Inteligencia Computacional

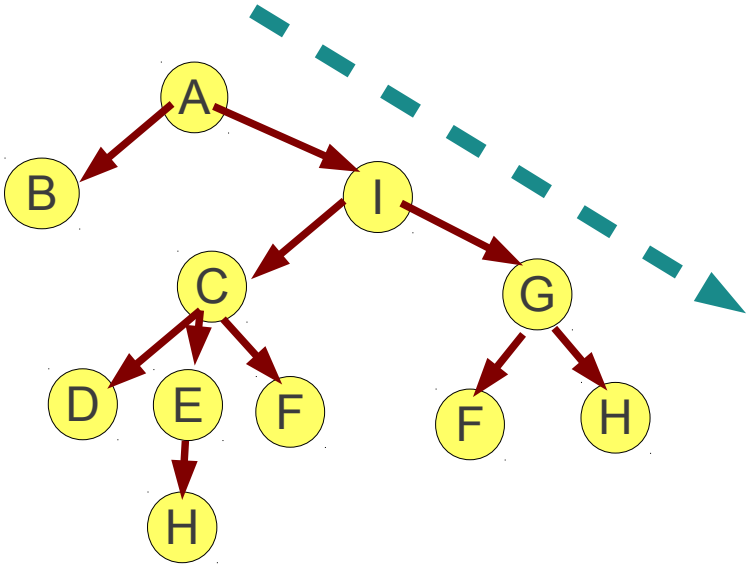
Búsqueda local: hill-climbing

http://blancavg.com/tc3023/

Blanca A. Vargas Govea * blanca.vg@gmail.com * Agosto 21, 2012

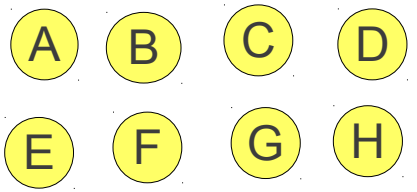
La solución es una secuencia de acciones

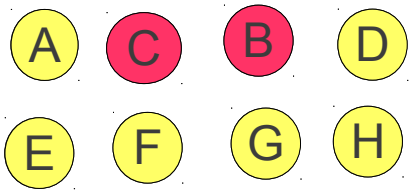
A-I-G-H

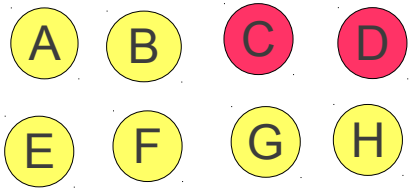


Métodos anteriores

En vez de explorar los caminos se evalúan y modifican uno o más estados





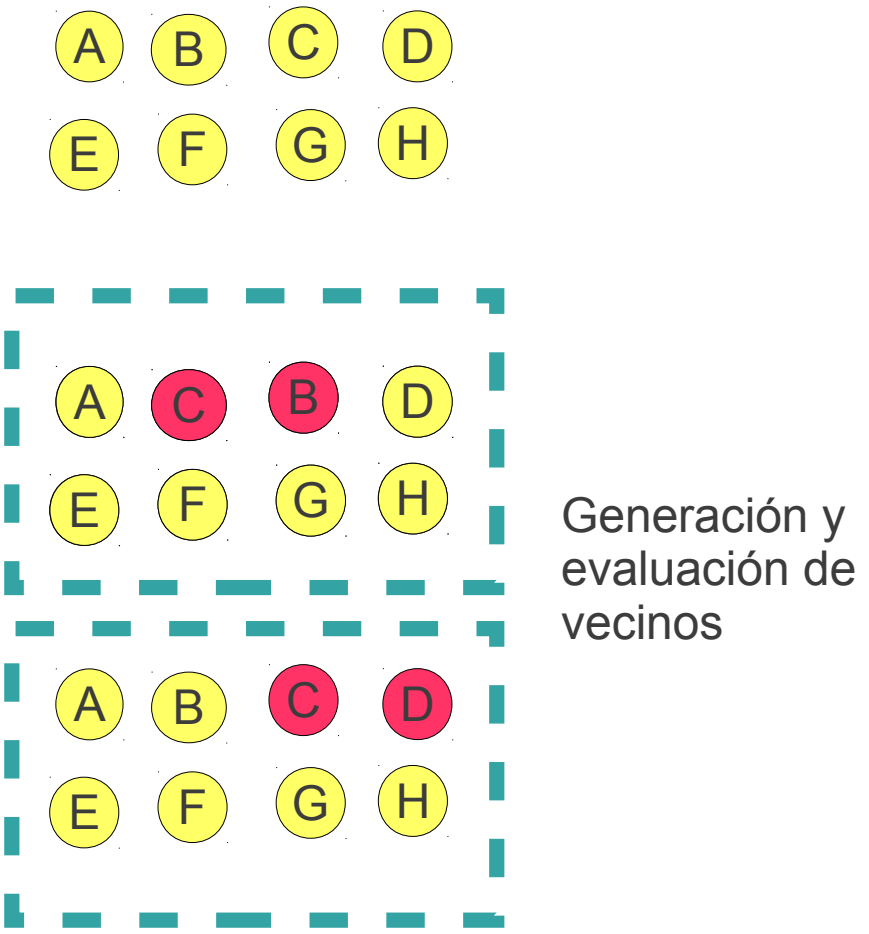


Los algoritmos son adecuados para problemas en los que importa el estado meta, no el camino

Búsqueda local

Los caminos no se almacenan

El movimiento es hacia vecinos del estado



Generación y evaluación de vecinos

Búsqueda local

simple

Selecciona la primer acción que mejora el estado actual

Algoritmo Hill climbing

steepest ascent

Evalúa a todos los vecinos y se mueve en la dirección en donde el valor aumenta (“cuesta arriba”)

