Sistemas Inteligentes

Búsqueda Local

Facultad de Informática
Universidad Nacional del Comahue

Segundo Cuatrimestre

Contenidos

Contenidos

Contenidos

Contenidos

Contenidos

¿Qué vimos?

- Definiciones de IA (pensamiento vs.comportamiento - humanidad vs. racionalidad).
- Sistemas Inteligentes

HOY

Búsqueda Local

¿Qué vimos?

- Definiciones de IA (pensamiento vs.comportamiento - humanidad vs. racionalidad).
- Sistemas Inteligentes

HOY

Búsqueda Local

Lo que IA dejó

Resolución de Problemas

Búsqueda

El proceso de buscar una secuencia de acciones que alcance una meta se llama Búsqueda.

Resolución de Problemas: Formulación

Un problema puede ser definido a través de los siguientes componentes:

- Estado Inicial: donde el agente comienza.
- Acciones: descripción de las acciones posibles de las que dispone el agente. Dado un estado s, ACTIONS(s) devuelve las acciones que pueden ser ejecutadas desde s. Decimos que cada una de estas acciones son aplicables a s.

Resolución de Problemas: Formulación

Un problema puede ser definido a través de los siguientes componentes:

- Modelo de Transición y Estados Sucesores: una descripción de lo que cada acción hace. La función RESULT(s, a) que devuelve el estado que resulta de hacer la acción a en el estado s y representa el modelo de transición.

  La función del estado sucesor de un estado se define como SUCCESSOR(x) y devuelve un par (a, s), donde a es la acción aplicada a x y s es el estado que se obtiene de aplicar a a x.

Resolución de Problemas: Formulación

Un problema puede ser definido a través de los siguientes componentes:

- Modelo de Transición y Estados Sucesores: una descripción de lo que cada acción hace. La función RESULT(s, a) que devuelve el estado que resulta de hacer la acción a en el estado s y representa el modelo de transición.
  La función del estado sucesor de un estado se define como SUCCESSOR(x) y devuelve un par (a, s), donde a es la acción aplicada a x y s es el estado que se obtiene de aplicar a a x.

Resolución de Problemas: Formulación

Un problema puede ser definido a través de los siguientes componentes:

- Test de Meta: determina si un estado es un estado meta.
- Función de Costo del Camino: asigna un costo numérico a cada camino. Notamos el costo de una acción a aplicada al estado s que lleva al estado s' como $c(s, a, s')$.

Resolución de Problemas: Espacio de Búsqueda

Con el estado inicial, las acciones y la función del estado sucesor se define el espacio de búsqueda de un problema.

El espacio de búsqueda forma un grafo en donde los nodos son los estados y los arcos las acciones aplicables desde ese nodo.

Un camino en un espacio de estados es una secuencia de estados conectados por una secuencia de acciones.

Resolución de Problemas: Espacio de Búsqueda

Con el estado inicial, las acciones y la función del estado sucesor se define el espacio de búsqueda de un problema.

El espacio de búsqueda forma un grafo en donde los nodos son los estados y los arcos las acciones aplicables desde ese nodo.

Un camino en un espacio de estados es una secuencia de estados conectados por una secuencia de acciones.

Resolución de Problemas: Espacio de Búsqueda

Con el estado inicial, las acciones y la función del estado sucesor se define el espacio de búsqueda de un problema.

El espacio de búsqueda forma un grafo en donde los nodos son los estados y los arcos las acciones aplicables desde ese nodo.

Un camino en un espacio de estados es una secuencia de estados conectados por una secuencia de acciones.

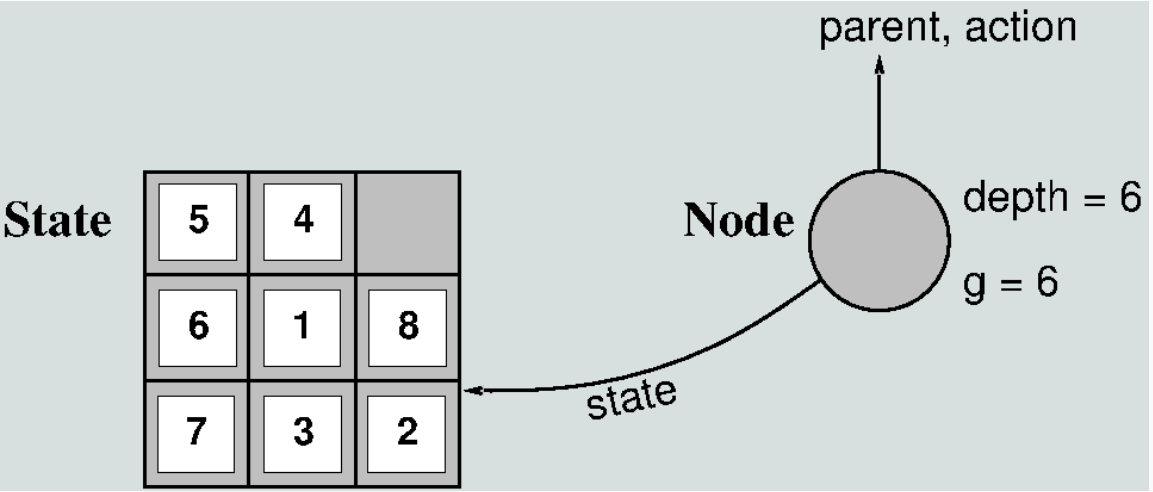
Implementación: estados vs. nodos

Estado

Es una *representación* de una configuración física

Nodo

Es una estructura de dato que constituye una parte del árbol de búsqueda, incluye padre, hijo, profundidad, costo del camino, $g(x)$



¡¡Los estados no tienen padre, hijos, profundidad ni costo!!

