Cómputo Evolutivo

Stalin Muñoz Gutiérrez

Centro de Ciencias de la Complejidad
Universidad Nacional Autónoma de Mexico (UNAM)

Resolver problemas por evolución

El cómputo evolutivo es una manera de resolver problemas enfocándose en las preguntas:

- ¿qué resolver?
- ¿qué es una buena solución al problema?

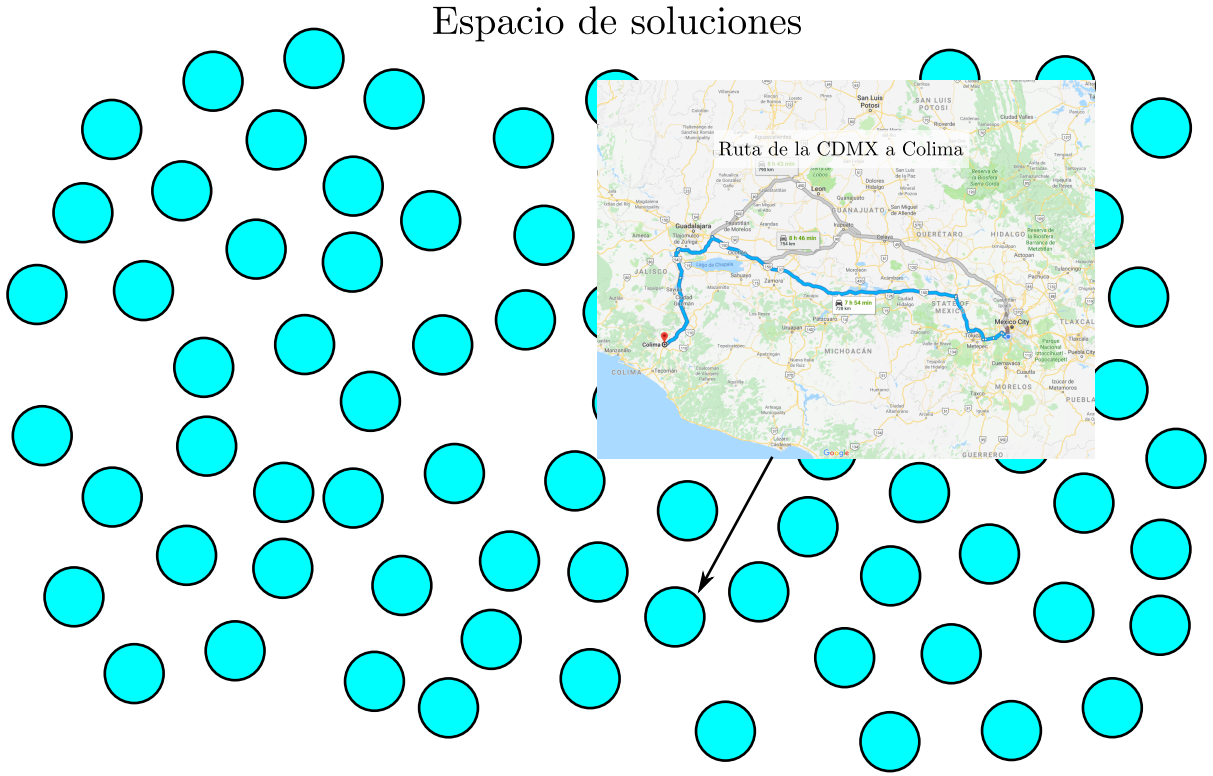
Los algoritmos de cómputo evolutivo son parte de una categoría de algoritmos denominados *algoritmos de búsqueda metaheurísticos*.

Espacio de soluciones

Espacio de soluciones

Espacio de soluciones a un problema

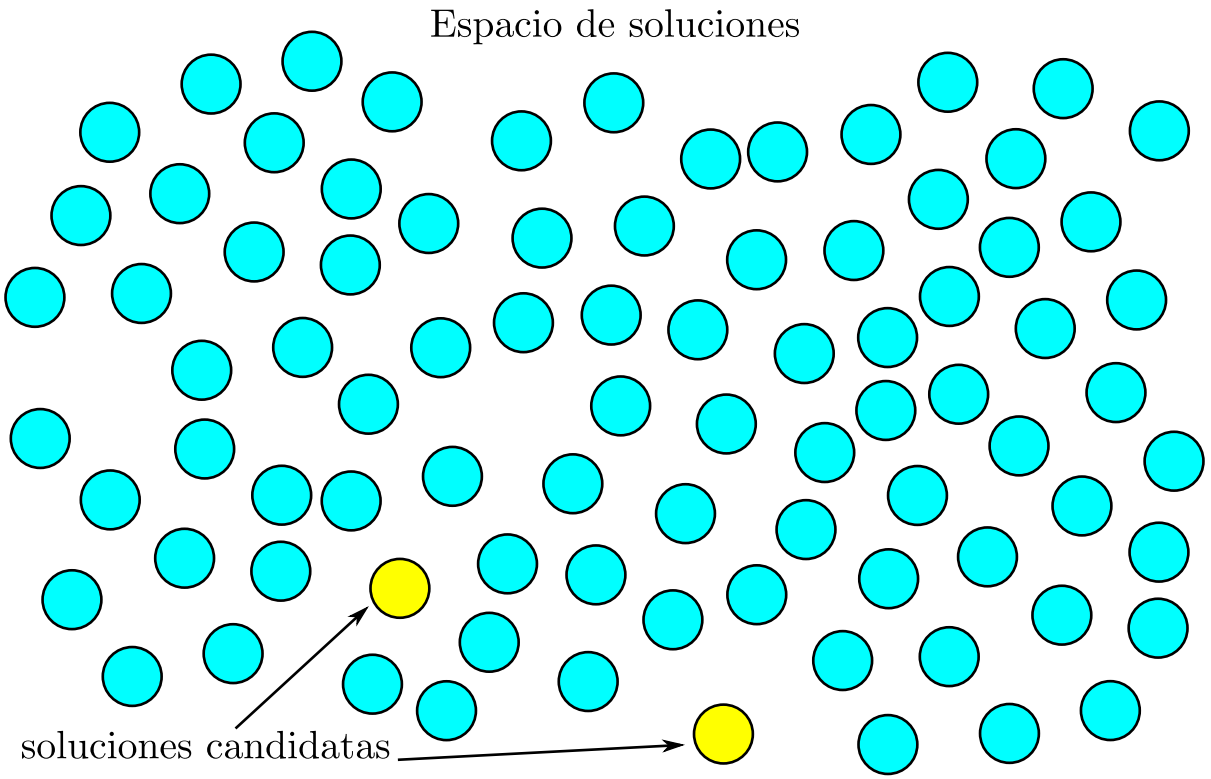
El conjunto donde existen todas las posibles soluciones a un problema.