Termina cuando alcanza un pico donde ningún vecino tiene mayor valor

Algoritmo Hill climbing

steepest ascent

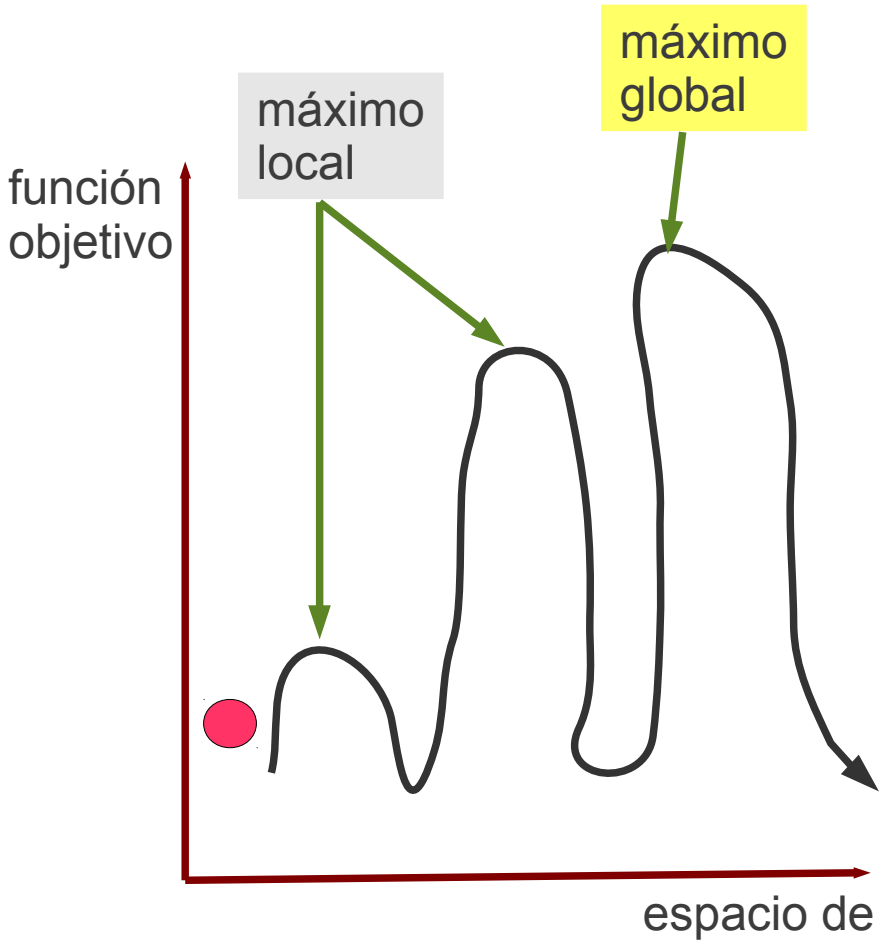
- No mantiene un árbol de búsqueda
- La estructura de datos del nodo actual solamente registra el estado y el valor de la función objetivo
- Si hay vecinos empatados, la estrategia más simple es seleccionar de forma aleatoria