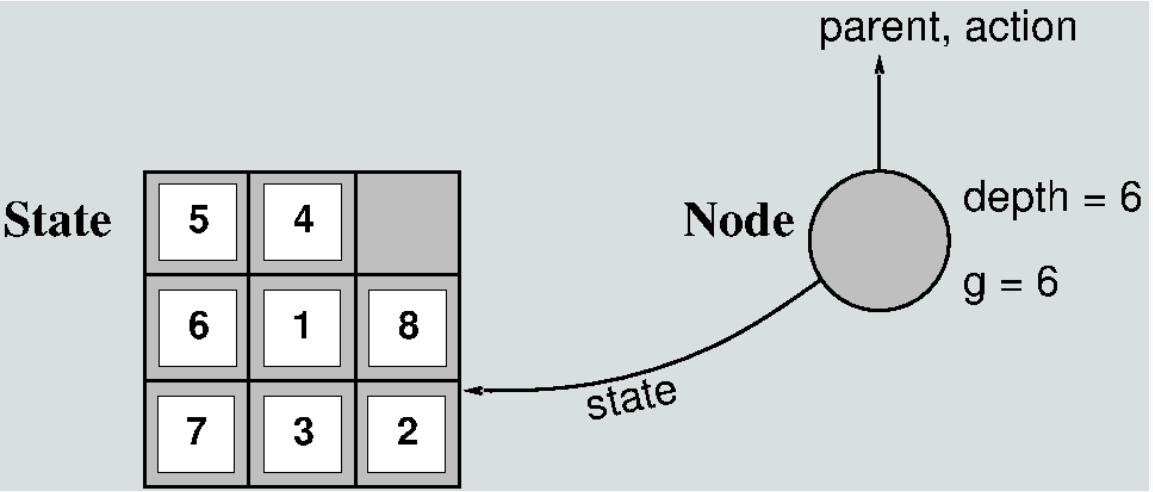
Implementación: estados vs. nodos

Estado

Es una *representación* de una configuración física

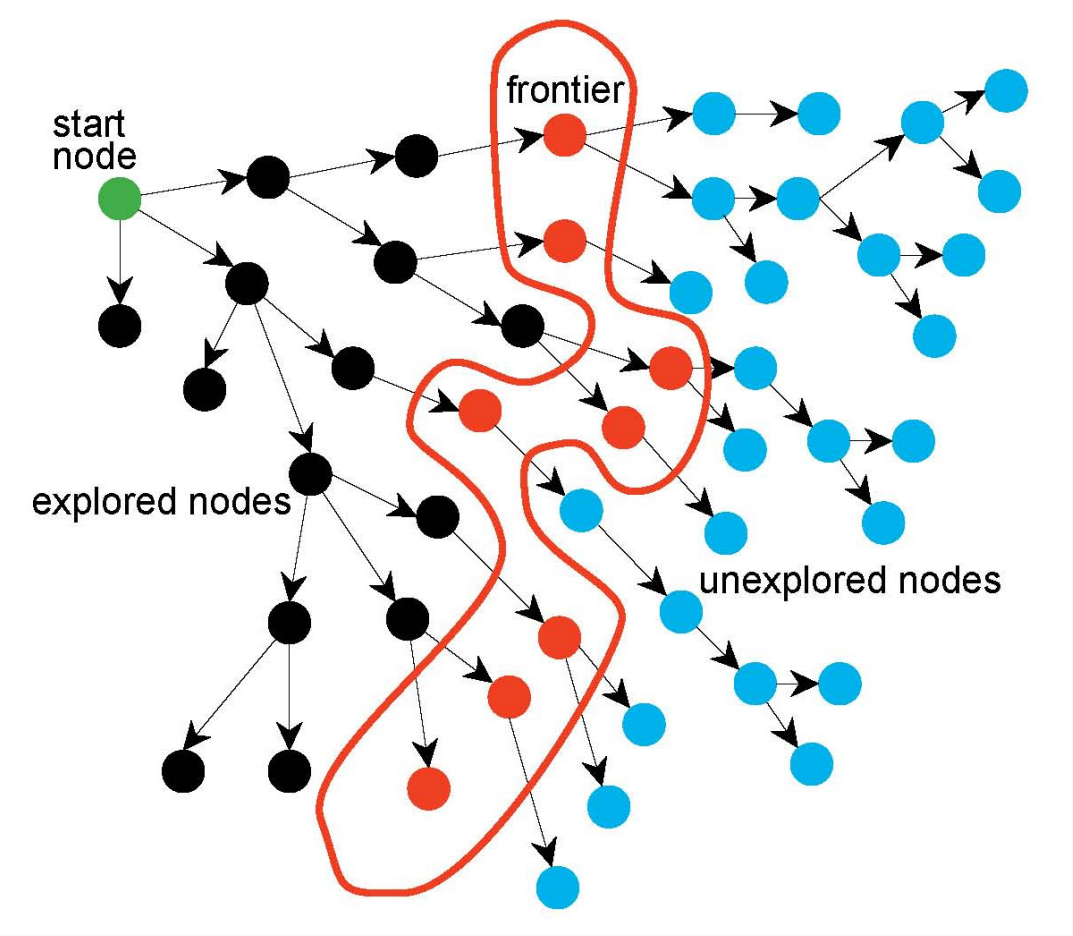
Nodo

Es una estructura de dato que constituye una parte del árbol de búsqueda, incluye padre, hijo, profundidad, costo del camino, $g(x)$



¡¡Los estados no tienen padre, hijos, profundidad ni costo!!

Árbol de Búsqueda



Una estrategia es definida eligiendo el orden de la expansión de nodos.

Lo que IA se llevó

- Búsquedas ciegas
  - Depth First
  - Breadth First
  - Iterative Deepening
  - Uniform Cost Search
  - Bidirectional Search
- Búsquedas Informadas
  - Greedy Best First Search
  - A*(heurística admisible + costo hasta el nodo)

Heurísticas

Judea Pearl's book *Heuristics: Intelligent Search Strategies for Computer Problem Solving*

Heuristics are criteria, methods, or principles for deciding which among several alternative courses of action promises to be the most effective in order to achieve some goal. They represent compromises between two requirements: the need to make criteria simple and, at the same time, the desire to see them discriminate correctly between good and bad choices.
A heuristic may be a rule of thumb that is used to guide one's actions.

Heurísticas

Espero no esté pasando :)

Lo que IA NO se llevó

Algoritmos de Búsqueda Local

En muchos problemas de optimización, el camino es irrelevante.

El estado meta en sí mismo es la solución.

Aún así, muchas veces la calidad de la solución no es la óptima.

