



Sistemas de Inteligencia Artificial

Algoritmos de Mejoramiento Iterativo





Algoritmos de Mejoramiento Iterativo

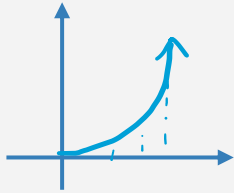
¿Qué significa que sean de
Mejoramiento Iterativo?





Algoritmos de Mejoramiento Iterativo

¿Qué significa que sean de
Mejoramiento Iterativo?

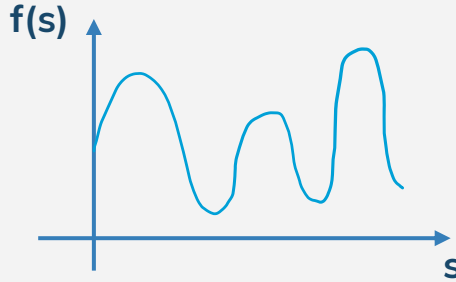


Parto de una solución y busco (mediante perturbaciones a mi solución inicial) una solución mejor en cada paso





Algoritmos de Mejoramiento Iterativo



s: Una solución a nuestro problema

f: Función que evalúa que tan buena es una solución particular

El objetivo de estos algoritmos es hallar:

$$s' \mid \forall s \quad f(s') \geq f(s)$$

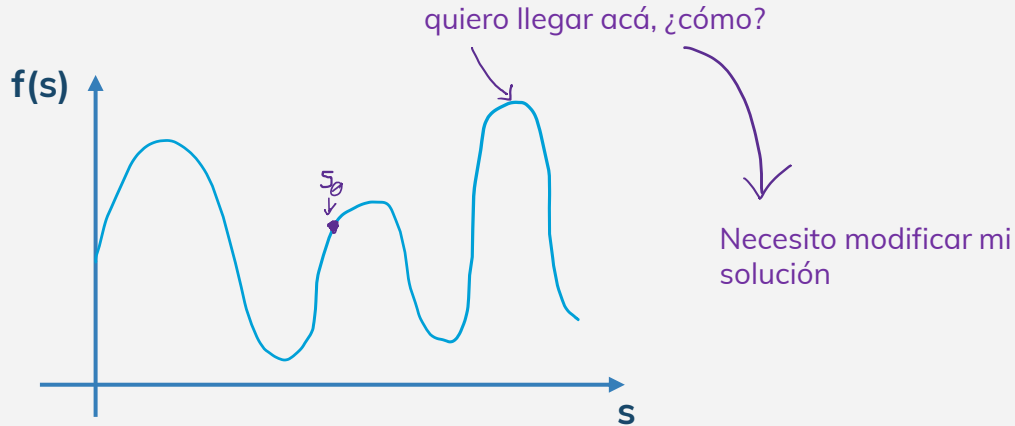
(la solución que maximiza mi función f)





Algoritmos de Mejoramiento Iterativo

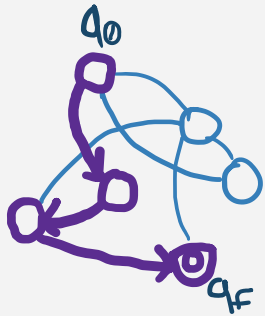
Para hallar esta solución óptima, realizamos **cambios azarosos** en nuestra solución original que nos den como resultado una mejor solución.



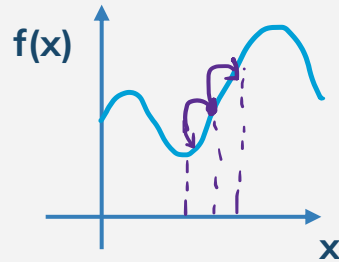


Algoritmos de Mejoramiento Iterativo

Es importante definir qué es un **cambio azaroso** en el problema particular que estoy queriendo optimizar. Necesito definir un **entorno** de soluciones.