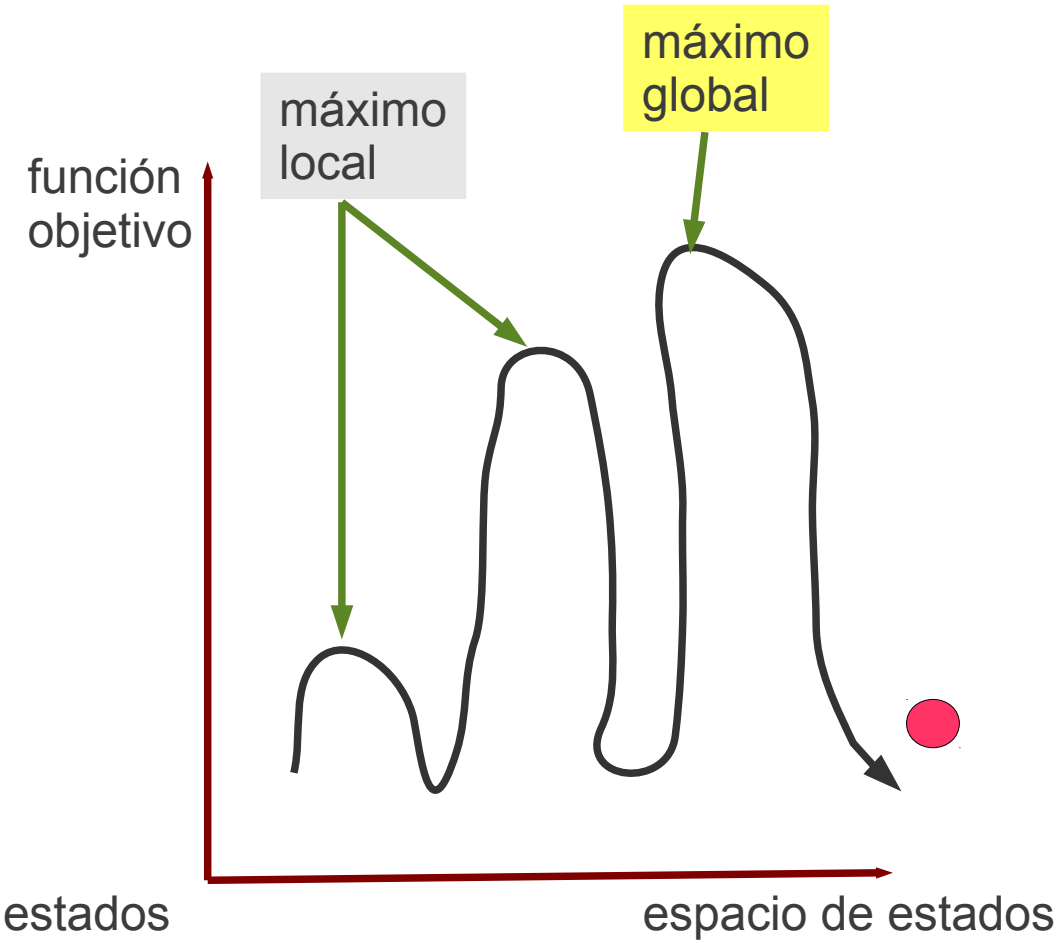
Algoritmo Hill climbing

steepest ascent

```
estado_actual = estado_inicial
loop
   Generar sucesores del estado_actual
   Obtener el sucesor con el valor más alto
   if valor(sucesor) < valor(estado_actual)
   then
      return estado_actual
   else
      estado_actual = sucesor
```

Algoritmo Hill climbing





Solución: óptimo local

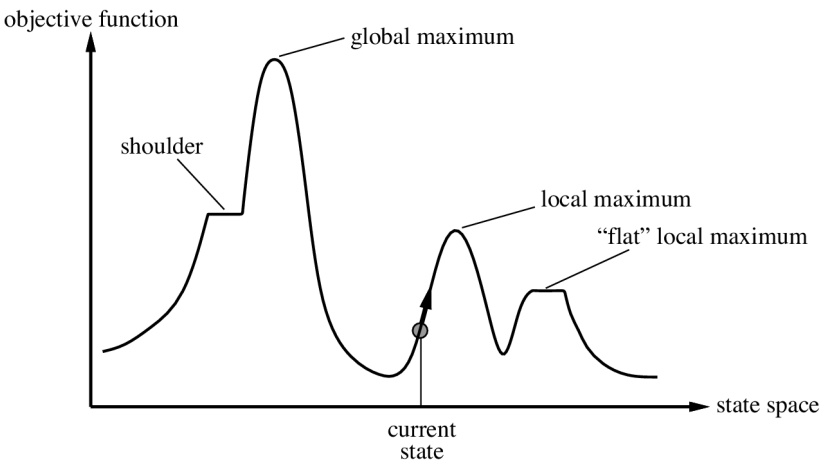
Solución: óptimo global

Algoritmo Hill climbing

Valor de la función objetivo

Posibles objetivos:

- Minimizar costo
- Maximizar el valor de la función



El estado inicial es importante

Panorama - espacio de stados

Razones de estancamiento

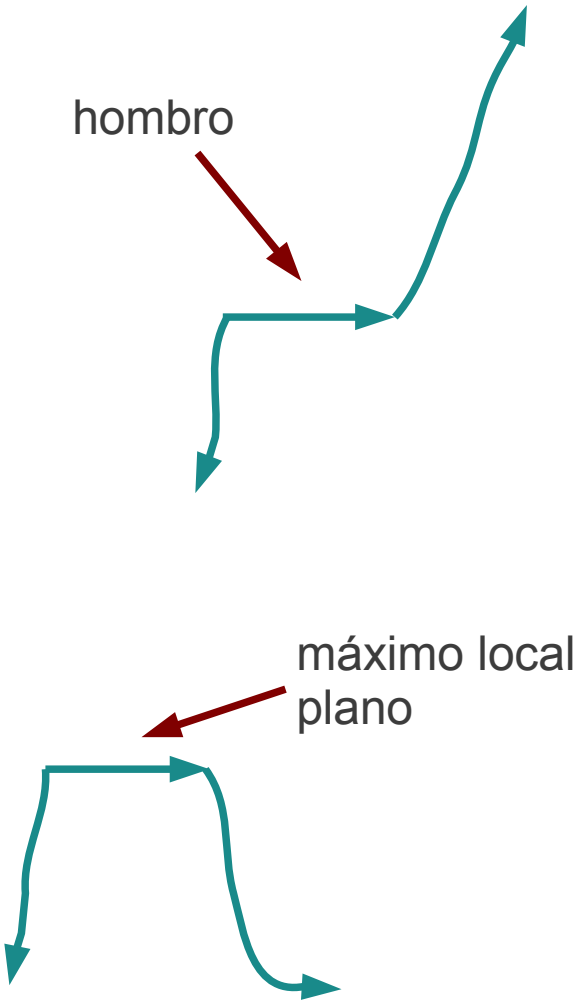
- Máximo local. Pico más alto que cualquier vecino pero menor al óptimo global.
- Cresta. Secuencia de máximos locales.
- Meseta. Área plana del espacio de estados (plano, hombro).

En cada caso, el algoritmo llega a un punto en el cual no puede alcanzar una mejor solución

Algoritmo Hill climbing

Saliendo de mesetas

- Se llega a una meseta cuando el vecino mejor evaluado tiene el mismo valor que el estado actual.
- Idea: movimientos laterales esperando que encuentre un mejor vecino.
- Riesgo: que no existan y se entre en un ciclo infinito.
- Posible solución: limitar el no. de movimientos laterales.



Algoritmo Hill climbing

Considera las 5 figuras geométricas de tamaño 1,2,3,4 y 5:

1 

2 

3 

4 

5 

Se da el estado inicial y el estado meta. Solamente se puede mover la pieza de arriba y usar 2 stacks adicionales.

