversificador)

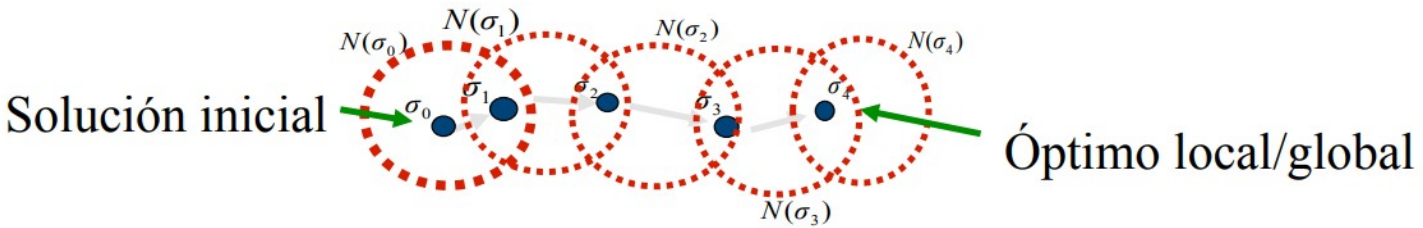


Metaheurísticas de búsquedas basadas en trayectorias

Búsqueda local

Elementos básicos:

- Establecer **solución inicial**
- Establecer una **codificación** para las soluciones
- Establecer un **operador de generación de vecino**(estructura de entorno)
- Establecer un **criterio de parada** para finalizar la iteración



Metaheurísticas de búsquedas basadas en trayectorias

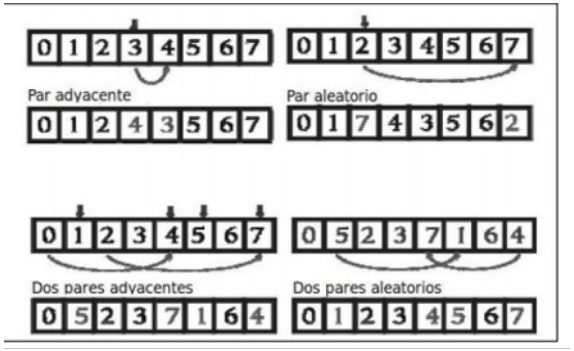
Búsqueda local

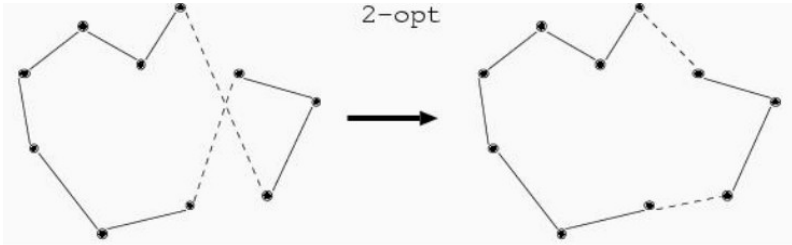
Ejemplo: TSP – Agente viajero

- **Esquema de representación**: Permutaciones de n ciudades {1,2,….n}
- **Función objetivo**: Distancia

$$Min\ C(S) = \sum_{i=1}^{n-1} (D[S[i], S[i+1]]) + D[S[n], S[1]]$$

- **Solución inicial:** Generar una permutación aleatoria
- **Operador generador de soluciones**: Intercambiar valores de 2 en 2





Metaheurísticas de búsquedas basadas en trayectorias

Búsqueda local

Procedimiento:

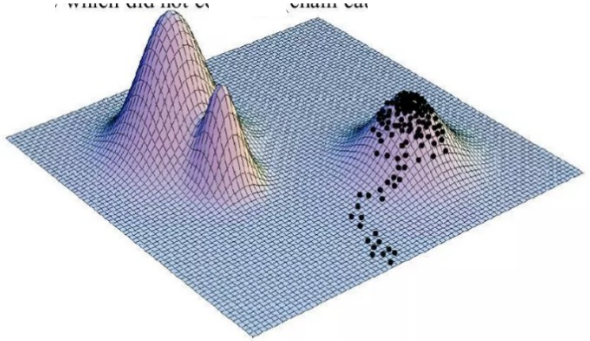
```
Inicio
    GENERA(Solución Inicial)
    Solución Actual ← Solución Inicial;
    Mejor Solución ← Solución Actual;
  Repetir
    Solución Vecina ← GENERA_VECINO(Solución Actual);
    Si Acepta(Solución Vecina)
        entonces Solución Actual ← Solución Vecina;
    Si Objetivo(Solución Actual) es mejor que Objetivo(Mejor Solución)
        entonces Mejor Solución ← Solución Actual;
  Hasta (Criterio de parada);
  DEVOLVER (Mejor Solución);
Fin
```

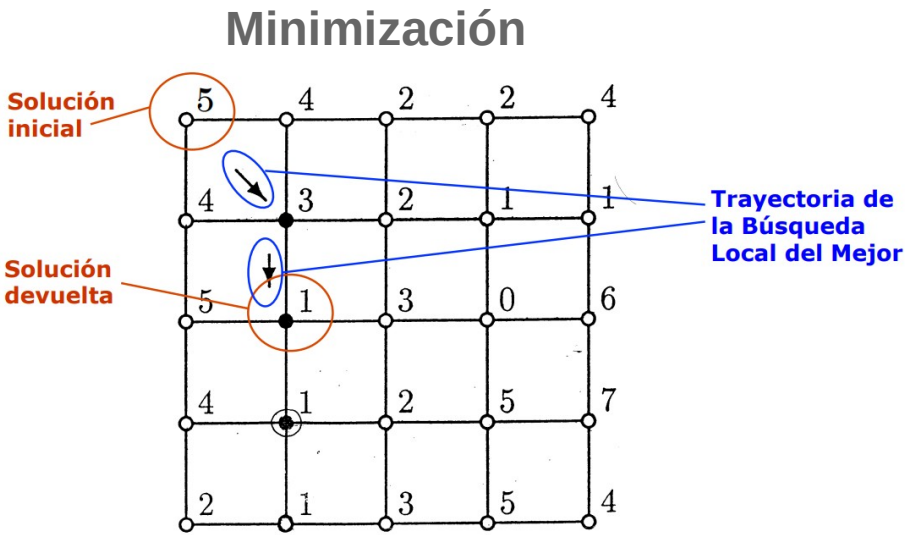
!Ojo! Pueden generarse peores soluciones