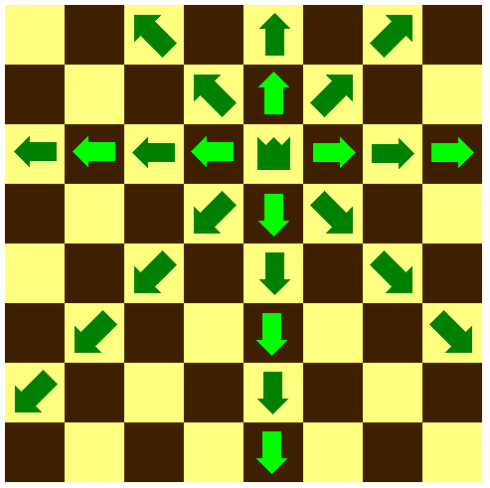
Estas estrategias:

- tratan de mejorar la complejidad en tiempo y espacio;
- sacrifican completitud y optimalidad.

Algoritmos de Búsqueda Local

8 Reinas

Dado un tablero de ajedrez (8x8) debemos ubicar 8 reinas de modo que ninguna amenace a la otra.
En general, el problema de las n-reinas, consiste en ubicar n reinas en un tablero de $n \times n$ de modo que ninguna amenace a otra.

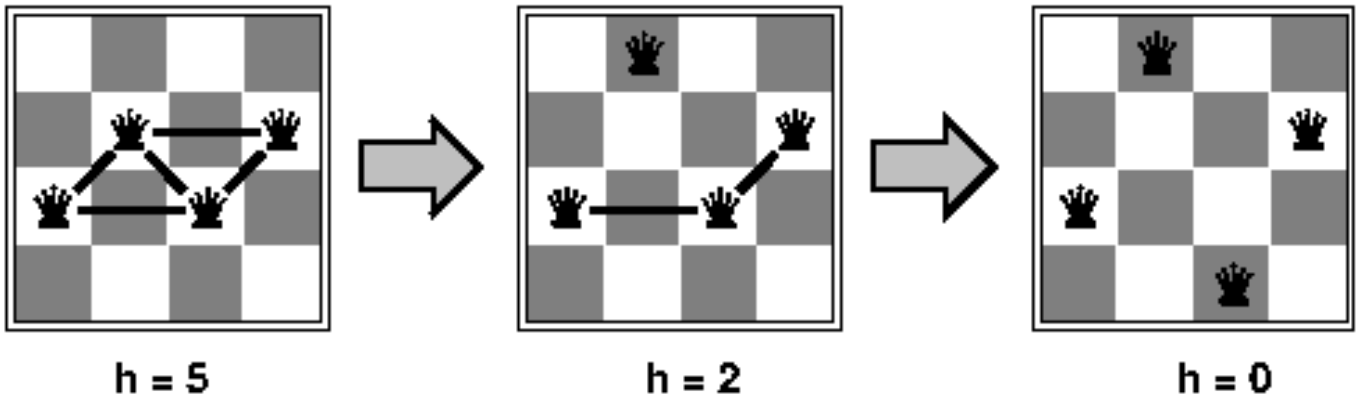


¿Cómo mueve una reina?

Ejemplo: 4-reinas

Heurística

Número de pares de reinas que se atacan.



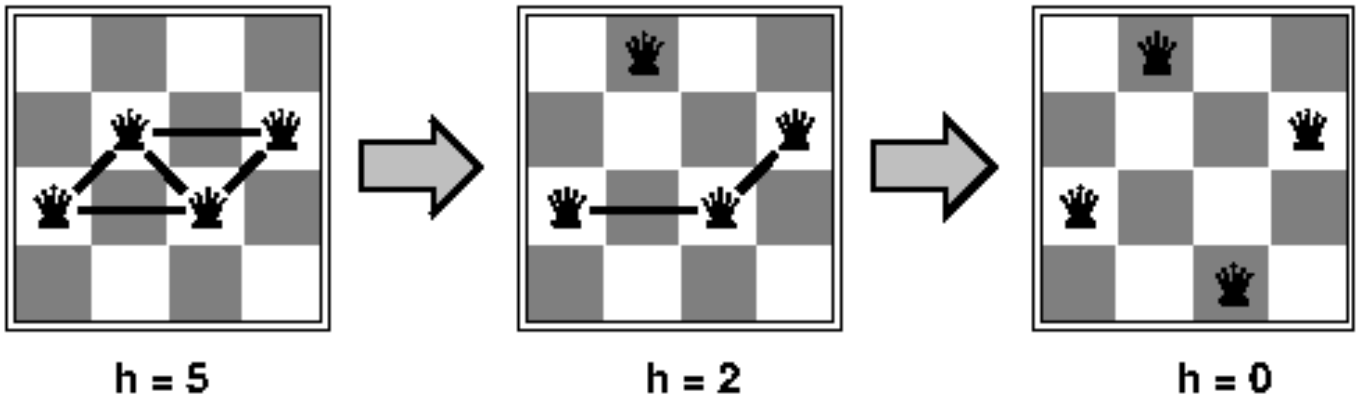
Buscamos mover una reina para minimizar el número de conflictos o amenaza.

En este problema no importa en qué orden se acomodaron las reinas, sino la configuración final.

Ejemplo: 4-reinas

Heurística

Número de pares de reinas que se atacan.



Buscamos mover una reina para minimizar el número de conflictos o amenaza.
En este problema no importa en qué orden se acomodaron las reinas, sino la configuración final.

Algoritmos de Búsqueda Local

- En muchos problemas de optimización, el camino es irrelevante; el estado meta en sí mismo es la solución.
- El espacio de estados es un conjunto de configuraciones completas. Se desea encontrar la configuración óptima o encontrar la configuración que satisface restricciones.
- En tales casos se puede usar algoritmos incrementales; conservando un único estado actual y tratar de mejorarlo.

Algoritmos de Búsqueda Local

- Los algoritmos de búsqueda local operan utilizando un único nodo actual y generalmente se mueven solamente a sus vecinos.
- Generalmente, los caminos seguidos en la búsqueda no son guardados.
- Son útiles para resolver problemas de optimización, en los que el objetivo es encontrar el mejor estado de acuerdo a una *función objetivo.*

Algoritmos de Búsqueda Local

Aunque no son sistemáticos, tienen dos ventajas claves:

- Usan muy poca memoria, usualmente una cantidad constante;
- Pueden generalmente encontrar soluciones razonables en espacios de estados grandes o infinitos para los que los algoritmos sistemáticos no son adecuados.