Espacio de soluciones

Soluciones candidatas

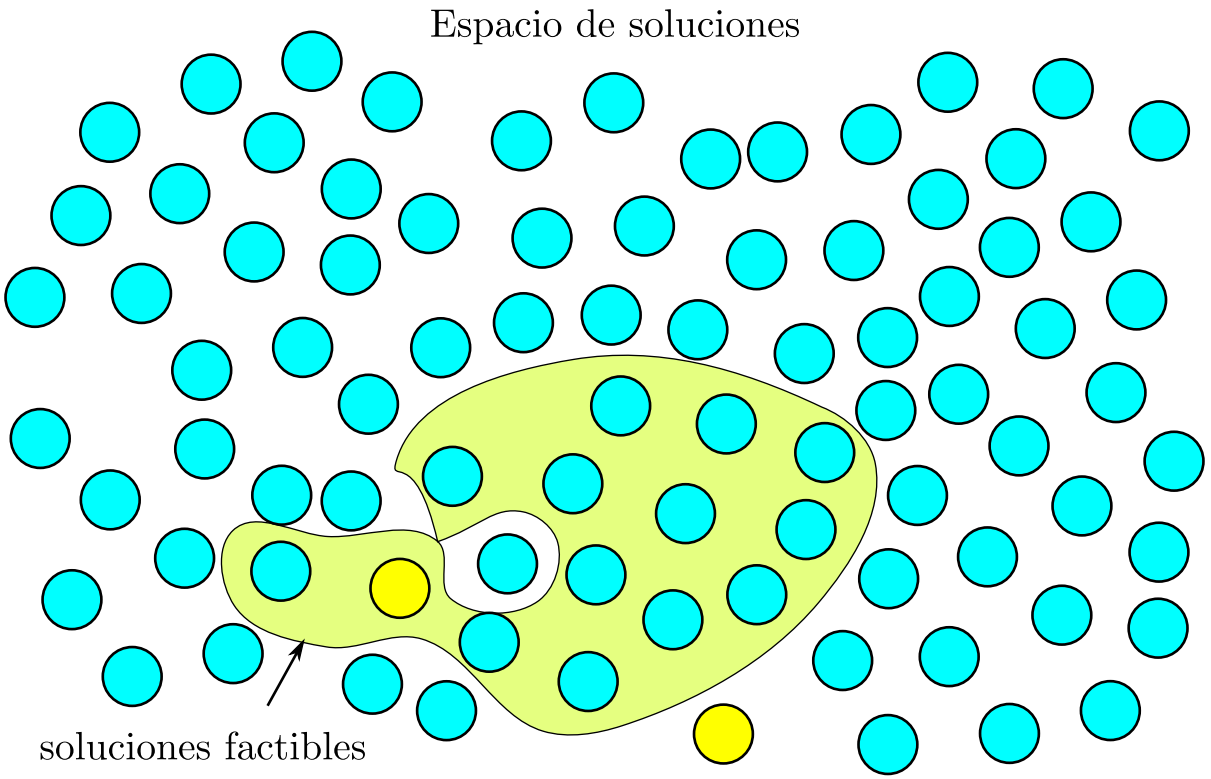
Cualquier elemento del espacio de soluciones.