Metaheurísticas de búsquedas basadas en trayectorias

Búsqueda local. Desventajas

- Escapar de máximos(o mínimos) locales





Metaheurísticas de búsquedas basadas en trayectorias

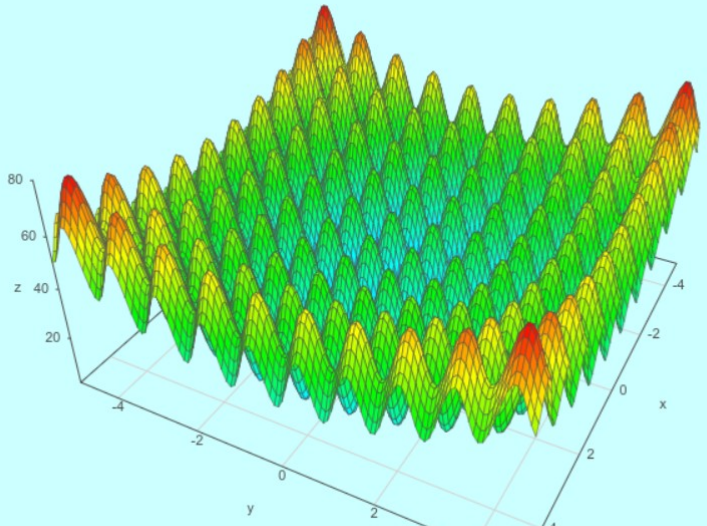
Búsqueda local. ¿Qué es escapar de máximos(mínimos) locales?

Ejemplo: Función Rastrigin

$$f(\mathbf{x}) = 10d + \sum_{i=1}^{d} \left[x_i^2 - 10\cos(2\pi x_i)\right]$$

Mínimo global:

$$f(\mathbf{x}^*) = 0, \text{ at } \mathbf{x}^* = (0, \ldots, 0)$$



Mathematical Expression: 20+x^2-10*cos(2*PI*x) + y^2-10*cos(2*PI*y)
http://al-roomi.org/3DPlot/index.html

Otras Funciones: https://www.sfu.ca/~ssurjano/optimization.html

Metaheurísticas de búsquedas basadas en trayectorias

Búsqueda local. Desventajas

- Escapar de máximos(mínimos) locales. 3 opciones(técnicas):

  - Modificar la estructura de entornos
    búsqueda en entornos variables

  - Permitir movimientos peores respecto a la solución actual
    búsqueda tabú, recocido simulado

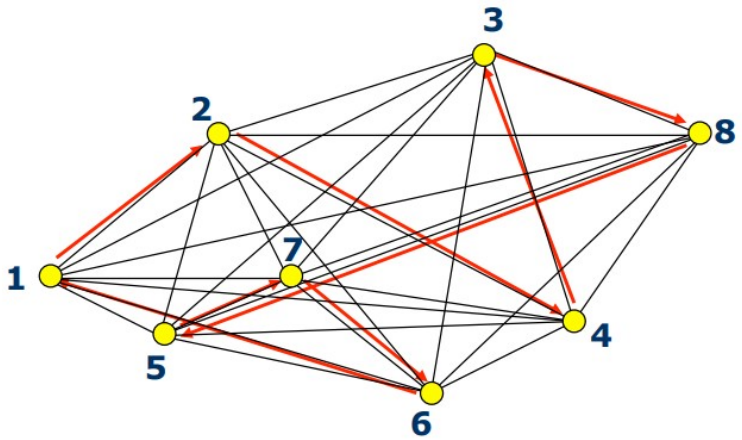
  - Volver a comenzar con otra solución inicial
    búsquedas multi-arranque

Metaheurísticas de búsquedas basadas en trayectorias

Búsqueda local.

Estructura de entornos para el ejemplo del agente viajero

- Solución (aleatoria) inicial: (1 2 4 3 8 5 7 6)



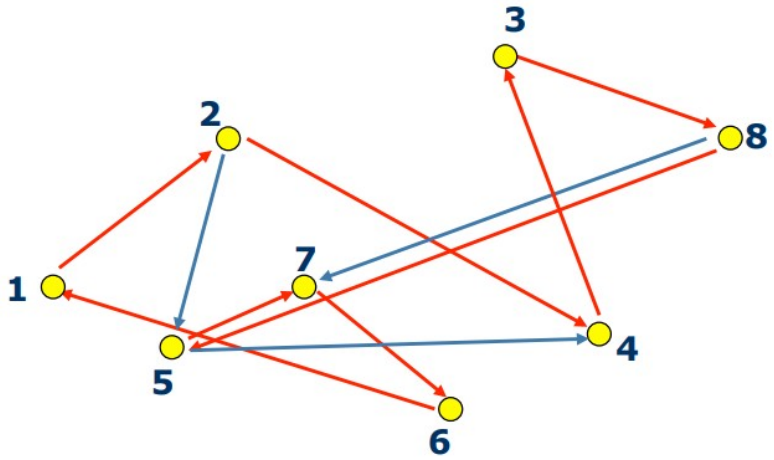
Metaheurísticas de búsquedas basadas en trayectorias

Búsqueda local.

Estructura de entornos para el ejemplo del agente viajero

- Opción 1: Se elige un elemento, se extrae y se inserta en una posición determinada

(1 2 _ 4 3 8 **5** 7 6)
(1 2 **5** 4 3 8 7 6)



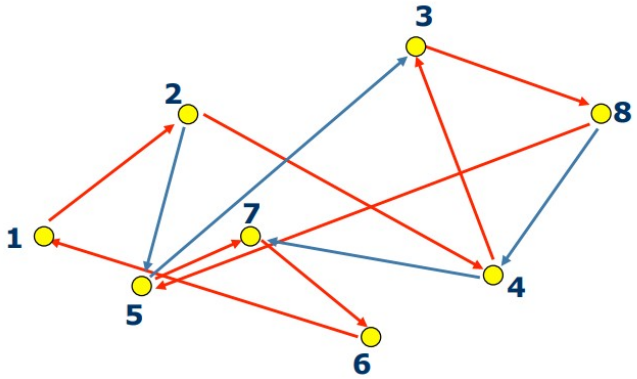
Metaheurísticas de búsquedas basadas en trayectorias

Búsqueda local.

Estructura de entornos para el ejemplo del agente viajero

- Opción 2: Se eligen dos elementos y se intercambian

(1 2 **4** 3 8 **5** 7 6)
(1 2 **5** 3 8 **4** 7 6)



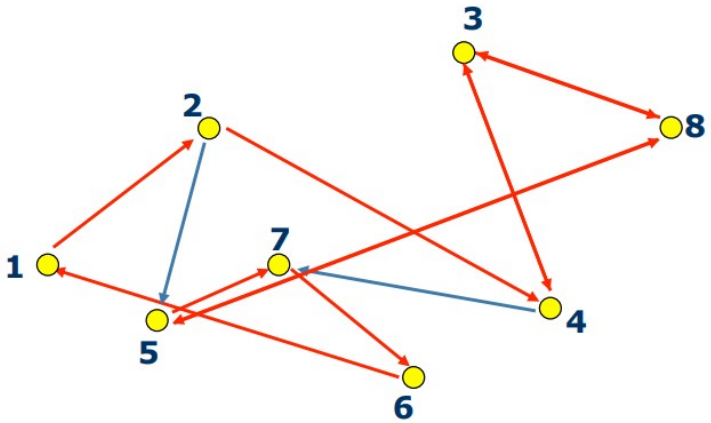
Metaheurísticas de búsquedas basadas en trayectorias

Búsqueda local.