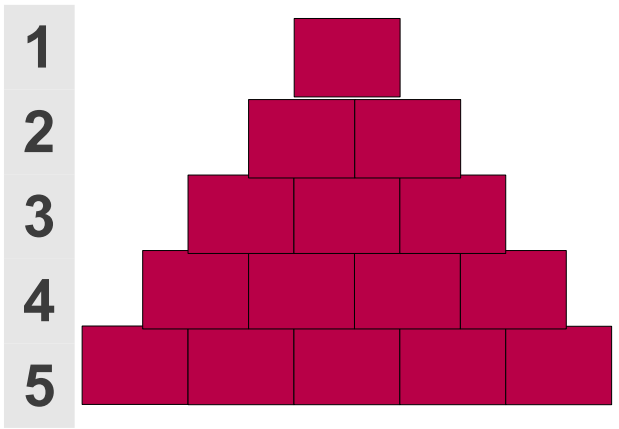
Ejemplo 1: acomoda los bloques

+1 por cada figura que esté sobre la figura correcta. El estado meta vale +5.

-1 por cada figura que esté en la figura incorrecta.

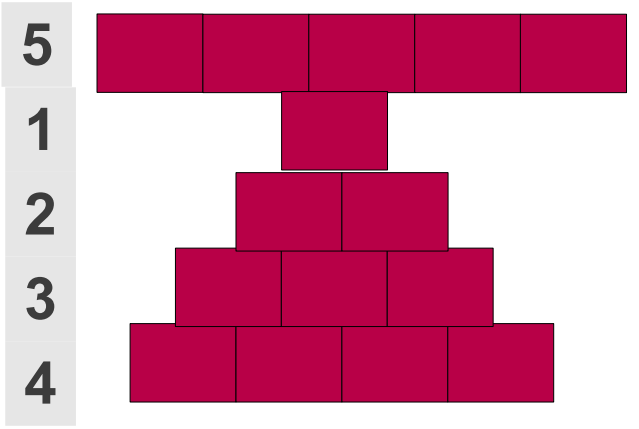
Ejemplo 1- heurística 1

Estado meta



1+1+1+1+1 = 5

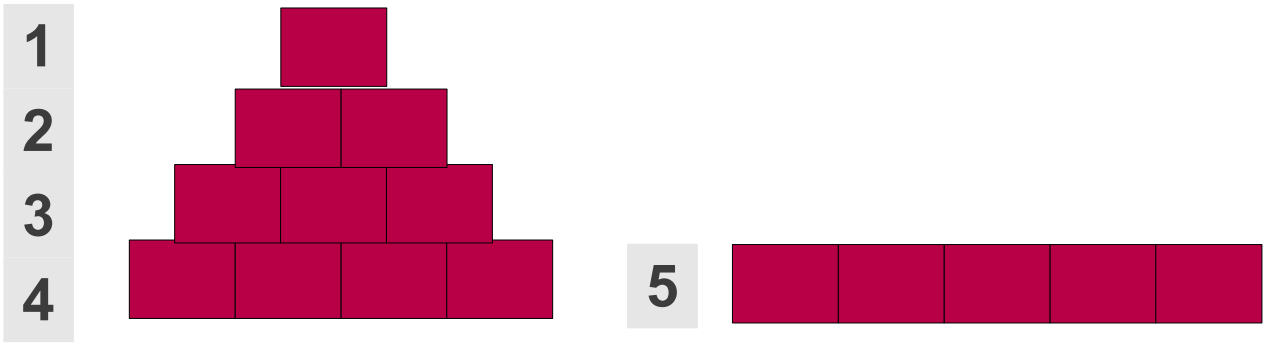
Estado inicial



-1+1+1+1-1 = 1

Ejemplo 1- heurística 1

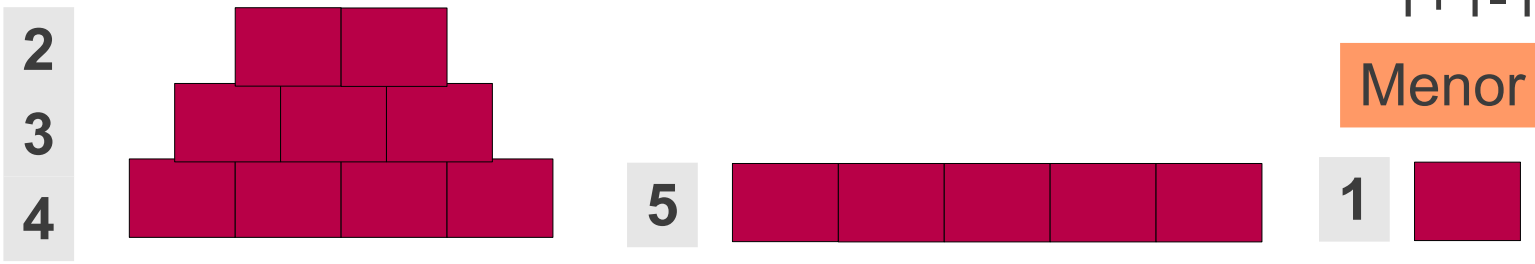
Movimiento 1



+1+1+1-1+1 = 3

Mejor evaluación que el estado inicial. Reemplaza al estado actual.