Puede ser un cambio en los nodos que recorro en un grafo



Puede ser moverse un delta en todas las dimensiones de una función (¡o sólo en algunas!)

$$g(\nabla f)$$

Puede ser en base a una función del gradiente de una función (dirección de máximo crecimiento)

	s_1	
s_2	s_0	s_3
	s_4	

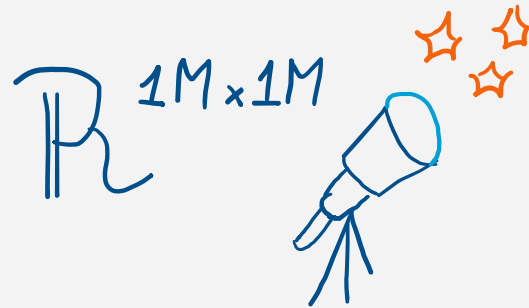
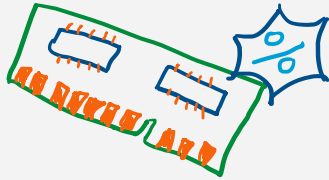
Puede ser un cambio en la posición de una matriz





Algunas características

- Usan muy **poca memoria**, incluso memoria constante.
- A veces son apropiados para espacios de **estados grandes o infinitos**, o para encontrar **soluciones parciales** rápidamente.





Travelling Salesman Problem

Problema del Vendedor Viajero

El Problema del Vendedor Viajero es un problema de recorrido de grafos donde nos dan n ciudades y la distancia entre cada una de ellas. Tenemos que encontrar la ruta más corta, visitando cada ciudad exactamente una vez y al final regresar a la ciudad de la que partimos.

Input

N = número de ciudades
 N líneas de la forma "label x y"

dónde:

label = un string

x = coordenada x de la ciudad (número real)

y = coordenada y de la ciudad (número real)

Output

Un camino óptimo con cada camino separado por espacios

Ejemplo 1:

4

```
0 0 0
1 0 1
2 1 0
3 1 1
```

Ejemplo 2:

5

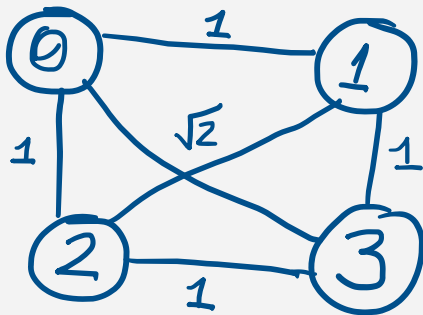
```
0 0 0
1 0 1
2 1 0
3 1 1
4 0.5 2
```





Travelling Salesman Problem

Problema del Vendedor Viajero



- 4º Elijo una en base a su valuación
sol = 0-1-3-2
 $f(sol) = 4$

- 1º Tomo una solución al azar
Por ejemplo: sol = 0-3-1-2(-0)

- ↓
2º Evaluo mi solución
 $f(sol) \approx 4,83$

- ↓
3º Realizo cambios azarosos en mi solución

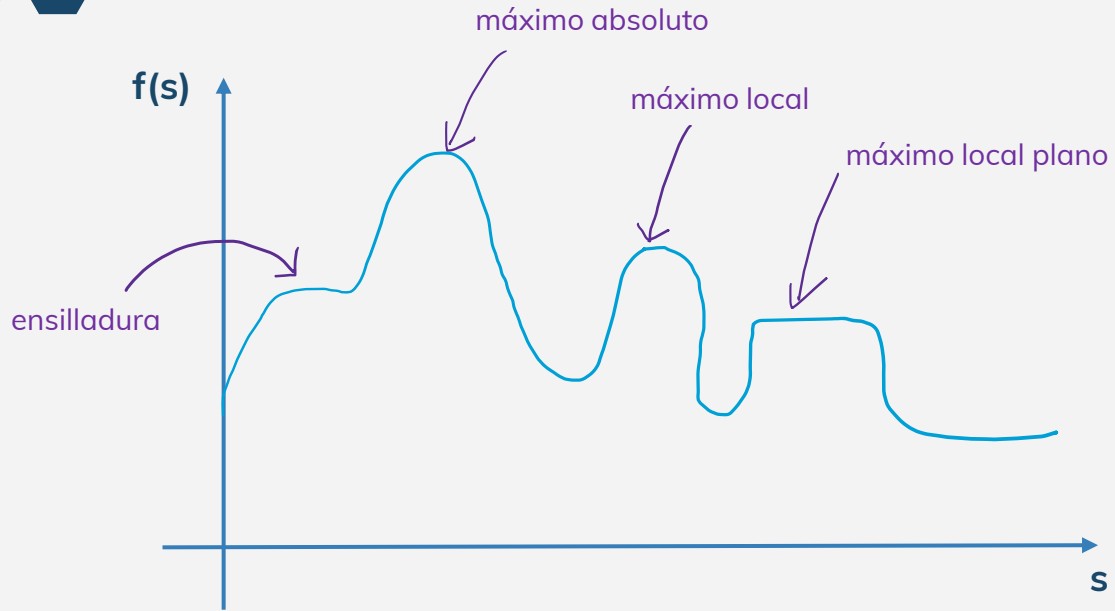
Puede ser: Elijo una ciudad y calculo las permutaciones de la misma