Estructura de entornos para el ejemplo del agente viajero

- Opción 3: se elige una sub-lista y se invierte el orden

(1 2 **4 3 8 5** 7 6)
(1 2 **5 8 3 4** 7 6)



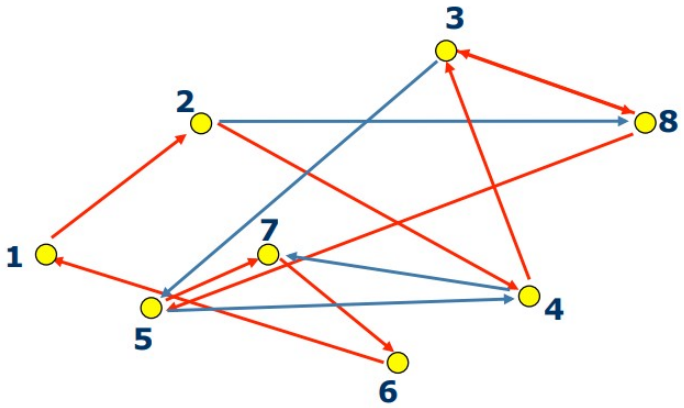
Metaheurísticas de búsquedas basadas en trayectorias

Búsqueda local.

Estructura de entornos para el ejemplo del agente viajero

- Opción 4: se elige una sub-lista y se baraja

(1 2 **4 3 8 5** 7 6)
(1 2 **8 3 5 4** 7 6)



Metaheurísticas de búsquedas basadas en trayectorias

Búsqueda local.

Estructura de entornos para el ejemplo del agente viajero

- ¿como de diferente pueden ser las soluciones vecinas según el operador?
- ¿cual es el tamaño del entorno según que operador?
- ¿es lo mismo un generador para el agente viajero que para otro problema?

Metaheurísticas: Búsqueda Tabú

Definición: Es un proceso de búsqueda por entornos con uso de **memoria adaptativa**.

- El termino proviene de comportamientos socioculturales en los que se establecen prohibiciones que desaparecen con el tiempo.

- La memoria adaptativa consiste en una lista tabú que permite:
  - **restringir** el entorno de búsqueda
  - **intensificar** sobre zonas de espacio de búsqueda ya visitadas
  - **diversificar** sobre zonas del espacio de búsqueda poco o nada visitadas

Metaheurísticas: Búsqueda Tabú

Fundamentos:

- Permitir movimientos a soluciones peores para **escapar de mínimos locales**
- **Evitar recorridos cíclicos** evitando generar (repetidamente) soluciones ya generadas anteriormente (lista tabú)
- Re-inicialización del proceso junto con la lista tabú para conseguir **intensificación(*)** y **diversificación(**)**

Estrategias:

- **Memoria a corto plazo** con el objetivo de evitar el ciclado.
- **Memoria a largo plazo** con el objetivo tanto de intensificar como de diversificar.

(*)intensificación: *regresar a regiones ya exploradas*
(**)diversificación: *visitar nuevas regiones no exploradas (con multiarranque)*

Metaheurísticas: Búsqueda Tabú

Estructura de la lista tabú:

- Lista de **soluciones** tabú: se identifican soluciones ya visitadas y se marcan como tabú para no volver a ellas (no se considera en el vecindario).
- Lista de **movimientos** tabú: se eliminan del entorno todos los vecinos que resultan de aplicar un movimiento tabú
- Lista de **atributos** tabú: se eliminan del entorno todos los vecinos que tienen un determinado atributo tabú

Metaheurísticas: Búsqueda Tabú

Estructura de la lista tabú.

Ejemplo: Lista tabú de **soluciones** para el TSP

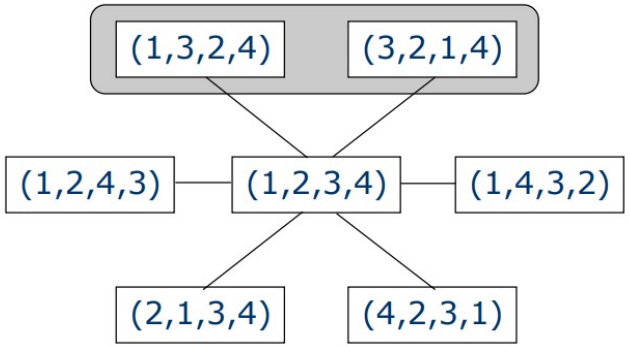
Lista tabú de soluciones:

$$LT = \{ (1,3,2,4), (3,1,2,4), (3,2,1,4) \}$$

Operador de vecino: 2-opt $S_{act} = (1,2,3,4)$

Vecindario reducido de S_{act}:

$E^*(S_{act}) = \{(2,1,3,4), \sout{(3,2,1,4)}, (4,2,3,1), \sout{(1,3,2,4)}, (1,4,3,2), (1,2,4,3) \}$



Metaheurísticas: Búsqueda Tabú

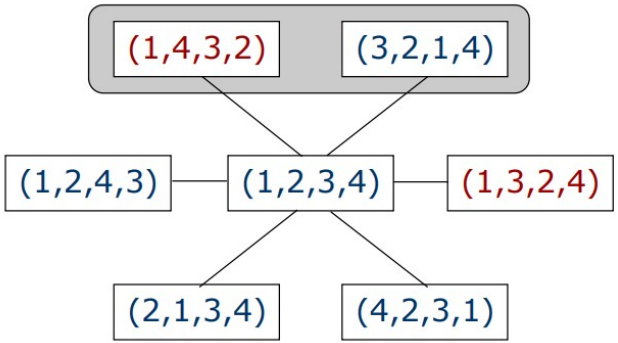
Estructura de la lista tabú.
Ejemplo: Lista tabú de **movimientos** para el TSP

LT = { (1,3), (2,4) }

Operador de vecino: 2-opt S_{act} **= (1,2,3,4)**

Vecindario reducido de S_{act}:

$E^*(S_{act})$ = {(2,1,3,4), ~~(3,2,1,4)~~, (4,2,3,1), (1,3,2,4), ~~(1,4,3,2)~~, (1,2,4,3) }



Metaheurísticas: Recocido simulado - SA(Simulated annealing)(*)

Definición: Es un proceso de búsqueda por entornos basado en la **aceptación probabilista** de soluciones.

- El termino proviene del tratamiento de metales y procesos termodinámicos. Se realiza un enfriamiento controlado con el objetivo de proporcionar determinada estructura al metal (simulated annealing).
- Se aceptan probabilísticamente soluciones peores según la frecuencia de movimientos de escape(aquellos que dan lugar a peores soluciones) por la que se van disminuyendo según una función de probabilidad.
- Propósito: Diversificar al principio e intensificar al final.