Espacio de soluciones

Soluciones factibles

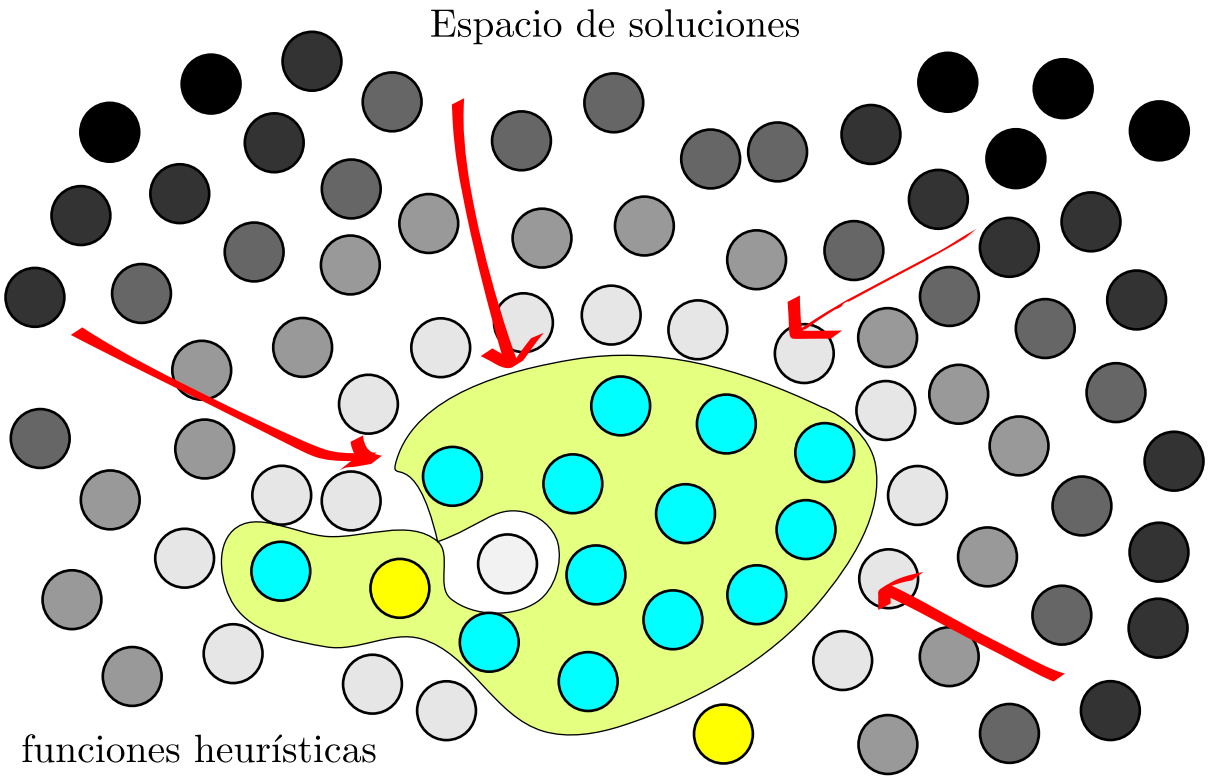
Satisfacen un conjunto de restricciones.



Espacio de soluciones

Funciones heurísticas

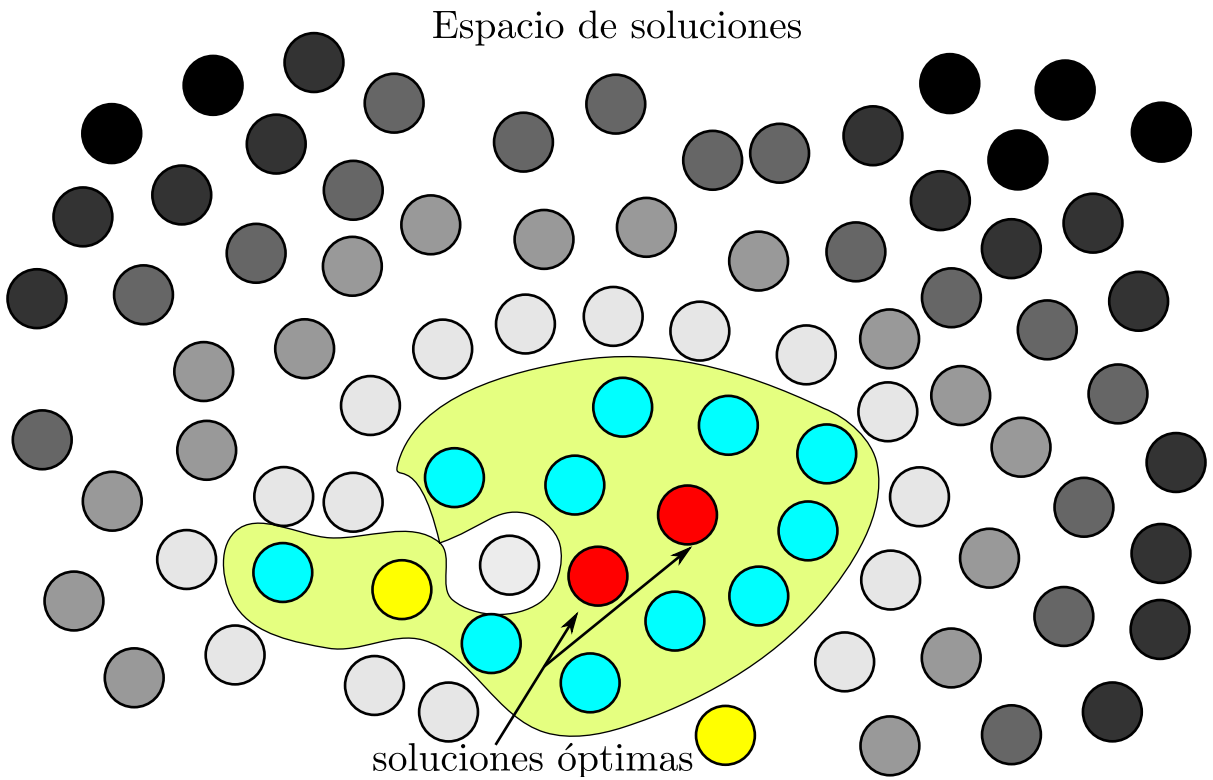
Guían al algoritmo hacia las regiones factibles.