Movimiento 2a

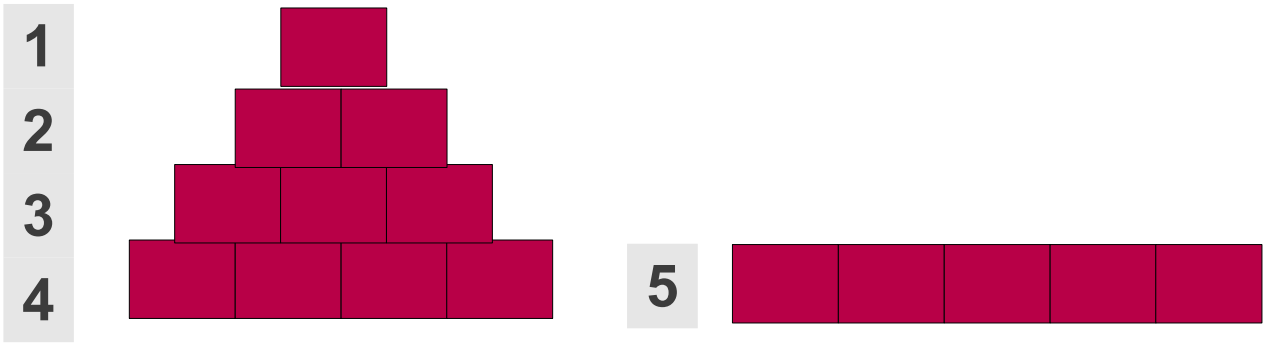


1+1-1+1-1 = 1

Menor al estado actual

Ejemplo 1- heurística 1

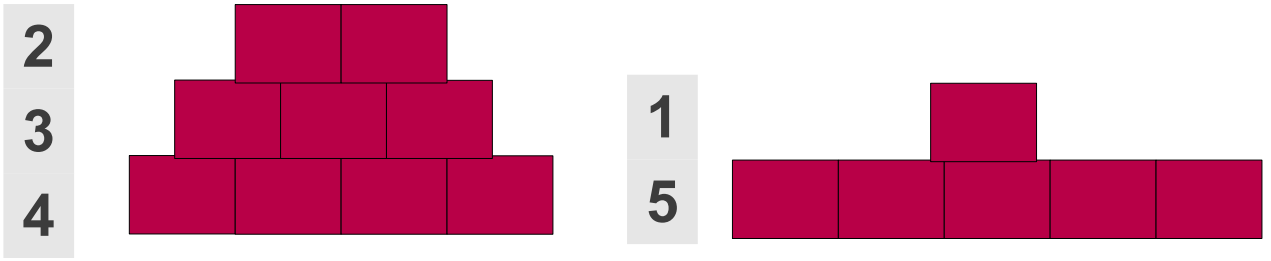
Movimiento 1



+1+1+1-1+1 = 3

Mejor evaluación que el estado inicial. Reemplaza al estado actual.

Movimiento 2b



1+1-1-1+1 = 1

Menor al estado actual

Para 2a y 2b la evaluación es menor que el estado inicial. El movimiento 1 es el mejor.

Se llega a un óptimo local

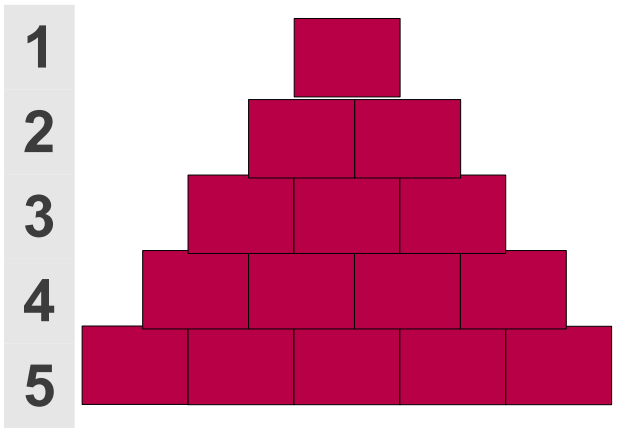
Ejemplo 1- heurística 1

+n por cada figura que esté en un grupo correcto de n figuras. El estado meta tiene el valor de 10.

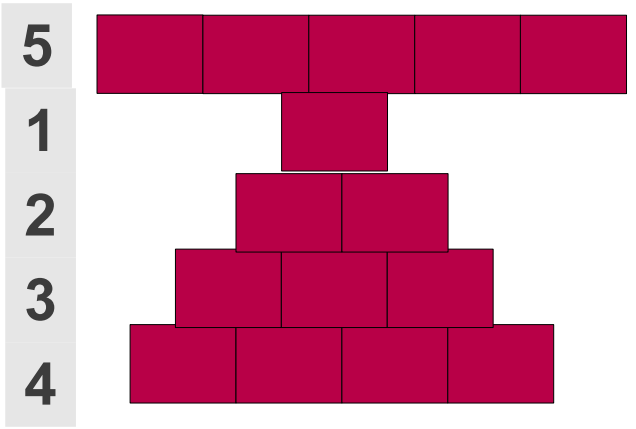
-n por cada figura que esté en un grupo incorrecto de n figuras.

Ejemplo 1- heurística 2

Estado meta



Estado inicial



4 está sobre 1 pieza correcta = 1
3 está sobre 2 piezas correctas = 2
2 está sobre 3 piezas correctas = 3
1 está sobre 4 piezas correctas = 4

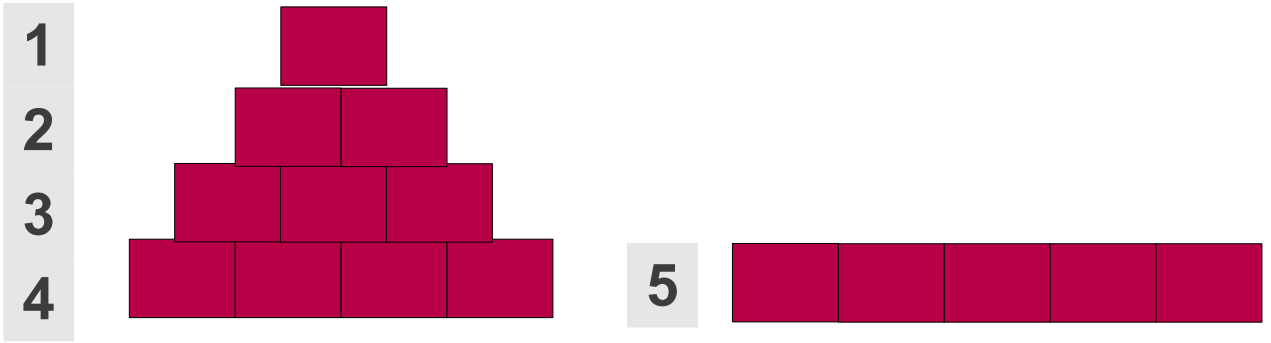
10

3 está sobre 1 pieza incorrecta = -1
2 está sobre 2 piezas incorrectas = -2
1 está sobre 3 piezas incorrectas = -3
5 está sobre 4 piezas incorrectas = -4