(*) Kirkpatrick, S.; Gelatt, C. D.; Vecchi, M. P. (1983). "Optimization by Simulated Annealing"

Metaheurísticas: Recocido simulado - SA

Analogías con el proceso físico

Simulación Termodinámica	Optimización Combinatoria
■ Estados del sistema	■ Soluciones factibles
■ Energía	■ Coste
■ Cambio de estado	■ Solución en el entorno
■ Temperatura	■ Parámetro de control
■ Estado congelado	■ Solución heurística

Metaheurísticas: Recocido simulado - SA

Esquema básico

$T \leftarrow T_0$
Generar una solución inicial x_1 en X;
$F^* \leftarrow F(x_1)$
$x^* \leftarrow x_1$
While la condición de parada no se satisfaga **do**
  Generar aleatoriamente un x en el entorno $V(x_n)$ de x_n
  if $F(x) \leq F(x_n)$, **then** $x_{n+1} \leftarrow x$
  if $F(x) \leq F^*$, **then** $F^* \leftarrow F(x)$ y $x^* \leftarrow x$
  else, generamos un número p aleatorio entre [0,1]
  end if
if $p \leq p(n)$ **then** $x_{n+1} \leftarrow x$
end if
Se desminuye la temperatura según el programa de enfriamiento
end do

Temperatura inicial alta

Criterio de parada:
T=0
ó
n.º de iteraciones

Criterio de aceptación
se solución actual

También se acepta con
probabilidad p(n)

Metaheurísticas: Recocido simulado - SA

Función de probabilidad p(n) para aceptar soluciones peores

- Depende la temperatura(T) y de de la **diferencia de costes de las soluciones**

$$P_{aceptación} = \exp(-\delta/T)$$

- A mayor temperatura => mayor probabilidad de aceptar peores soluciones
- A menor diferencia de costes => mayor probabilidad de aceptar peores soluciones

$\delta = C(s') - C(s)$
s = solución actual
s' = solución vecina

Metaheurísticas: Recocido simulado - SA

Mecanismos de enfriamiento. Descenso de la temperatura

- Descenso constante
- Basado en descensos sucesivos por tramos dependiendo de la iteración
- Descenso exponencial $T_{k+1} = \alpha \cdot T_k$
- Criterio de Boltzmann: $T_k = T_0 / (1 + \log(k))$
- Esquema de Cauchy: $T_k = T_0 / (1 + k)$

Metaheurísticas. Métodos constructivos. Multiarranque

Definición: Es un proceso de búsqueda en el que se **repiten** los dos procesos principales de búsqueda, generación de **solución inicial** y **búsqueda local** hasta satisfacer un **criterio de parada** establecidos.

- **La soluciones iniciales** puede ser puramente aleatoria o “dirigidas” según las características del problema para que sean de buena calidad.
- Parar la búsqueda local en cada repetición se puede usar cualquier método de búsqueda.
- El criterio de parada puede establecerse en fijar el número de iteraciones en cada repetición o basado en la evolución de las soluciones.

Metaheurísticas. Métodos constructivos. GRASP

Definición: Es un proceso **multiarranque** de búsqueda **voraz** , **aleatorio** y **adaptativo** (*greedy randomized adaptative search procedures – GRASP, Feo y Resende, 1995*)

Se trata de un proceso combinado:

- **Voraz** se refiere a la construcción de soluciones iniciales por algún método voraz para que sean de calidad.
- **Aleatorio** se refiere a controlar la “voracidad”. En lugar de comportarse 100% voraz (probabilidad = 1 de elegir el mejor elemento para la solución) se añade una probabilidad sobre un conjunto de elementos candidatos.
- **Adaptativo** se refiere a que los beneficios del nuevo elemento se evalúan para condicionar la elección de los siguientes.

Metaheurísticas. Métodos constructivos. GRASP

Esquema básico

```
begin
    procedure GRASP(maxIt, seed)
        ReadInput();
        for i := 1 to maxIt do
            Solution:=GreedyRandomizeConstruction(seed);
            Solution:=LocalSearchSolution(Solution);
            UpdateSolution(Solution);
        end for;
        return BestSolution;
    end GRASP
```

Genera soluciones GRASP

Criterio de parada por número de iteraciones

Búsqueda local

Metaheurísticas. Métodos constructivos. GRASP

GRASP.
Ejemplo: TSP

- **Voracidad,** criterio al elegir siempre la ciudad más cercana(ya sabemos que esto no da el óptimo pero puede dar mejores soluciones que una elección aleatoria)
- **Aleatoriedad**, no se elige la ciudad más cercana, sino con alguna probabilidad de entre las más cercanas.
- **Adaptativo,** no lo tendremos en cuenta en este caso

Metaheurísticas. Métodos constructivos. GRASP

GRASP.

Ejemplo: TSP – 6 ciudades comenzando por la ciudad 1

LRC – Lista Restringida de Candidatos

Uniforme – Elección aleatoria según distribución Uniforme

	Distancias								
	1	2	3	4	5	6	LRC	Uniforme	Solución
1	-	23	54	17	132	41	{4, 2, 6}	6	(1 6 - - - -)
6	-	48	31	142	39	-	{3, 5, 2}	3	(1 6 3 - - -)
3	-	12	-	45	28	-	{2, 5, 4}	5	(1 6 3 5 - -)
5	-	59	-	22	-	-	{4, 2}	2	(1 6 3 5 2 -)
2	-	-	-	100	-	-	{4}	4	(1 6 3 5 2 4)

Metaheurísticas Bioinspiradas.

- **Colonias de hormigas(ACO):** Marco Dorigo (1992)
- **Optimización del enjambre de partículas(PSO) :** Eberhart, R. y Kennedy, J. (1995).
- **Optimización de la búsqueda de bacterias(BFOA) :** Passino, KM (2002).
- **Cat Swarm Optimization(CAT) :** Chu, SC, Tsai, PW y Pan, JS (2006).
- **Colonia de abejas artificiales(ABC) :** Karaboga, D. y Basturk, B. (junio de 2007).
- **Optimización de ballenas(WOA) :** Mirjalili, S. y Lewis, A. (2016).
- **Optimización de peces vela(Sailfish) (SFO):** Shadravan, S., Naji, HR y Bardsiri, VK (2019).
- **Optimización del Halcón de Harris(HHO) :** Heidari, AA, Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M. y Chen, H. (2019).
- ...

Metaheurísticas. Métodos constructivos. Colonias de hormigas

Definición: Es un proceso **multiagente** de búsqueda en el que cada agente se encarga de **construir probabilísticamente** soluciones aprovechando **información de otros agentes**.
(*Ant Colony Optimization – ACO, Marco Dorigo, 1992*)

Donde:

- **Multiagente** se refiere la utilización de varios agentes que realizan una determinada tarea de búsqueda basada en instrucciones sencillas.