Espacio de soluciones

Soluciones óptimas

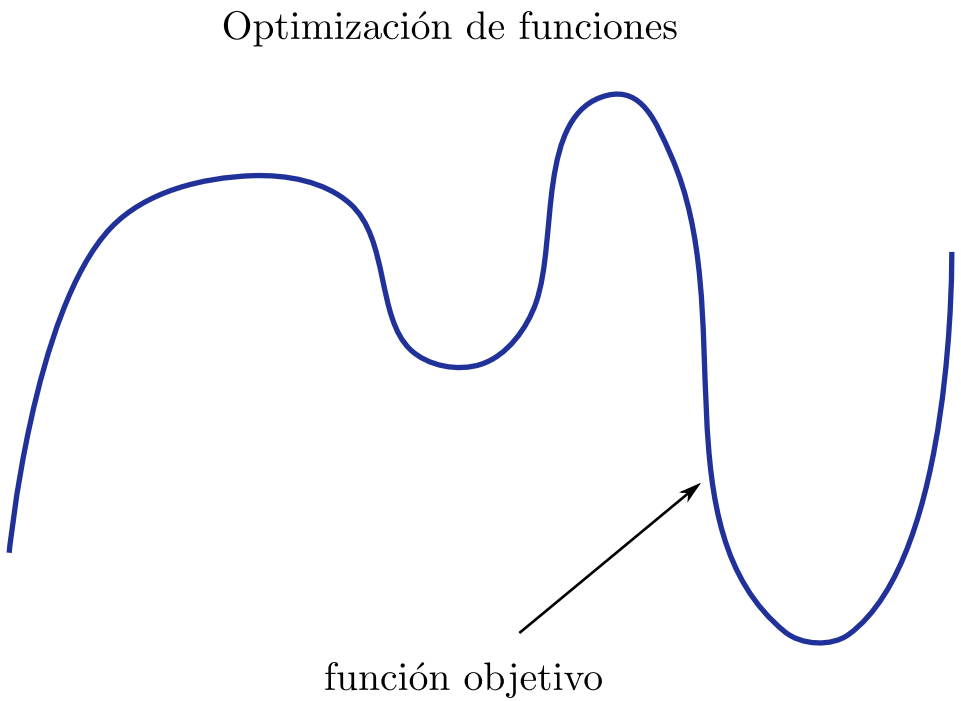
Las mejores soluciones factibles posibles.



Optimización de funciones

Problema de optimización

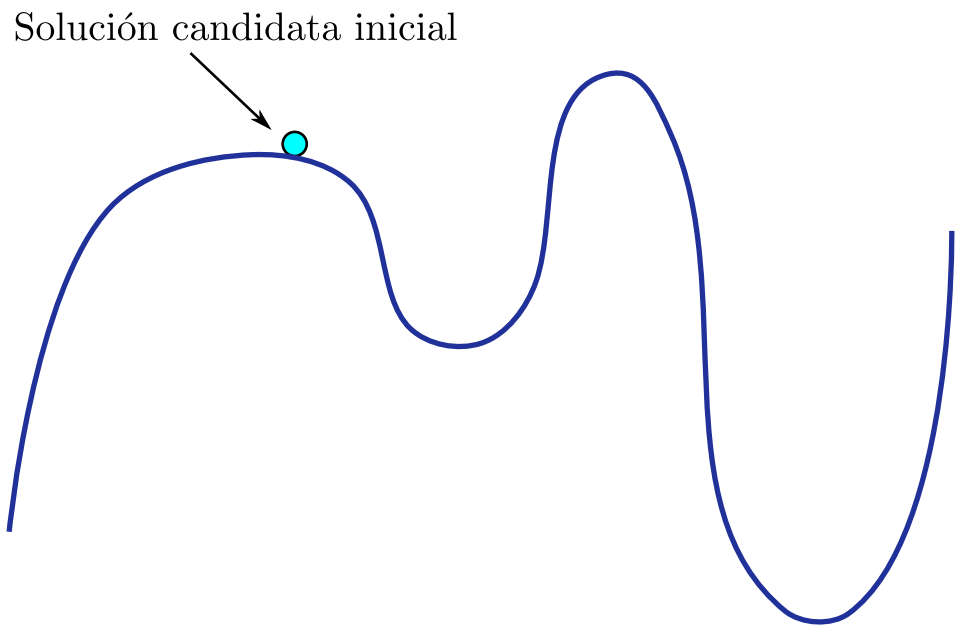
Encontrar los valores extremos, máximos o mínimos, de una función objetivo.