Por ejemplo: Elijo la ciudad 3,
las permutaciones son:
[3-0-1-2, 0-1-3-2, 0-2-1-3]

Freno cuando tengo una solución que me sirve o es la mejor de su vecindad



Hill Climbing



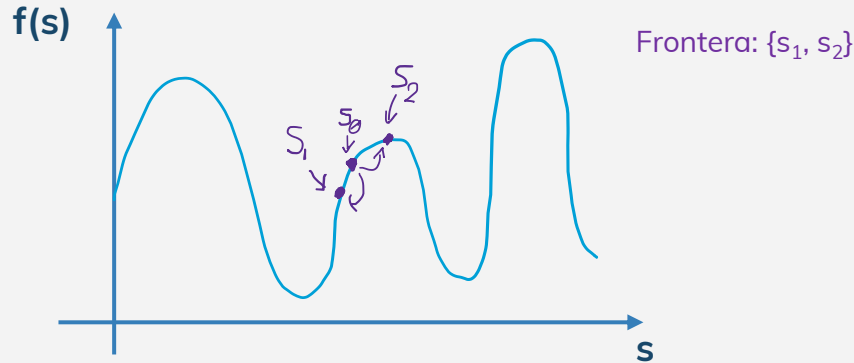
- ◇ Se mueve constantemente en la pendiente positiva.
- ◇ Se estanca fácilmente en máximos locales.





Hill Climbing

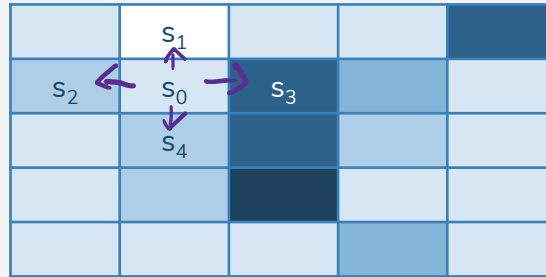
- Tomar una solución al azar.
- Moverse un paso arbitrario en ambas direcciones de cada una de las dimensiones. Estas nuevas soluciones nos generan una **Frontera**.
- Tomar una solución de la frontera con mejor valoración.
- ITERAR





Hill Climbing

- Tomar una solución al azar.
- Moverse un paso arbitrario en ambas direcciones de cada una de las dimensiones. Estas nuevas soluciones nos generan una **Frontera**.
- Tomar una solución de la frontera con mejor valoración.
- ITERAR



Frontera: $\{s_1, s_2, s_3, s_4\}$





Algunos tipos de Hill Climbing

Simple Hill Climbing: Elige la primer solución de la frontera con mejor valuación que el estado actual.

	s_1	
s_2	s_0	s_3
	s_4	

Analiza s_1 y s_2 , elige s_2

Steepest Hill Climbing: Siempre elige la solución de la frontera con mejor valuación que el estado actual.

	s_1	
s_2	s_0	s_3
	s_4	

Analiza s_1 , s_2 , s_3 y s_4 , elige s_3





Algunos tipos de Hill Climbing

Stochastic Hill Climbing: Elige al azar una solución de la frontera y, en base a que tan buena es, decide si moverse o elegir otra.

	s_1	
s_2	s_0	s_3
	s_4	

Elige s_4 , decide moverse a dicha solución





Hill Climbing

¡Hill Climbing es un algoritmo del tipo Heurística Local por lo que se genera una nueva **frontera** en cada paso!

(No confundir frontera de métodos de búsqueda con la **frontera** de Hill Climbing)

Hill Climbing

Posibles sucesores de la solución actual

VS

Métodos de Búsqueda

Los nodos sucesores de los nodos expandidos





Simulated Annealing

Parecido al Hill Climbing, pero involucra una **temperatura** (inicialmente alta). Esto provoca que al principio se puedan tomar peores soluciones.

A medida que pasan las iteraciones, esta temperatura disminuye arbitrariamente según se la configure.

Entonces cada vez que toma un nodo de **Frontera**, ignora su valuación con una probabilidad relativa a la temperatura (y, opcionalmente, también proporcional a su valuación).





Simulated Annealing



Pero $f(s_1) < f(s_2)$





Simulated Annealing

Sea $f(s)$ la función a maximizar s y $E(s)$ su entorno.