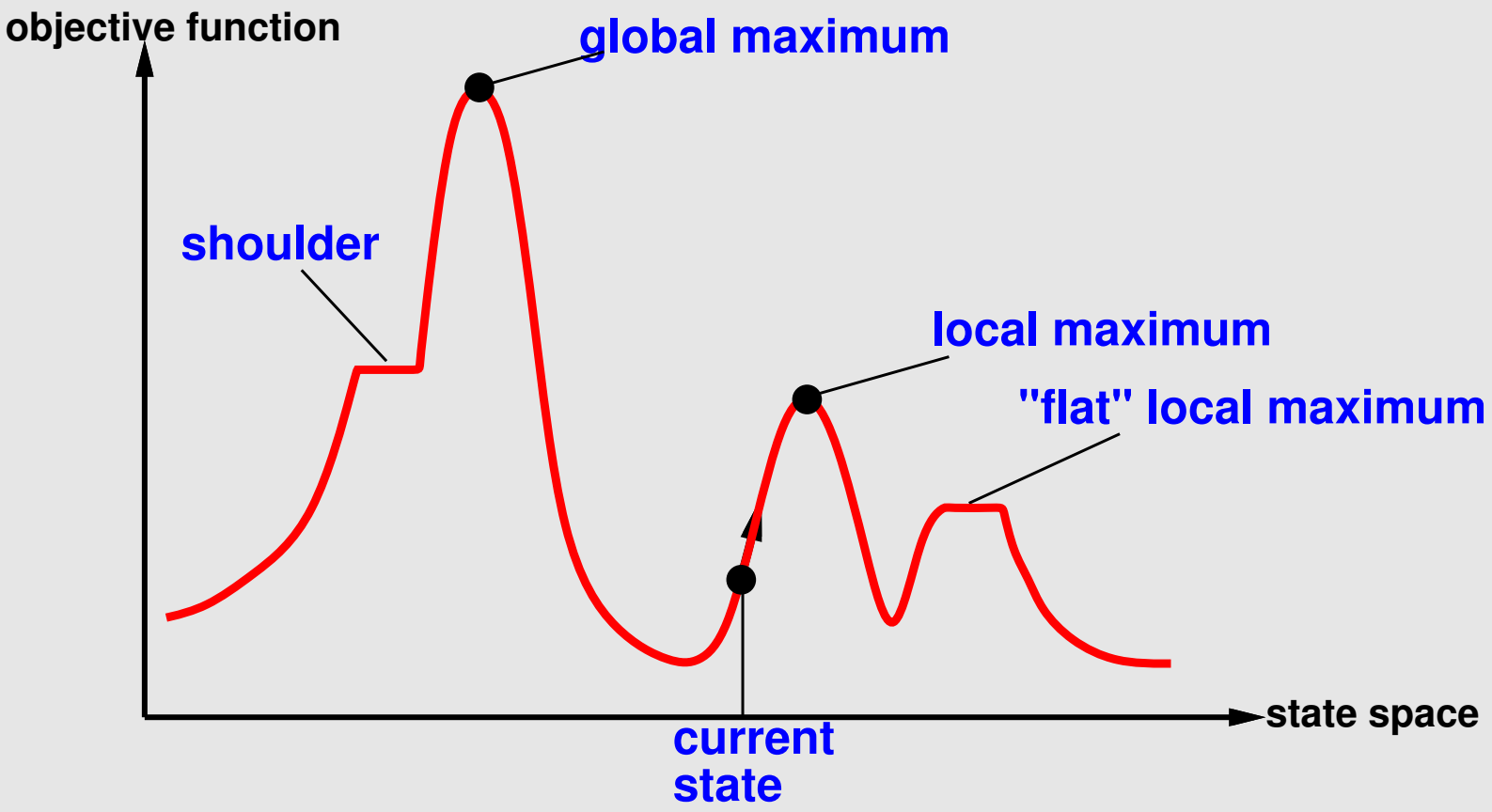
Algoritmos de Búsqueda Local

Analogía y diferencias con los métodos de búsqueda vistos en IA:

- Estado Inicial: configuración de estado completo.
- Operators (función objetivo): cambios aplicados al nodo actual para mejorar heurísticamente la calidad.
- Función de Evaluación: a cambio de un test de estado final,se usa una función de evaluación. El problema puede no tener una meta exacta.
- Búsqueda: consiste en mejorar la calidad del estado actual hasta que un número de iteraciones del algoritmo se han alcanzado o alguna condición de finalización se ha cumplido. Generalmente no se recuerda los estados visitados

Una Foto del Paisaje del Espacio de Estado

Sonrían CLICK!!



Hill-climbing

Idea: continuar mejorando el estado actual y parar cuando no podemos mejorar más.

Es como subir al Everest con una fuerte neblina, sin mapa, mientras se sufre de amnesia :)

Hill-climbing

```
function HILL-CLIMBING( problem) returns un estado que es un máximo
local
entradas: problem, un problema
variables locales: current, un nodo
                   neighbor, un nodo


current ← MAKE-NODE(INITIAL-STATE[problem])
loop do
    neighbor ← el sucesor de current con el valor más alto
    if VALUE[neighbor] ≤ VALUE[current] then return STATE[current]
    current ← neighbor
end
```

Hill-climbing

Uphill - En subida

- Up: Siempre se mueve en la dirección que incrementa el valor de los vecinos del nodo actual.
- Lo más alto que pude: Termina cuando alcanza un pico en el que ningún vecino tiene valor mayor a él.
- No recuerdo por donde vine: No mantiene el árbol de búsqueda (diferencia con Depth-First Heurístico), por lo que sólo requiere guardar la estructura de datos del nodo actual y su valor heurístico.
- No veo más que las alternativas inmediatas:No mira más allá de los vecinos inmediatos del estado actual.

Según por donde comencemos...