Optimización de funciones

Algoritmos de búsqueda local

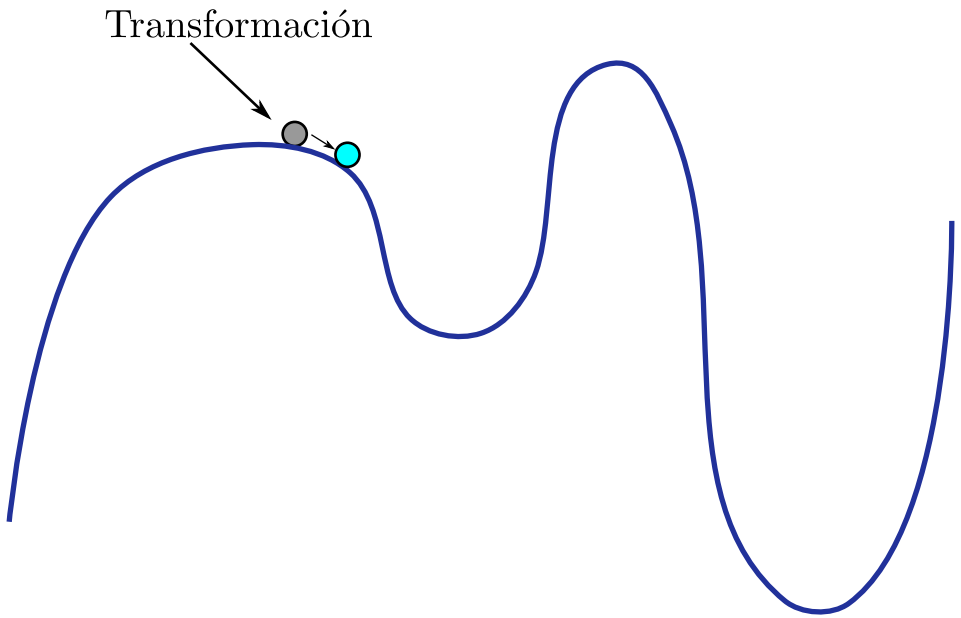
Exploran el espacio de soluciones haciendo transformaciones locales a las soluciones candidatas.



Optimización de funciones

Algoritmos de búsqueda local

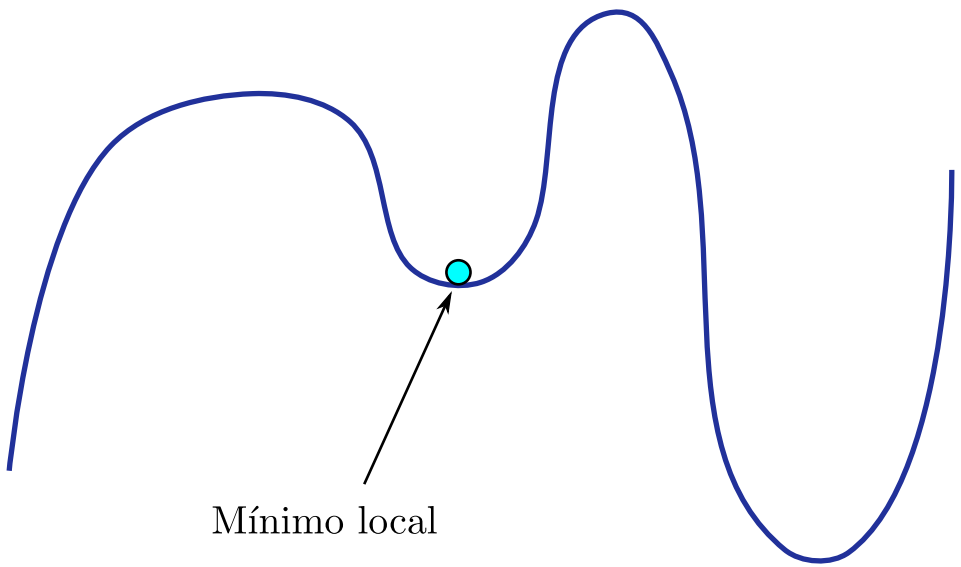
Exploran el espacio de soluciones haciendo transformaciones locales a las soluciones candidatas.



Optimización de funciones

Algoritmos de búsqueda local

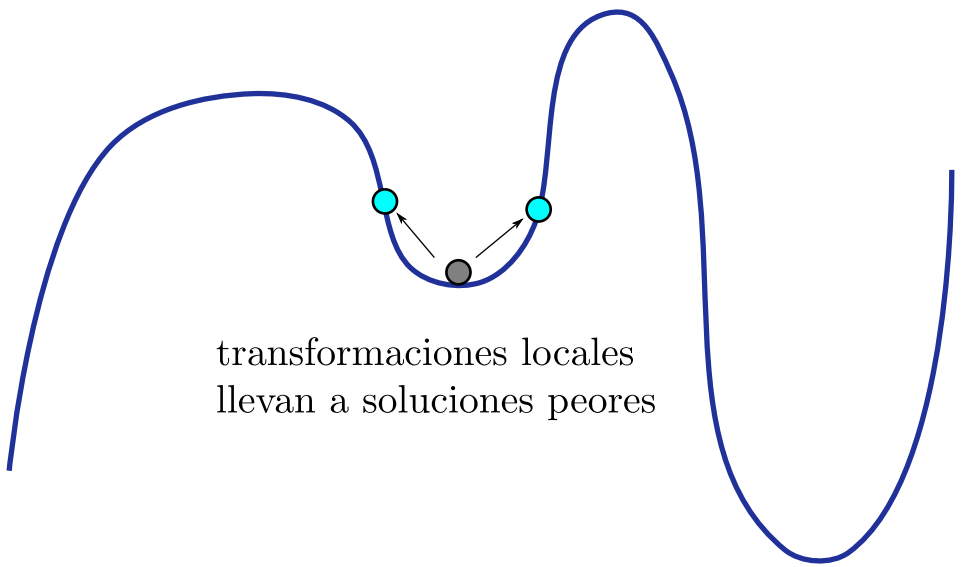
Exploran el espacio de soluciones haciendo transformaciones locales a las soluciones candidatas.