-10

Ejemplo 1- heurística 2

Movimiento 1

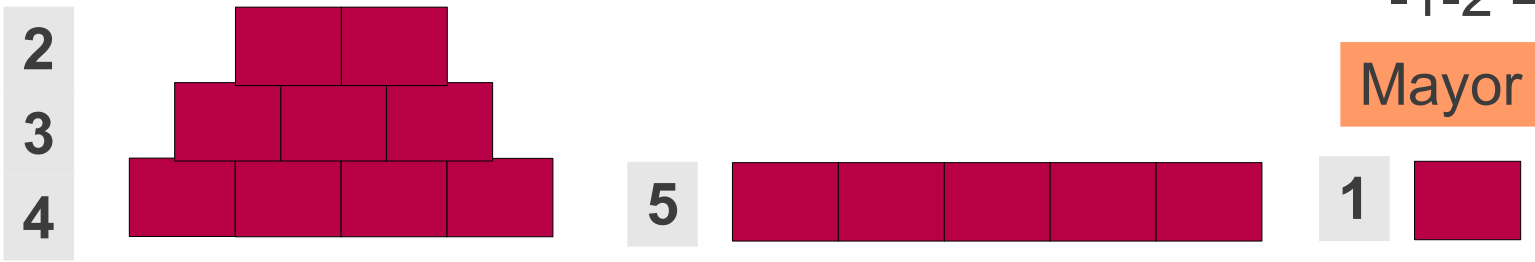


-1-2-3 = -6

Mejor evaluación que el estado inicial. Reemplaza al estado actual.

3 está sobre 1 pieza incorrecta = -1
2 está sobre 2 piezas incorrectas = -2
1 está sobre 3 piezas incorrectas = -3

Movimiento 2a

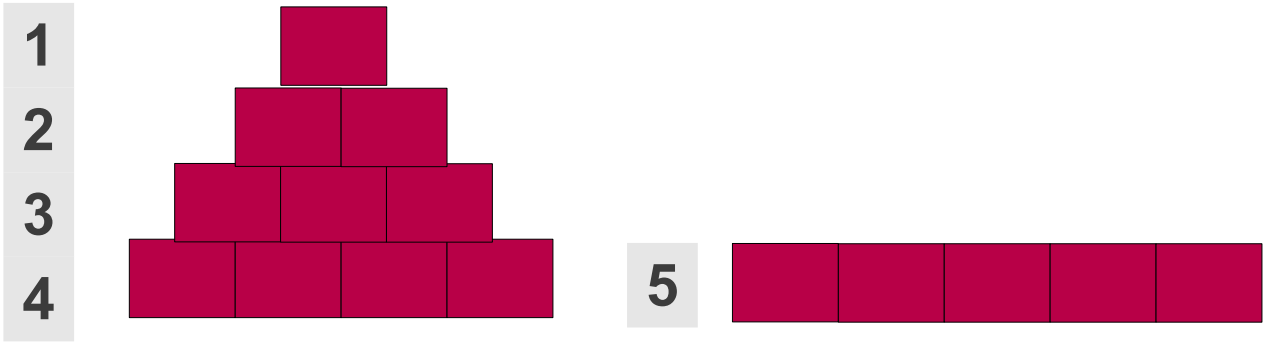


-1-2 = -3

Mayor al estado actual

Ejemplo 1- heurística 2

Movimiento 1

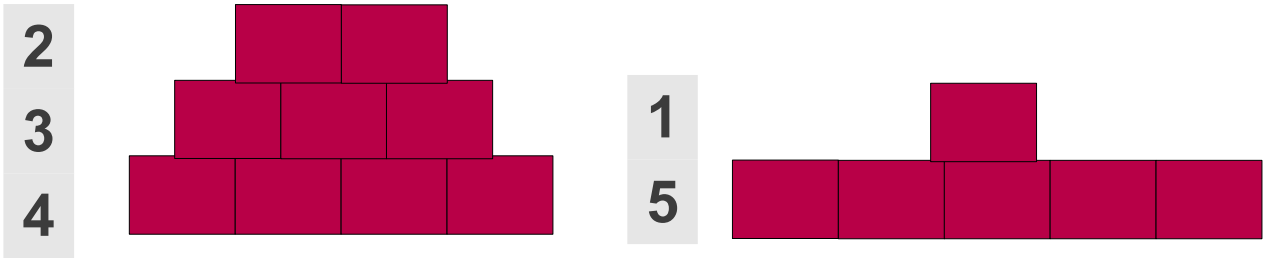


-1-2-3 = -6

Mejor evaluación que el estado inicial. Reemplaza al estado actual.