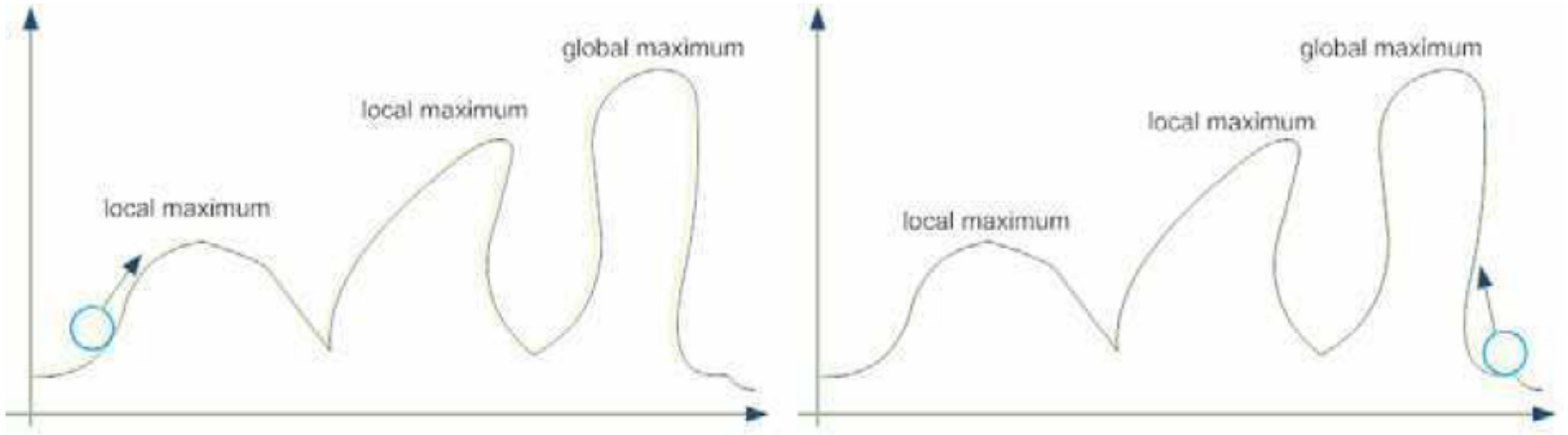


Fig. 4.2 Example of different starting states for the hill-climbing search leading to a local maximum (left) and global maximum respectively (right).

Hill-climbing: *n*-Reinas

- Utilizamos una formulación del estado completo donde cada estado tiene las *n*-reinas sobre el tablero.
- Los sucesores de un estado son todos los posibles estados generados por el movimiento de una única reina a otro cuadrado.
- La función heurística *h* es el número de pares de reinas que se atacan.
- El mínimo global es cero.
- Este algoritmo generalmente resuelve problemas de *n*-reinas casi instantáneamente, para *n* muy grandes, e.g., $n = 1$ millón

4-reinas

Ejemplo

Los estados son una configuración completa del problema.

Heurística: Número de pares de reinas que se atacan.

Ejecute el algoritmo hill- climbing desde la siguiente configuración inicial: las 4 reinas ubicadas todas en la primer fila.

4-reinas

Ejemplo

Los estados son una configuración completa del problema.

Heurística: Número de pares de reinas que se atacan.

Ejecute el algoritmo hill- climbing desde la siguiente configuración inicial: las 4 reinas ubicadas todas en la primer fila.

Hill-climbing: Problemas

Máximo Local

Es un pico que es mayor que todos sus vecinos pero que es menor que el máximo global.Si Hill-climbing alcanza la base de un máximo local, subirá hasta la cima de ese máximo local.

Hill-climbing: Problema del Máximo Local

Hill-climbing: Problemas

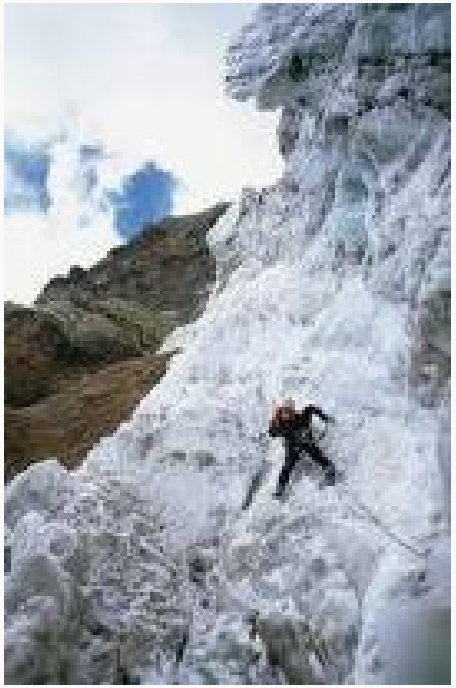
Plateaux - Planicie

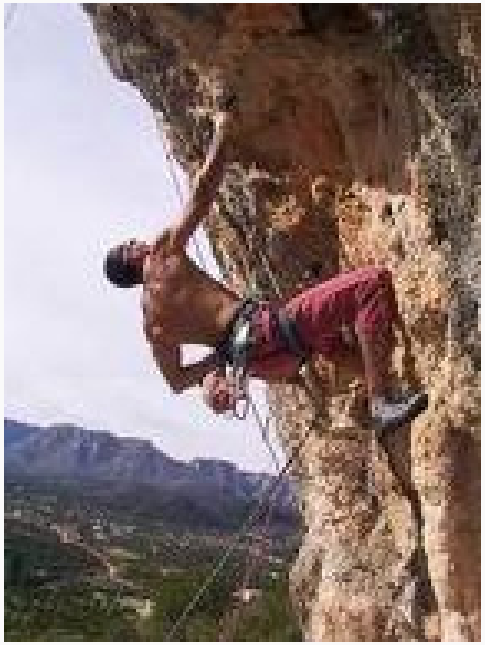
Es un área plana. La búsqueda Hill-climbing se pierde en una llanura.

Hill-climbing: Problemas

Risco

En esta situación hay que perder un poco para ganar.





Hill-climbing: resumiendo

Ventajas

1. Fácil de implementar
2. Requiere poca memoria(ya que no requiere backtracking)
3. Puede ser fácilmente usado para obtener soluciones aproximadas cuando la exacta es difícil o imposible de encontrar (ej. Problema del viajante o NP-completos).