Optimización de funciones

Algoritmos de búsqueda local

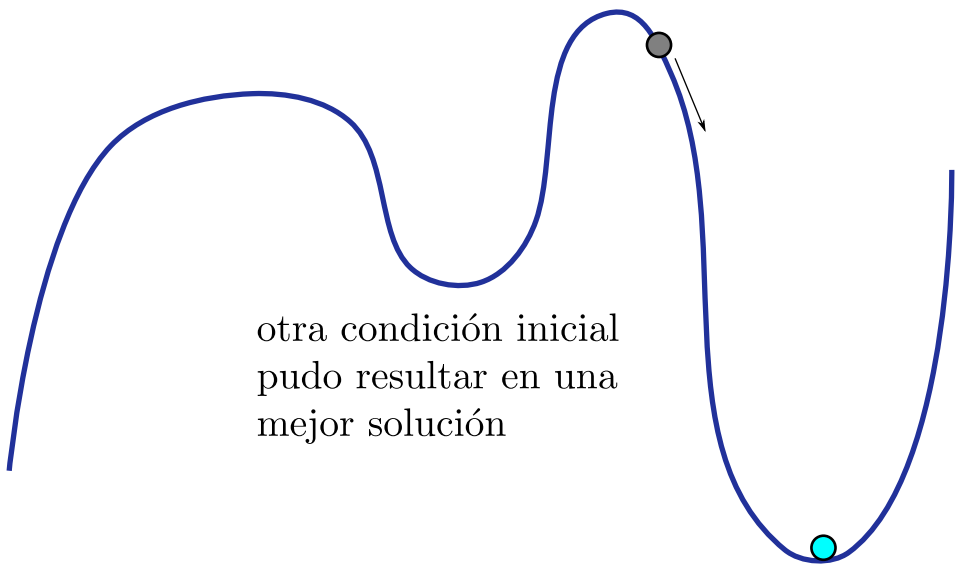
Exploran el espacio de soluciones haciendo transformaciones locales a las soluciones candidatas.



Optimización de funciones

Algoritmos de búsqueda local

Exploran el espacio de soluciones haciendo transformaciones locales a las soluciones candidatas.

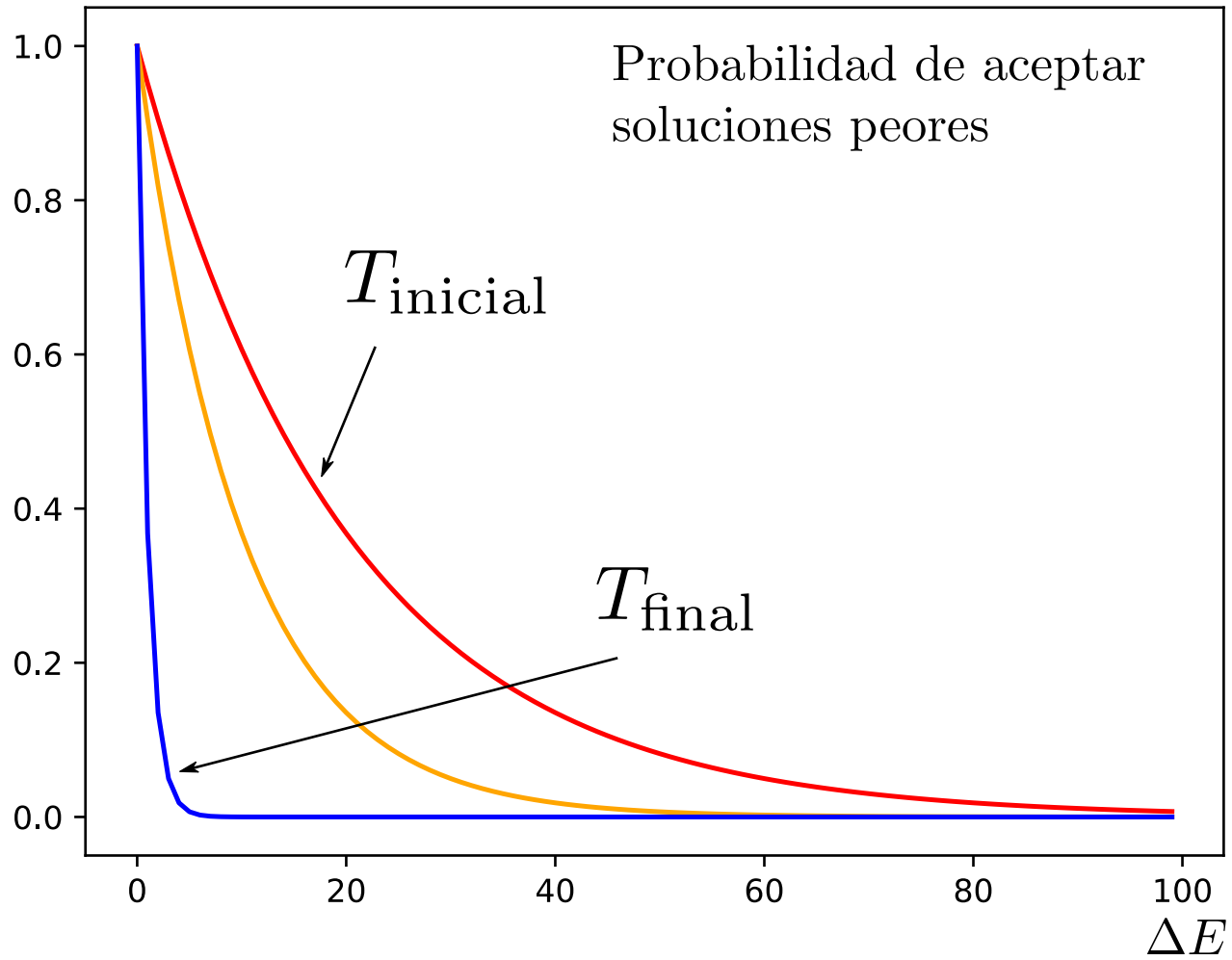


El algoritmo de recocido simulado (SA)