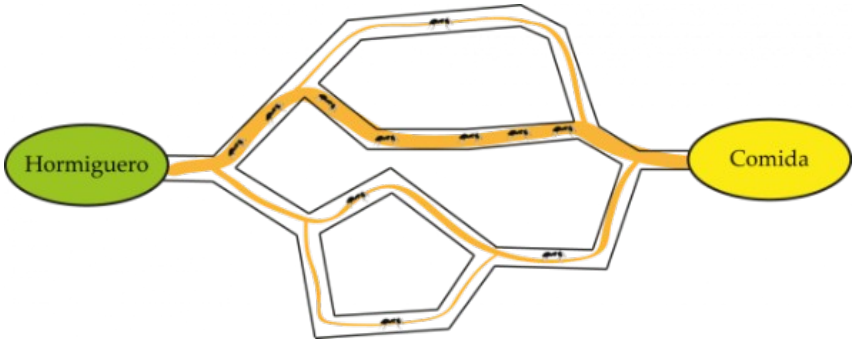
- **Construcción probabilista** se refiere a que cada agente tomará decisiones según las instrucciones con alguna función de probabilidad.

- La información generada por cada agente es compartida con el resto de la comunidad de agentes (feromonas).

Metaheurísticas. Métodos constructivos. Colonias de hormigas

Fundamentos

- Se basa en el comportamiento de comunidades de individuos generan comportamientos aparentemente complejos pero basados en reglas sencillas.
- Un rastro de feromonas indica a otros agentes(hormigas) que deben seguir ese rastros.
- Un componente aleatorio para seguir el rastro puede provocar el descubrimiento de nuevas soluciones mejores que se verán reforzadas por que el rastro permanecerá más tiempo que otros.



Metaheurísticas. Métodos constructivos. Colonias de hormigas

Definición de los rastros de feromonas

- Depositar en cada componente de la solución una cantidad de feromonas proporcional a la calidad de la solución.(En problemas de minimización, inversamente proporcional a valor de la función objetivo)
- Evaporación de feromonas en el tiempo para evitar convergencia a mínimos locales (función exponencial en el tiempo para que sea intensa al principio)
- Incorporación de reglas centralizadas (sin interpretación biológica) que en algunos problemas pueden mejorar el rendimiento (procedimiento demonio). P.Ej. Rastros iniciales de feromonas o rastros extras en algunos nodos o elementos de las soluciones.

Metaheurísticas. Métodos constructivos. Colonias de hormigas

Ventajas sobre otros métodos constructivos

- La componente probabilista permite encontrar gran variedad de soluciones
- Compartir información sirve de guía para encontrar mejores soluciones(aprendizaje reforzado).
- Es posible mejorar la robustez del algoritmo aumentando el número de agentes.
- Permite un buen balance entre intensificación(con el refuerzo de la feromona) y diversificación(con el componente probabilístico y la evaporación)

Metaheurísticas. Métodos constructivos. Colonias de hormigas

Esquema básico

```
Depositar una cantidad de feromona inicial en todas las aristas
Crear m hormigas
Repetir:
    Reiniciar hormigas
    Cada hormiga Construir solución usando feromonas y coste de aristas
    Cada hormiga: Depositar feromona en aristas de la solución
    Evaporar feromona de todas las aristas
Devolver: la mejor solución
```

Metaheurísticas. Métodos constructivos. Colonias de hormigas

Colonia de hormigas - ACO

Ejemplo: TSP (fue el primer problema abordado con esta técnica)

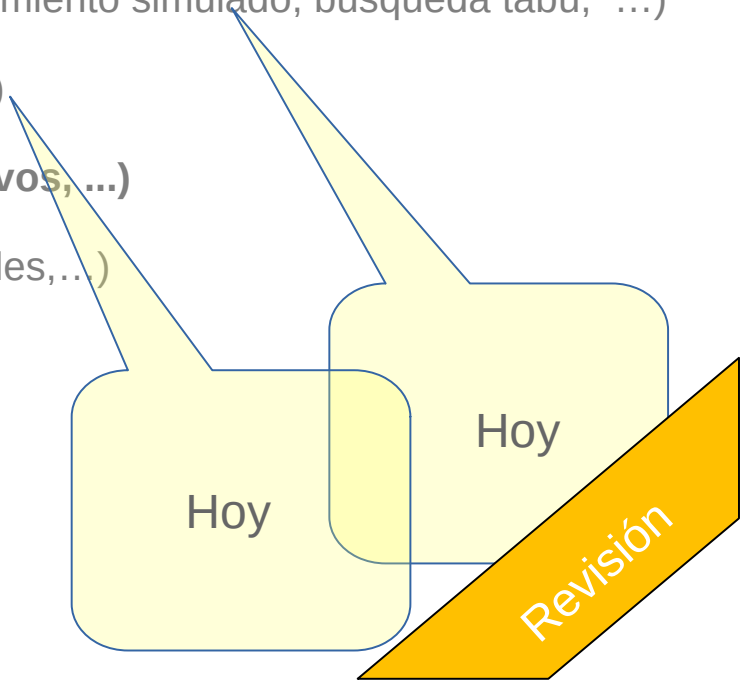
- Cada agente debe guardar el recorrido parcial.
- En cada paso elegir entre las ciudades más cercanas según una probabilidad asociada a la cantidad de feromonas existente(*)
- Depositar(incrementar) una cantidad de feromona sobre la arista utilizada.

(*) Inicializar con alguna cantidad inicial de feromonas o nunca considerar 0 feromonas para no evitar sistemáticamente ciertas rutas.

Introducción a las metaheurísticas

Tipos de Metaheurísticas

- Métodos basados en trayectorias(búsqueda local, enfriamiento simulado, búsqueda tabú, …)
- Métodos constructivos(GRASP, colonia de hormigas, ...)
- **Métodos basados en poblaciones(genéticos, evolutivos, ...)**
- Otros métodos(híbridos, enjambre de partículas, culturales,…)



Metaheurísticas. Otros métodos constructivos

Técnicas híbridas

- Están tomando protagonismo en los últimos años dado los estudios con buenos resultados para algunos problemas(académicos y prácticos).

- Diferentes modos de combinación entre las métodos metaheuristicas.
  - Un método se introduce dentro de otro.
  - Dos o más métodos combinados en serie para mejorar soluciones.
  - Dos o más métodos combinados en paralelo para mejorar la diversidad.
  - Dividiendo el espacio de soluciones en subproblemas.