Definir:

- s_0 : solución inicial.
- t_0 : temperatura inicial (> 0)
- α : función de reducción de temperatura
- N_{rep} : Número de repeticiones
- P : Criterio de parada





Simulated Annealing

- Tomar $s=s_0$, $t = t_0$
- Repetir mientras que no se cumpla **P**:
 - Repetir N_{rep} veces:
 - Seccionar s' perteneciente a **E(s)**
 - $d = f(s') - f(s)$
 - Si $d > 0$:
 - Tomo s' (es una mejor solución)
 - Si no:
 - $u = \text{rand}(0, 1)$
 - Si $u < e^{(-d/t)}$:
 - $s = s'$
 - Luego de repetir N_{rep} veces:
 - $t = \alpha(t)$ (reduzco la temperatura)





Beam Search

Similar a Hill Climbing, pero comienza con **k** nodos iniciales.

En cada paso, se generan los sucesores de los **k** nodos, que pasarán a conformar la frontera.

De todos los sucesores, se toman los mejores **k** nodos de la frontera y se vuelve a repetir.

Es diferente a **k** hill climbings en paralelo, ya que puede abandonar rápidamente los nodos iniciales malos para tomar otros caminos mejores.

Puede converger mucho entre los **k** nodos, generando tan poca diversidad que se elabora un hill climbing mucho más costoso.

k se conoce como el ancho del rayo o beam width

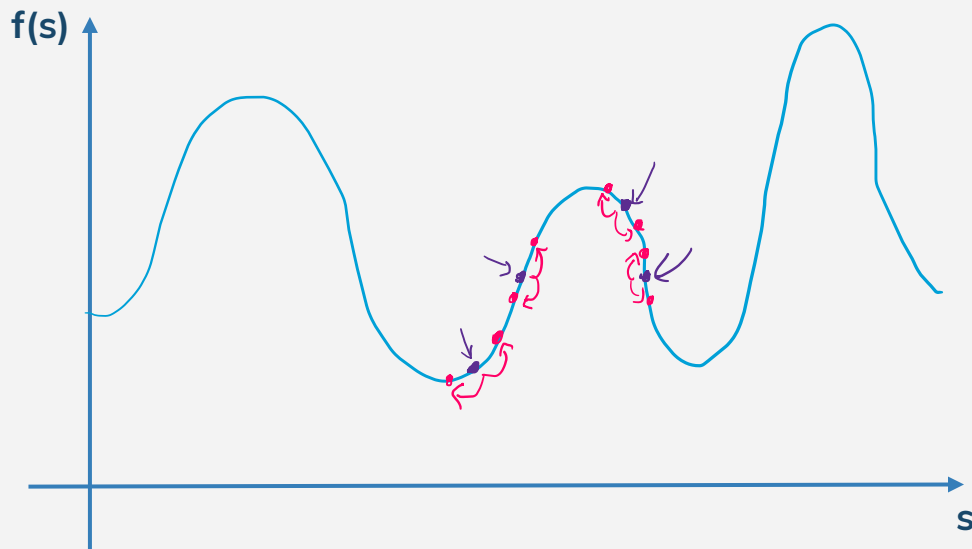



Beam Search



Beam Search

Ejemplo de convergencia entre los k nodos



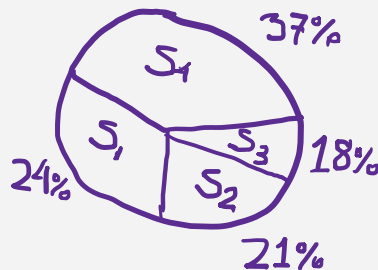
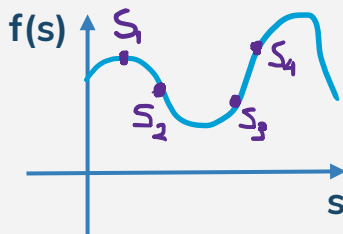


Stochastic Beam Search

Similar a Beam Search, pero elije a los k nodos con una probabilidad proporcional a su valuación.

De esta forma, los nodos “malos” también pueden llegar a elegirse (con menor probabilidad).

Similar al proceso de selección natural.



Random entre $[0, 1]$

$[0.00, 0.37) : s_4$

$[0.37, 0.61) : s_1$

$[0.61, 0.82) : s_2$

$[0.82, 1.00) : s_3$





Algoritmos Genéticos

Una variante de SBS donde los estados se generan como combinación de 2 o más estados previamente analizados.

De la misma forma que la evolución natural:

- Los estados corresponden a **individuos**.
- La función que mide la valuación de estos individuos se le llama **aptitud**.
- Combinando buenos individuos aumenta la probabilidad de que se generen mejores.