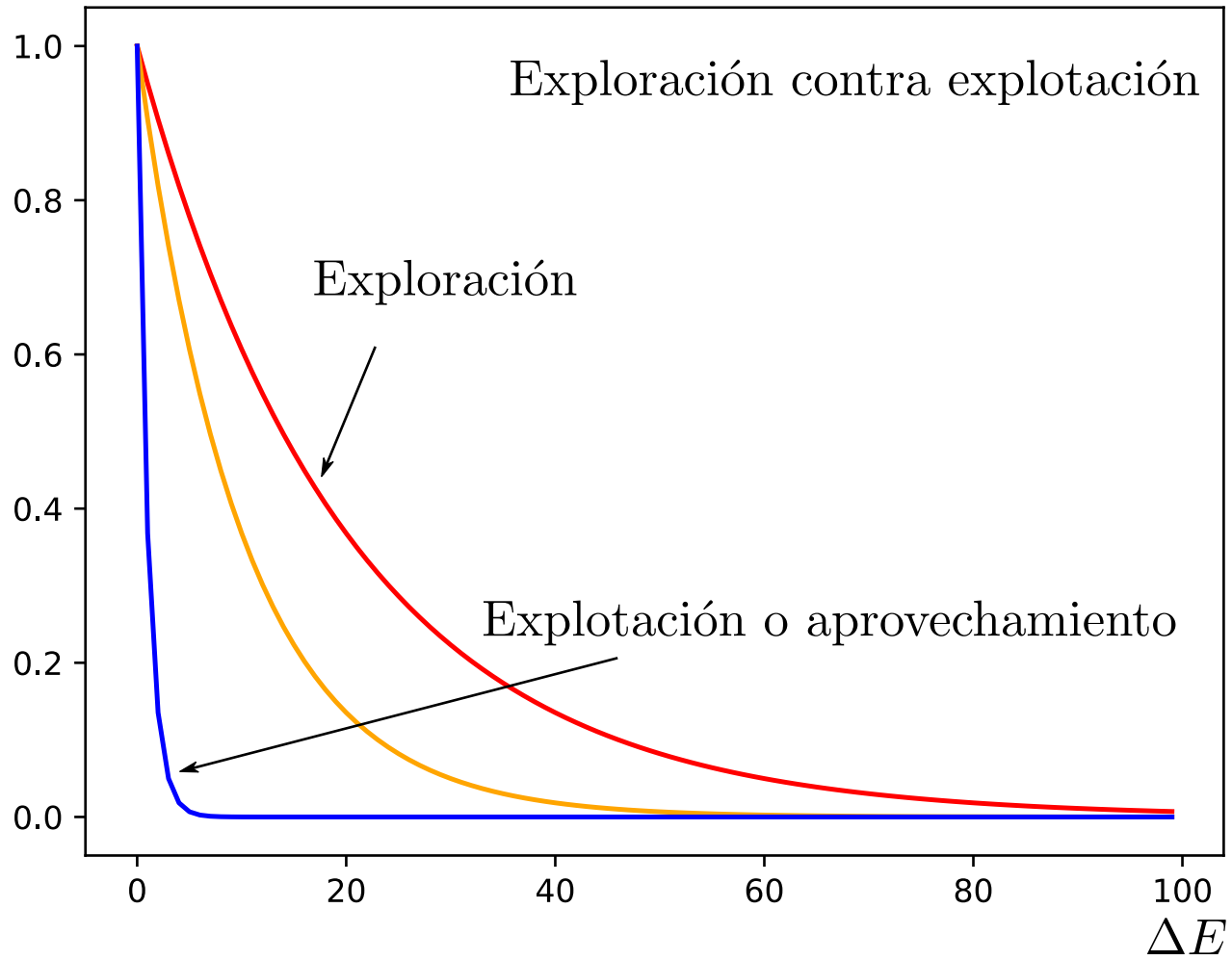
Un algoritmo capaz de escapar óptimos locales

El algoritmo de recocido (o templado) simulado (*simulated annealing*) es un algoritmo metaheurístico capaz de escapar de óptimos locales. Esta inspirado en un proceso de calentamiento y enfriamiento controlado en el cual un sólido puede alcanzar un arreglo cristalino muy regular al ser enfriado de manera suficientemente lenta.