Otras Metaheuristicas

https://en.wikipedia.org/wiki/List_of_metaphor-based_metaheuristics

Próximo día, prácticas sobre TSP:

Resolución por búsqueda aleatoria

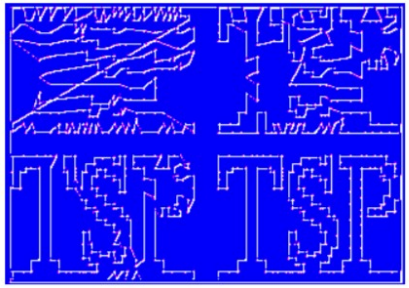
Resolución por Búsqueda local

Resolución por recocido simulado

Planteamiento por Colonia de Hormigas - ACO

El problema del agente viajero – TSP. TSPLIB

Juegos de datos para poner a prueba nuestros diseños al resolver el problema del TSP



TSPLIB http://elib.zib.de/pub/mp-testdata/tsp/tsplib/

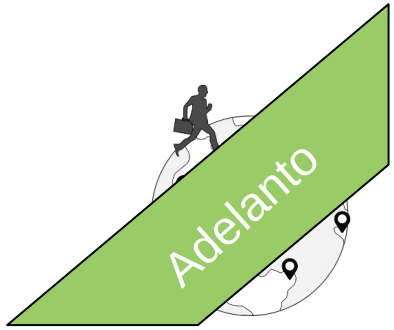
TSPLIB is a library of sample instances for the TSP (and related problems) from various sources and of various types. Instances of the following problem classes are available.

Symmetric traveling salesman problem (TSP)

Given a set of n nodes and distances for each pair of nodes, find a roundtrip of minimal total length visiting each node exactly once. The distance from node i to node j is the same as from node j to node i.

TSP data

Best known solutions for symmetric TSPs

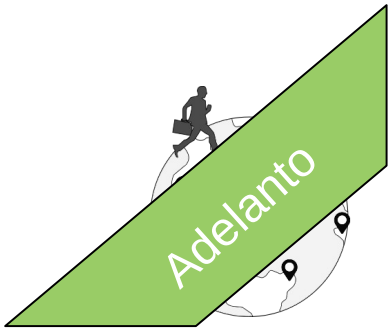


Preparación de los datos

Descargar el juego de datos

```
import urllib.request

file = "swiss42.tsp"

urllib.request.urlretrieve("http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/swiss42.tsp", file)

#42 cities Switzerland (Fricker) en formato matriz
#http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/swiss42.tsp

#48 capitals of the US (Padberg/Rinaldi) en formato coordenadas
#http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/att48.tsp
```

http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/index.html

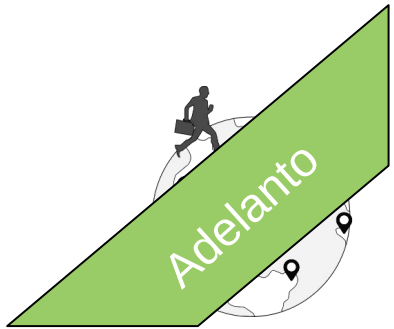


Preparación de los datos

Instalar y cargar módulos

```
#https://tsplib95.readthedocs.io/installation.html
!pip install tsplib95
```

```
import tsplib95
import random
from math import e
```

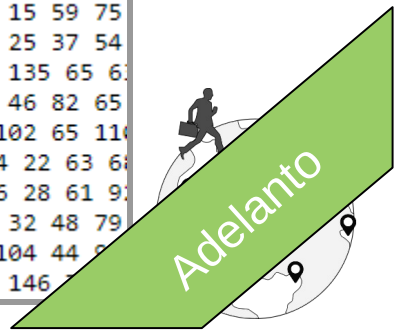


Preparación de los datos

Cargar datos del problema

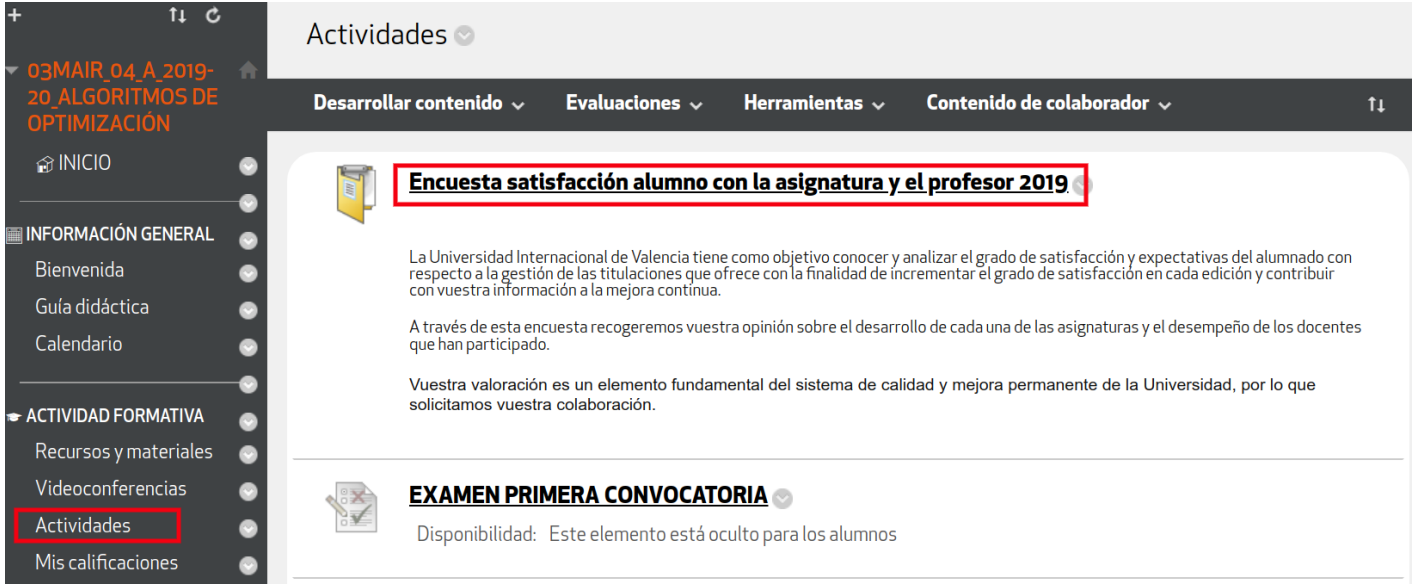
```
problem = tsplib95.load_problem(file)


#Nodos
Nodos = list(problem.get_nodes())

#Aristas
Aristas = list(problem.get_edges())
```

```
NOMBRE: swiss42
TIPO: TSP
COMENTARIO: 42 Staedte Schweiz (Fricker)
DIMENSION: 42
EDGE_WEIGHT_TYPE: EXPLICIT
EDGE_WEIGHT_FORMAT: FULL_MATRIX
EDGE_WEIGHT_SECTION
0 15 30 23 32 55 33 37 92 114 92 110 96 90 74 76 82 72 78 82 159 122 131 206 112 57 28 43 70
  15 0 34 23 27 40 19 32 93 117 88 100 87 75 63 67 71 69 62 63 96 164 132 131 212 106 44 33 5
  30 34 0 11 18 57 36 65 62 84 64 89 76 93 95 100 104 98 57 88 99 130 100 101 179 86 51 4 18
  23 23 11 0 11 48 26 54 70 94 69 75 75 84 84 89 92 89 54 78 99 141 111 109 89 89 11 11 11 54
  32 27 18 11 0 40 20 58 67 92 61 78 65 76 83 89 91 95 43 72 110 141 116 105 190 81 34 19 35
  55 40 57 48 40 0 23 55 96 123 78 75 36 36 66 66 63 95 34 34 137 174 156 129 224 90 15 59 75
  33 19 36 26 20 23 0 45 85 111 75 82 69 60 63 70 71 85 44 52 115 161 136 122 210 91 25 37 54
  37 32 65 54 58 55 45 0 124 149 118 126 113 80 42 42 40 40 87 87 94 158 158 163 242 135 65 6
  92 93 62 70 67 96 85 124 0 28 29 68 63 122 148 155 156 159 67 129 148 78 80 39 129 46 82 65
 114 117 84 94 92 123 111 149 28 0 54 91 88 150 174 181 182 181 95 157 159 50 65 27 102 65 11
  92 88 64 69 61 78 75 118 29 54 0 39 34 99 134 142 141 157 44 110 161 103 109 52 154 22 63 6
 110 100 89 89 78 75 82 126 68 91 39 0 14 80 129 139 135 167 39 98 187 136 148 81 186 28 61 9
  96 87 76 75 65 62 69 113 63 88 34 14 0 72 117 128 124 153 26 88 174 136 142 82 187 32 48 79
  90 75 93 84 76 36 60 80 122 150 99 80 72 0 59 71 63 116 56 25 170 201 189 151 252 104 44
  74 63 95 84 83 56 63 42 148 174 134 129 117 59 0 11 8 63 93 35 135 223 195 184 273 146
```