Hill-climbing: resumiendo

Desventajas

1. La función de evaluación puede ser dificultosa de diseñar
2. Si el número de movimientos es muy grande, el algoritmo puede ser ineficiente
3. Por el contrario, si el número de movimientos es menor, el algoritmo se atasca fácilmente
4. Muchas veces es más barato evaluar un cambio incremental de un objeto ya evaluado que evaluar desde cero

Hill-climbing: Variaciones

- Hill-Climbing Estocástico: elige al azar entre los movimiento de subida. La probabilidad de selección puede variar con la inclinación del movimiento en subida.
- First-choice hill-climbing: implementa hill climbing con azar generando aleatoriamente los sucesores hasta que uno es generado que es mejor que el estado actual. Es una buena estrategia cuando un estado tiene muchos sucesores.
- Random-restart Hill-climbing: *Si al principio no tiene éxito, inténtelo de nuevo*. Recomienzos aleatorios para evitar máximos locales.

Hill-climbing y Random Walk

- Hill-Climbing nunca hace un movimiento de descenso en la colina.
  Además se atasca cuando el mejor estado sucesor del actual no es mejor que el actual.
- Random Walk: moverse a un sucesor elegido en forma uniforme al azar del conjunto de sucesores, aún cuando no sea el mejor.

¿Y si los combinamos?

Hill-climbing y Random Walk

- Hill-Climbing nunca hace un movimiento de descenso en la colina.
  Además se atasca cuando el mejor estado sucesor del actual no es mejor que el actual.
- Random Walk: moverse a un sucesor elegido en forma uniforme al azar del conjunto de sucesores, aún cuando no sea el mejor.

¿Y si los combinamos?

Hill-climbing y Random Walk

- Hill-Climbing nunca hace un movimiento de descenso en la colina.
  Además se atasca cuando el mejor estado sucesor del actual no es mejor que el actual.
- Random Walk: moverse a un sucesor elegido en forma uniforme al azar del conjunto de sucesores, aún cuando no sea el mejor.

¿Y si los combinamos?

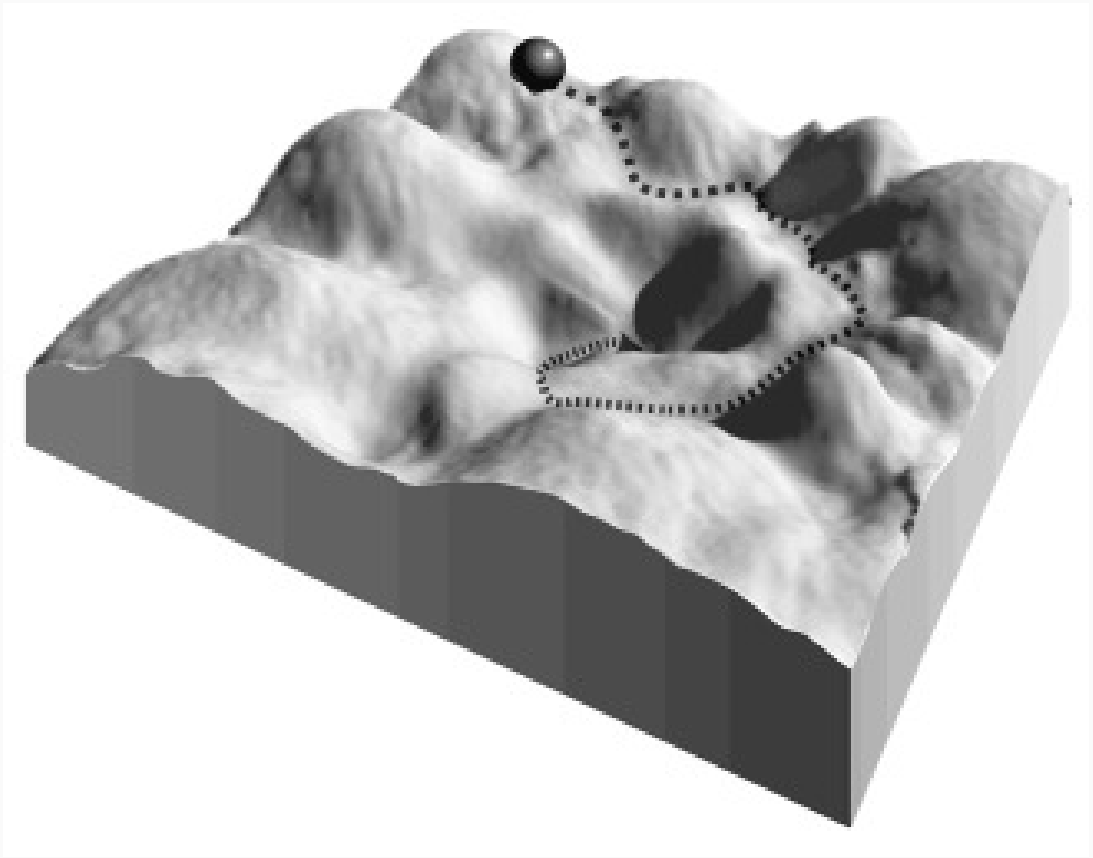
Simulated Annealing

El templado en metalurgia, es el proceso usado para endurecer los metales calentándolos a una alta temperatura y gradualmente enfriándolos. El mismo proceso se usa para el chocolate en pastelería :)

Simulated Annealing

Idea

Comienza sacudiendo fuertemente (alta temperatura) y luego gradualmente reduce la intensidad del batido (es decir baja la temperatura)



Simulated Annealing

Idea

Permite moverse en la dirección equivocada con una probabilidad base.

La probabilidad de un movimiento hacia atrás decrece a medida que la búsqueda continua.

Al principio en la búsqueda cuando se está lejos de la meta, la heurística podría no ser tan buena y debería mejorar a medida que se está más cerca de la meta.

Simulated Annealing

Idea

Permite moverse en la dirección equivocada con una probabilidad base.

La probabilidad de un movimiento hacia atrás decrece a medida que la búsqueda continua.