Soluciones peores pueden aceptarse



Soluciones peores pueden aceptarse



Algoritmo de templado simulado

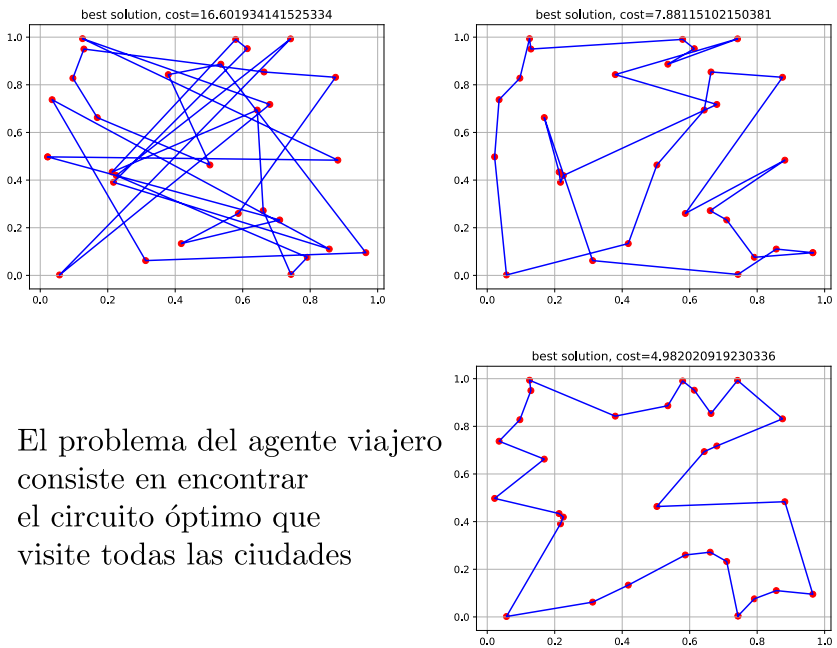
Algoritmo: SA

Entradas: T_i ; // temperatura inicial
T_f ; // temperatura final
c ; // constante de enfriamiento
n ; // número de movimientos hasta el equilibrio térmico
muestrear ; // muestra aleatoria en el espacio de soluciones
mover : $S \rightarrow S$; // vecino aleatorio de una solución
energia : $S \rightarrow \mathbb{R}$; // energía de una solución
Salida : Solución subóptima
$T \leftarrow T_i$; // Se inicializa la temperatura
$x \leftarrow$ muestrear() ; $e \leftarrow$ energia(x) ; // solución candidata inicial y su energía
mientras $T > T_f$ **hacer**
  $i \leftarrow 1$;
  mientras $i \leq n$ **hacer**
    $x' \leftarrow$ mover(x) ; $e' \leftarrow$ energia(x') ; // solución candidata vecina y su energía
    si $e' < e$ **entonces**
      $(x, e) \leftarrow (x', e')$; // acepta un estado de menor energía
    en otro caso
      $r \leftarrow aleatorio.uniforme(0, 1)$; // r es un número aleatorio entre 0 y 1
      si $r < exp\left(\frac{-(e'-e)}{kT}\right)$ **entonces**
        $(x, e) \leftarrow (x', e')$; // acepta un estado de mayor energía
      fin
    $i \leftarrow i + 1$;
  fin
  $T \leftarrow cT$; // se enfria el sistema
fin
devolver x

Ejemplo de aplicación de SA a un problema de permutaciones

El problema del agente viajero

Un agente viajero debe visitar exactamente n ciudades. Se desea encontrar el orden en que dichas ciudades deben visitarse tal que el costo sea mínimo.



El problema del agente viajero consiste en encontrar el circuito óptimo que visite todas las ciudades

Soluciones candidatas y vecinas

Solución candidata

Una solución candidata es una permutación de ciudades:

$$x = \begin{bmatrix} B & C & G & H & J & D & A & E & G & F \end{bmatrix}$$

Solución vecina

1. intercambiando dos ciudades adjacentes:

$$x = \begin{bmatrix} B & C & G & H & J & D & A & E & G & F \end{bmatrix}$$
$$x = \begin{bmatrix} B & C & G & J & H & D & A & E & G & F \end{bmatrix}$$

2. intercambiando dos ciudades cualesquiera:

$$x = \begin{bmatrix} B & C & G & H & J & D & A & E & G & F \end{bmatrix}$$
$$x' = \begin{bmatrix} B & C & G & G & J & D & A & E & H & F \end{bmatrix}$$

3. invirtiendo el orden de una subruta:

$$x = \begin{bmatrix} B & C & G & H & J & D & A & E & G & F \end{bmatrix}$$
$$x' = \begin{bmatrix} B & C & G & A & D & J & H & E & G & F \end{bmatrix}$$

Función de energía

Energía de una solución candidata

La energía es el costo de la ruta.

$$\text{energía}(x) = \sum_{i=1}^{n} \text{costo}(x_i, x_{(i \mod n)+1})$$

Búsquedas metaheurísticas

[Gendreau y Potvin 2010; *Handbook of metaheuristics*]

Las metaheurísticas, son métodos de solución que orquestan procedimientos de mejora local con estrategias de alto nivel para crear un proceso capaz de escapar de un óptimo local y desempeñar una búsqueda robusta en el espacio de soluciones.

Búsquedas metaheurísticas

[Gendreau y Potvin 2010; *Handbook of metaheuristics*]

Las metaheurísticas, son métodos de solución que orquestan procedimientos de mejora local con estrategias de alto nivel para crear un proceso capaz de escapar de un óptimo local y desempeñar una búsqueda robusta en el espacio de soluciones.

El término fue acuñado por Fred Glover en 1986 para categorizar su algoritmo *Búsqueda Tabú (Tabu Search (TS))*.

Metáforas

Matemáticas + Inspiración

Las metaheurísticas muchas veces combinan técnicas formales con estrategias informales observadas en procesos naturales (incluye humanos y sus artefactos).

Parten de la aceptación de que el espacio de búsqueda es computacionalmente intratable. Es decir, de que no conocemos si existen algoritmos eficientes para resolverlos de manera exacta. Por lo tanto, se vale utilizar métodos informales, estratégias algoritmicas que subjetivamente intuímos pueden funcionar para abordar el problema a resolver.

Los algoritmos admiten *hibridación* de técnicas.

¿Cuándo usar algoritmos metaheurísticos?

La última línea de defensa

En palabras del investigador Carlos Coello Coello del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, "los algoritmos metaheurísticos constituyen la última línea de defensa en optimización".