3 está sobre 1 pieza incorrecta = -1
2 está sobre 2 piezas incorrectas = -2
1 está sobre 3 piezas incorrectas = -3

Movimiento 2b

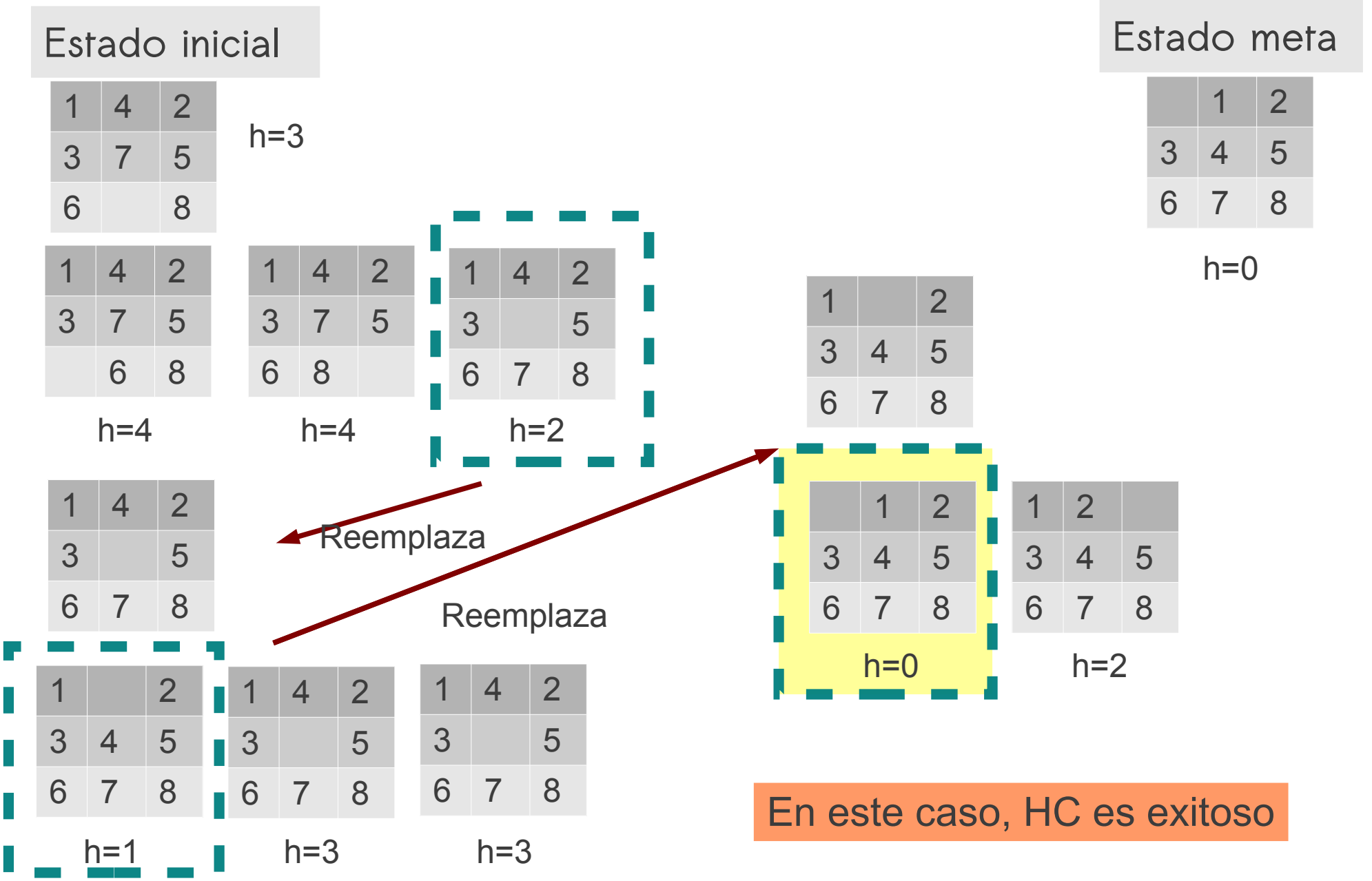


-2-1-1 = -4

Mayor al estado actual

Se evita el óptimo local

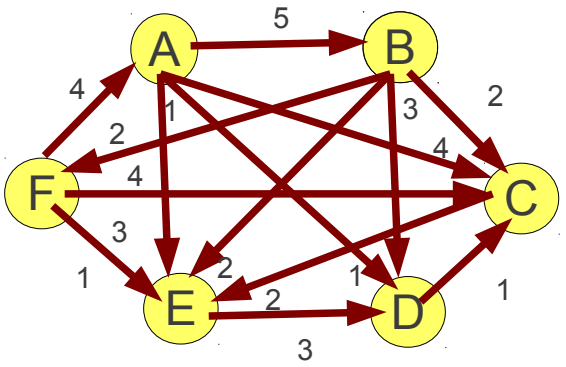
Ejemplo 1- heurística 2



Ejemplo 2 - 8 puzzle

Agente viajero. Traveling Salesman Problem (TSP).

Dado un conjunto de n ciudades y el costo del viaje entre cada par, el problema es encontrar la forma menos costosa de visitarlas todas, sin repetición y regresar al punto de partida.



Pueden usarse distintos operadores. El más simple: intercambiar el orden en que dos ciudades son visitadas.

Ejemplo 3: agente viajero

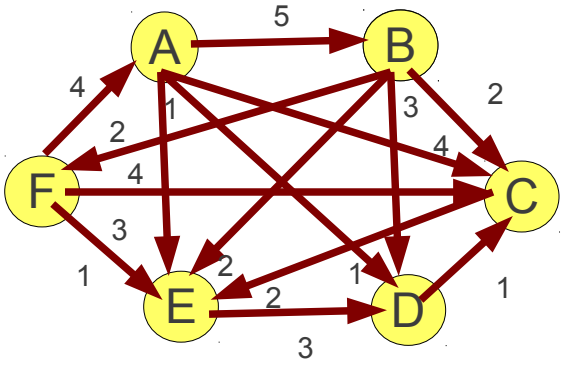
Estado inicial

ABCDEF (16)

ACBDEF (17)
ABDCEF (17)
ABCEDF (19)
ABCDFE (15) reemplaza

ABCDFE (15)

ACBDFE (16)
ABDCFE (17)
ABCFDE (20)
Ninguno es mejor.