Al principio en la búsqueda cuando se está lejos de la meta, la heurística podría no ser tan buena y debería mejorar a medida que se está más cerca de la meta.

Simulated Annealing

Idea

Permite moverse en la dirección equivocada con una probabilidad base.

La probabilidad de un movimiento hacia atrás decrece a medida que la búsqueda continua.

Al principio en la búsqueda cuando se está lejos de la meta, la heurística podría no ser tan buena y debería mejorar a medida que se está más cerca de la meta.

Simulated Annealing: Fundamento Probabilístico

El templado en metalurgia es el proceso utilizado para templar o endurecer metales y vidrios calentándolos a altas temperaturas y luego enfriándolos en forma gradual, permitiendo al material alcanzar un estado cristalino de menor energía.

La energía sigue la distribución de Boltzmann:

$$F(state) = e^{E/kT}$$

donde E es la energía del estado, T la temperatura y k la constante de Boltzmann

Simulated Annealing: Fundamento Probabilístico

La proporción de la distribución de Boltzmann calculada para dos estados, conocida como el Factor de Boltzmann se calcula como:

$$\frac{F(state2)}{F(state1)} = \frac{e^{E_2/kT}}{e^{E_1/kT}} =$$

$$e^{(E_2-E_1)/kT} = e^{(\Delta E)/kT}$$

donde E_i es la energía del estado i , T la temperatura y k la constante de Boltzmann.

Simulated Annealing: Fundamento Probabilístico

La proporción de la distribución de Boltzmann calculada para dos estados, conocida como el Factor de Boltzmann se calcula como:

$$\frac{F(state2)}{F(state1)} = \frac{e^{E_2/kT}}{e^{E_1/kT}} =$$

$$e^{(E_2-E_1)/kT} = e^{(\Delta E)/kT}$$

donde E_i es la energía del estado i , T la temperatura y k la constante de Boltzmann.

Simulated Annealing: Fundamento Probabilístico

La proporción de la distribución de Boltzmann calculada para dos estados, conocida como el Factor de Boltzmann se calcula como:

$$\frac{F(state2)}{F(state1)} = \frac{e^{E_2/kT}}{e^{E_1/kT}} =$$

$$e^{(E_2-E_1)/kT} = e^{(\Delta E)/kT}$$

donde E_i es la energía del estado i , T la temperatura y k la constante de Boltzmann.

Simulated Annealing: Fundamento Probabilístico

La proporción de la distribución de Boltzmann calculada para dos estados, conocida como el Factor de Boltzmann se calcula como:

$$\frac{F(state2)}{F(state1)} = \frac{e^{E_2/kT}}{e^{E_1/kT}} =$$

$$e^{(E_2-E_1)/kT} = e^{(\Delta E)/kT}$$

donde E_i es la energía del estado i , T la temperatura y k la constante de Boltzmann.

Simulated Annealing: Fundamento Probabilístico

El algoritmo utiliza una búsqueda random, que no solo acepta cambios que aumentan la función f sino algunos cambios que lo decrementan.

Esto último es hecho con probabilidad:

$$p = e^{(\Delta E)/T}$$

donde ΔE es la diferencia de la función en el estado sucesor y el actual y T la temperatura.

Simulated Annealing: Fundamento Probabilístico

El algoritmo utiliza una búsqueda random, que no solo acepta cambios que aumentan la función f sino algunos cambios que lo decrementan.

Esto último es hecho con probabilidad:

$$p = e^{(\Delta E)/T}$$

donde ΔE es la diferencia de la función en el estado sucesor y el actual y T la temperatura.

Simulated Annealing: Fundamento Probabilístico

Si $\Delta E \leq 0$, entonces el estado sucesor es peor que el actual.

$$p = e^{(E_2 - E_1)/T} = e^{(\Delta E)/T} = \frac{1}{e^{(-\Delta E)/T}}$$

Si la temperatura T tiende a ∞, luego $(-\Delta E)/T \longrightarrow 0$.

Luego $e^{(-\Delta E)/T} \longrightarrow 1$.

Así, la probabilidad de seleccionar un mal estado tiende a 1 cuando la temperatura es muy alta.

A medida que disminuye la temperatura esta probabilidad también disminuye (tiende a 0).

Simulated Annealing: Fundamento Probabilístico

Si $\Delta E \leq 0$, entonces el estado sucesor es peor que el actual.

$$p = e^{(E_2 - E_1)/T} = e^{(\Delta E)/T} = \frac{1}{e^{(-\Delta E)/T}}$$

Si la temperatura T tiende a ∞, luego $(-\Delta E)/T \longrightarrow 0$.

Luego $e^{(-\Delta E)/T} \longrightarrow 1$.

Así, la probabilidad de seleccionar un mal estado tiende a 1 cuando la temperatura es muy alta.

A medida que disminuye la temperatura esta probabilidad también disminuye (tiende a 0).

Simulated Annealing: Fundamento Probabilístico

Si $\Delta E \leq 0$, entonces el estado sucesor es peor que el actual.

$$p = e^{(E_2 - E_1)/T} = e^{(\Delta E)/T} = \frac{1}{e^{(-\Delta E)/T}}$$

Si la temperatura T tiende a ∞, luego $(-\Delta E)/T \longrightarrow 0$.

Luego $e^{(-\Delta E)/T} \longrightarrow 1$.

Así, la probabilidad de seleccionar un mal estado tiende a 1 cuando la temperatura es muy alta.

A medida que disminuye la temperatura esta probabilidad también disminuye (tiende a 0).

Simulated Annealing: Fundamento Probabilístico

Si $\Delta E \leq 0$, entonces el estado sucesor es peor que el actual.

$$p = e^{(E_2 - E_1)/T} = e^{(\Delta E)/T} = \frac{1}{e^{(-\Delta E)/T}}$$

Si la temperatura T tiende a ∞, luego $(-\Delta E)/T \longrightarrow 0$.

Luego $e^{(-\Delta E)/T} \longrightarrow 1$.

Así, la probabilidad de seleccionar un mal estado tiende a 1 cuando la temperatura es muy alta.

A medida que disminuye la temperatura esta probabilidad también disminuye (tiende a 0).