Actividad. Encuesta de satisfacción

- Disponible hasta 6/7/2020



Ampliación de conocimientos y habilidades

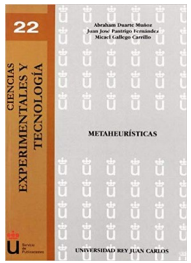
- Bibliografía

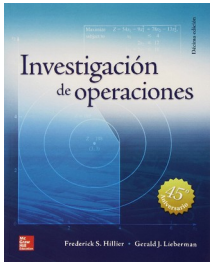
- Duarte, A. (2008). Metaheurísticas. Madrid: Dykinson.

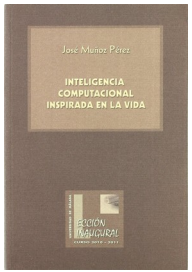
- Hillier, F. S., y Lieberman, G. J. (2015). Investigación de Operaciones.(Capítulo 14. Metaheurísticas)

- Inteligencia Artificial inspirada en la Vida – José Muñoz Pérez
http://libros.metabiblioteca.org:8080/bitstream/001/257/8/978-84-9747-330-9.pdf

- Practicar







SE – Problemas del seminario

03MAIR – Algoritmos de optimización

viu Universidad Internacional de Valencia

Problemas

1. Organizar sesiones de doblaje

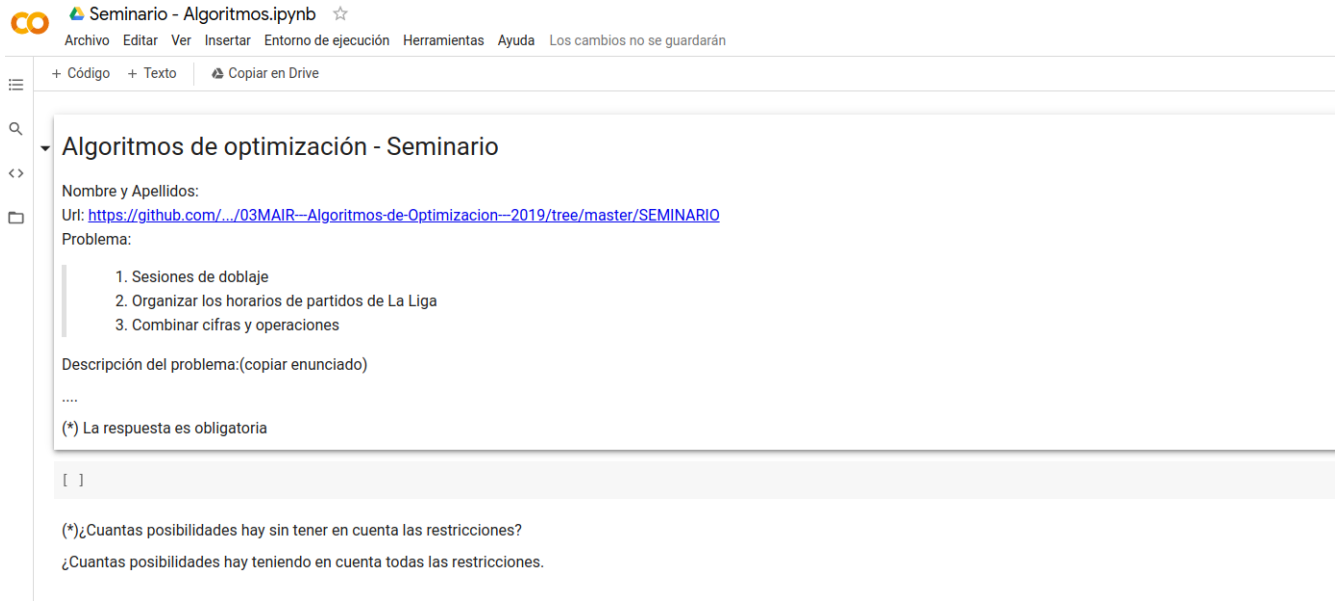
2. Planificar los horarios de una jornada de la Liga