Óptimo local: ABCDFE

Ejemplo 3: agente viajero. Óptimo local.

Si consideramos que cualquier par de ciudades puede intercambiarse al mismo tiempo:

ABCDEF (16)

ACBDEF (17)
ADCBEF (11)
AECDBF (15)
AFCDEB (19)
ABDCEF (16)
ABEDCF (19)
ABCEDF (17)
ABCFED (16)
ABCDFE (15)

ADCBEF (11)

ACDBEF (15)
AECBDF (17)
ADBCEF (13)
ADFBEC (14)
ADCFEB (14)
AFCBED (16)
ADEBCF (16)
ADCEBF (12)
ADCBFE (10)

Al reemplazar y continuar, no hay otro menor

Óptimo global: ADCBFE

Ejemplo 3: agente viajero. Óptimo global.

Stochastic hill: no examina a todos los vecinos, selecciona aleatoriamente a uno y con base en la mejora decide si revisa otro o se queda con ése.

Random restart hill climbing: realiza series de búsqueda hill climbing a partir de estados iniciales generados aleatoriamente hasta que se encuentra la meta.

Variantes

Ventajas

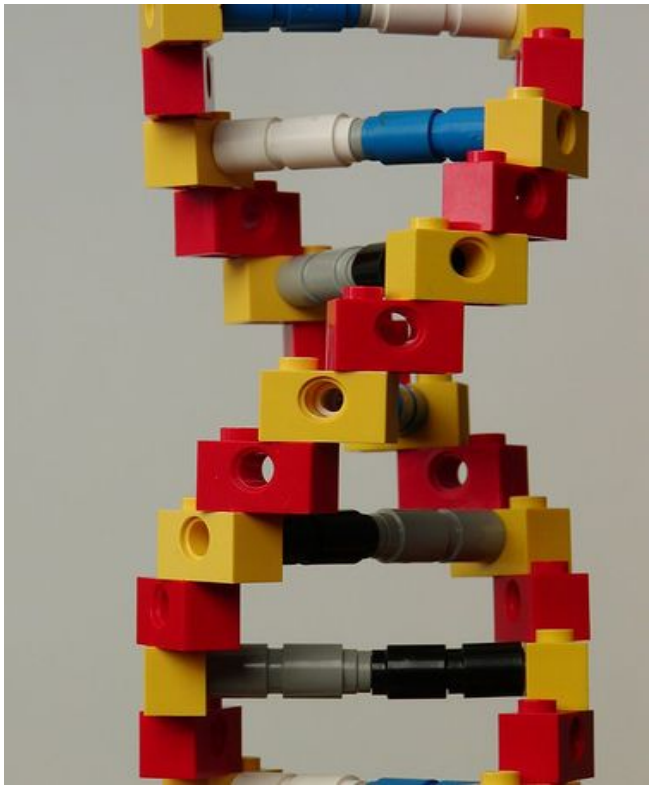
- Fácil de implementar, poca memoria
- Fácil para obtener una solución aproximada

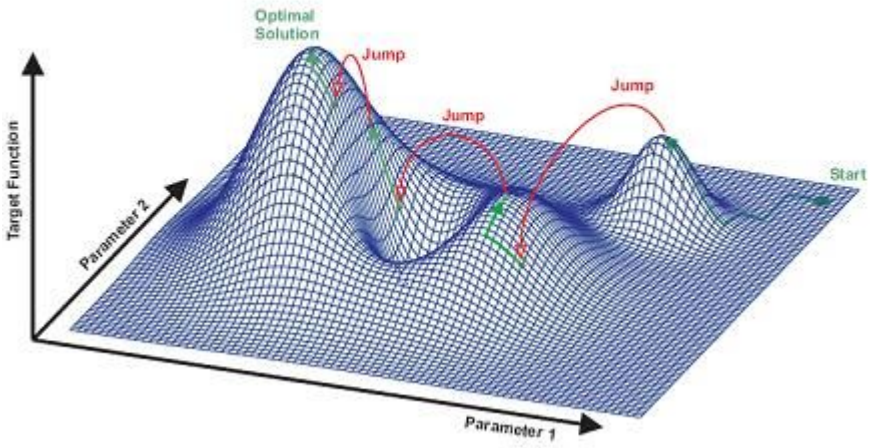
Desventajas

- El diseño de la función de evaluación puede ser difícil
- Si el no. de movimientos es muy grande puede ser ineficiente
- Si el no. de movimientos es pequeño puede estancarse fácilmente

Ventajas/desventajas de Hill Climbing

Recocido simulado





Algoritmos genéticos

Familia de algoritmos de búsqueda local

http://www.frankfurt-consulting.de/img/SimAnn.jpg

http://www.flickr.com/photos/42156072@N00/47457221/

Problema de los misioneros y caníbales.

3 misioneros y 3 caníbales están en la orilla izquiera de un río. Los 6 quieren cruzar el río. Un bote está disponible pero el bote solamente puede llevar 2 personas a la vez. Además, los misioneros no deben ser menos (en número) que los caníbales en ningún momento.

Ejercicio: resolver usando Hill Climbing

Referencias

Russell, S., y Norvig, P. (2003). Artificial intelligence: A modern approach (2nd edition ed.). Prentice-Hall, Englewood Cliffs, NJ.

Grosan C., y Abraham A. (2011) Intelligent Systems: A Modern Approach. Intelligent Systems Reference Library, Volume 17. Springer.