Simulated Annealing

function SIMULATED-ANNEALING(*problem, schedule*) **returns** a estado solución
entradas: *problem*, un problema
schedule, una función de tiempo a “temperatura”
variables locales: *current*, un nodo
next, un nodo
T, una “temperatura” controlando la prob. de pasos hacia abajo

current ← MAKE-NODE(INITIAL-STATE[*problem*])
for *t* ← 1 **to** ∞ **do**
T ← *schedule*[*t*]
if *T* = 0 **then return** *current*
next ← un sucesor elegido aleatoriamente de *current*
$\Delta E \leftarrow$ VALUE[*next*] – VALUE[*current*]
if $\Delta E > 0$ **then** *current* ← *next*
else *current* ← *next* only with probability $e^{\Delta E / T}$

Simulated Annealing: ¿Cómo funciona?

- En la selección del sucesor, en vez de tomar el mejor movimiento como Hill Climbing, elige un movimiento al azar.
- Si el movimiento mejora la situación, es siempre aceptado.
- De otro modo, el algoritmo acepta el movimiento con una probabilidad menor que 1.
- La probabilidad decrece exponencialmente con lo malo del movimiento (ΔE) y con el enfriamiento de la temperatura: un mal movimiento es más probable que sea aceptado al principio con alta temperatura y luego se vuelve menos probable a medida que la temperatura decrece.
- Si el schedule baja la temperatura suficientemente lento, el algoritmo encontrará el óptimo global con probabilidad tendiendo a 1.

Simulated Annealing: Detalles de Implementación

- Temperatura inicial: suficientemente alta para movernos a cualquier otro vecindario. Si no es muy alta, el comportamiento será el del Hill Climbing, pero no tanto como para vagar entre vecindarios y comportarse como random walk.
- Temperatura final: El algoritmo la establece en 0, pero esto podría hacer correr el algoritmo por mucho tiempo. No es necesario hacerla llegar a 0.

Simulated Annealing: Detalles de Implementación

- Paso de Decremento: la temperatura es decrementada con un paso. Esto establece la cantidad de iteraciones del algoritmo.
- Nro. de iteraciones por cada temperatura: Existen diferentes posibilidades:
  - Nro. constante de iteraciones por temperatura.
  - Cambio dinámico del número de iteraciones a medida que el algoritmo progresa.
  - Una iteración en cada temperatura pero decrementando la temperatura muy lento.

Referencia Bibliográfica


S. Russell y P.Norvig
Artificial Intelligence: A Modern Approach (Third Edition).
Capítulo 4, hasta sección 4.2 (sin incluir)
2009.


Crina Grosan y Ajith Abraham
Intelligent Systems: A modern approach
2011.

Desafíos

Considere las siguientes ciudades, caminos directos y costo de estos caminos:

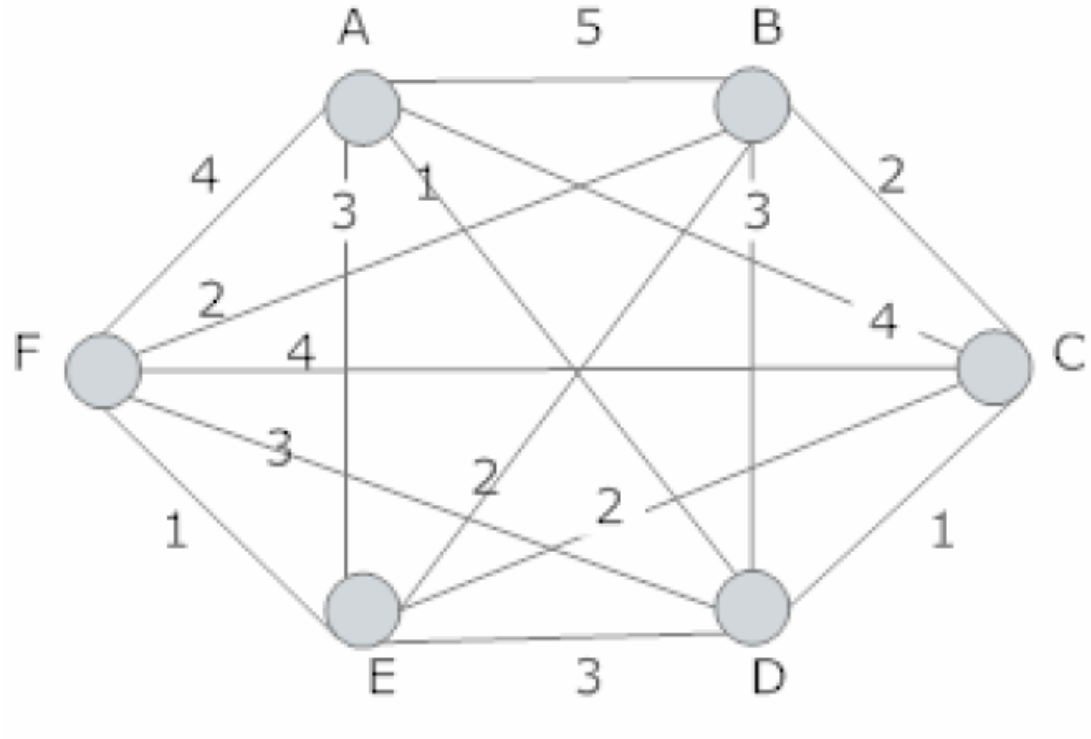


Fig. 4.6 The graph for the TSP example.

Desafíos

- Plantear el problema de optimización combinatorial NP-Completo del Viajante TSP (Travelling Salesperson Problem) como un problema de búsqueda local.
  Este problema también se conoce como Ciclo de Hamilton (volver a ETC :)

  TSP:*Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?*
- Resolver con Hill Climbing el problema TSP hasta 4 iteraciones partiendo de la configuración inicial ABCDEF.
- Determine los parámetros del Recocido Simulado y compare su resolución con Hill Climbing.

Desafíos

- Muestre un ejemplo en el que el problema de las 4-reinas entre en un risco o borde y que hill-climbing no resuelva en forma adecuada.
- Muestre un ejemplo de planicie y uno de risco para 8-puzzle.
- Explique ventajas del Recocido Simulado sobre el Hill Climbing.

¿Preguntas?