3. Cifras y operaciones

Entregable

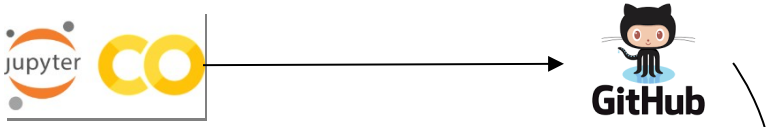
- **Plantilla para el documento**

  https://colab.research.google.com/drive/1NVFHsnmrE-wFLX8y1SC3tKlh2et5FOz8



Entregable

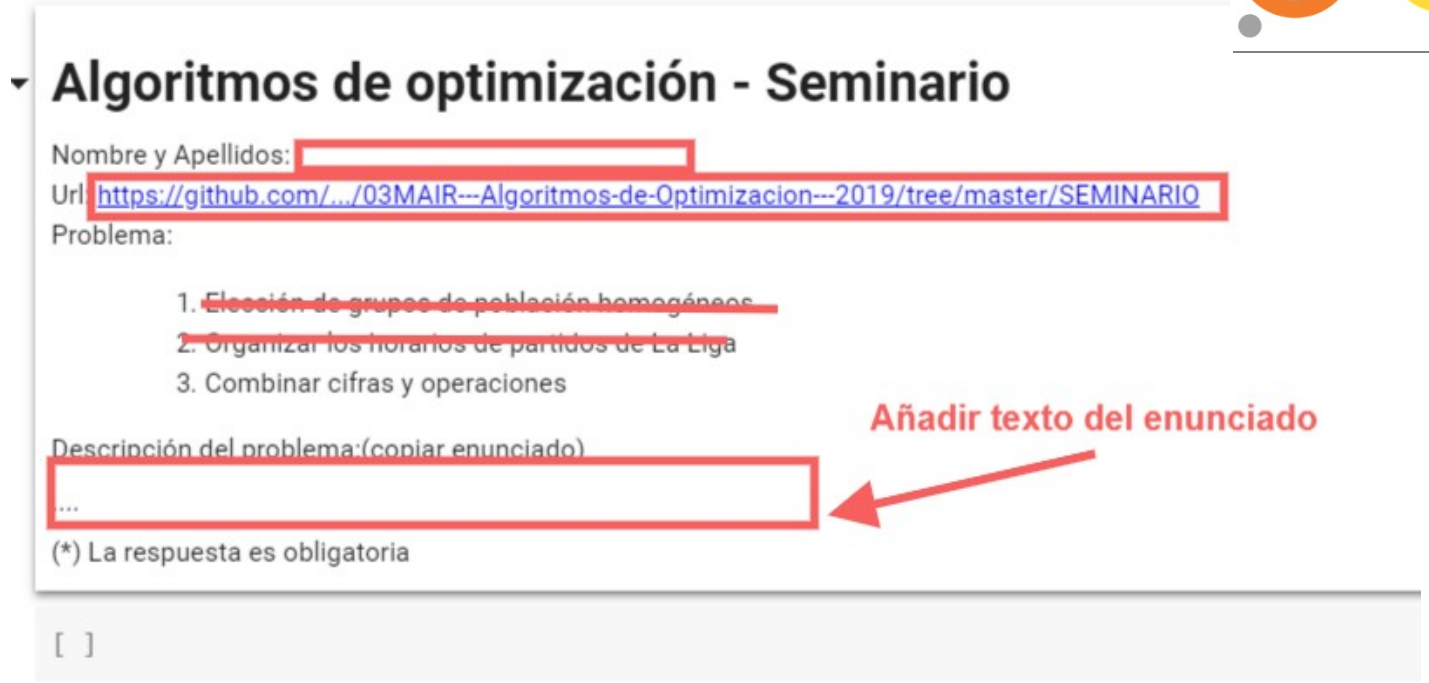
- Generar un Notebook en GitHub (carpeta SEMINARIO)



- Entrega de documento .pdf con en Notebook (como las A. Guiadas)

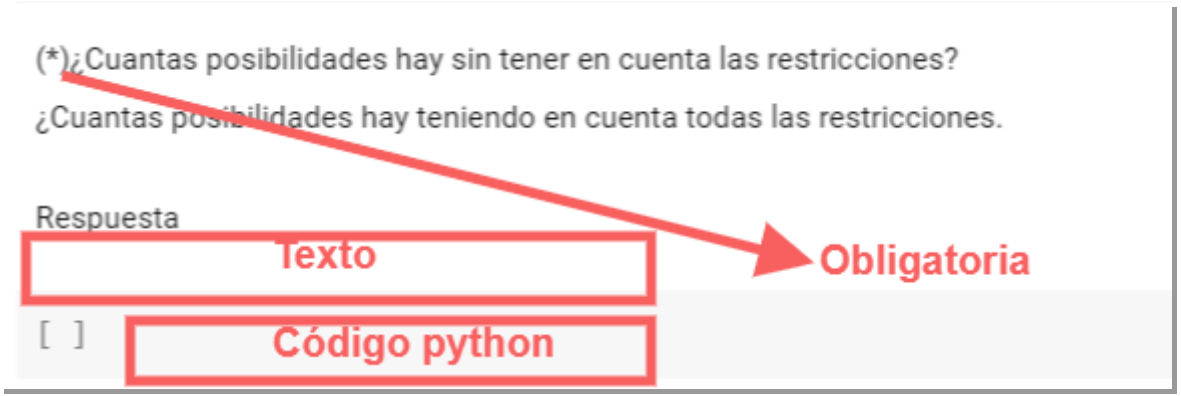
Entregable

- Cabecera



Entregable

- Pregunta – Respuesta (texto + Python)



Entregable

Pregunta – Respuesta (texto + Python)

- **(*)¿Cuantas posibilidades hay sin tener en cuenta las restricciones?**
- ¿Cuantas posibilidades hay teniendo en cuenta todas las restricciones.
- **(*) ¿Cual es la estructura de datos que mejor se adapta al problema? Argumenta la respuesta**
  (Es posible que hayas elegido una al principio y veas la necesidad de cambiar, argumenta)
- **(*)¿Cual es la función objetivo?**
- **(*)¿Es un problema de maximización o minimización?**

(*) Obligatorio responder

Entregable

Pregunta – Respuesta (texto + Python)

- Diseña un algoritmo para resolver el problema por fuerza bruta
- Calcula la complejidad del algoritmo por fuerza bruta
- **(*)Diseña un algoritmo que mejore la complejidad del algoritmo por fuerza bruta. Argumenta porque crees que mejora el algoritmo por fuerza bruta**
- **(*)Calcula la complejidad del algoritmo**
- Según el problema (y tenga sentido), diseña un juego de datos de entrada aleatorio.

(*) Obligatorio responder

Entregable

Pregunta – Respuesta (texto + Python)

- Aplica el algoritmo al juego de datos aleatorio generado.
- Enumera las referencias que has utilizado(si ha sido necesario) para llevar a cabo el trabajo
- Describe brevemente en unas líneas como crees que es posible avanzar en el estudio del problema. Ten en cuenta incluso posibles variaciones del problema y/o variaciones al alza del tamaño.

Evaluación.

Total 13 cuestiones:

6 obligatorias(*) , aseguran 7/10

- **7 opcionales , añaden 2 puntos más: 9/10**
- **1 punto por presentación, descripción**

  - lenguaje claro
  - código comentado
  - acompaña ilustraciones si es necesario(imágenes)

  ...

Fecha limite de entrega 1ª convocatoria: 25/01/2021

Fecha limite de entrega 2ª convocatoria: 11/02/2021

Gracias

raul.reyero@campusviu.es

viu Universidad Internacional de Valencia