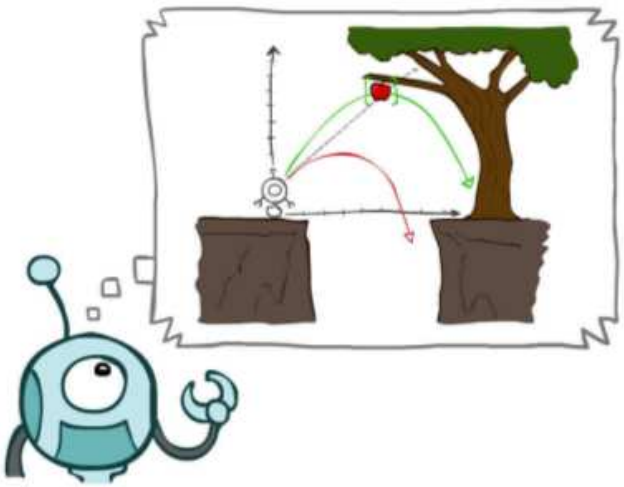
Unidad 2 - Agentes de Búsqueda y Planificación

Msc. Lic. Víctor Rodríguez Estévez

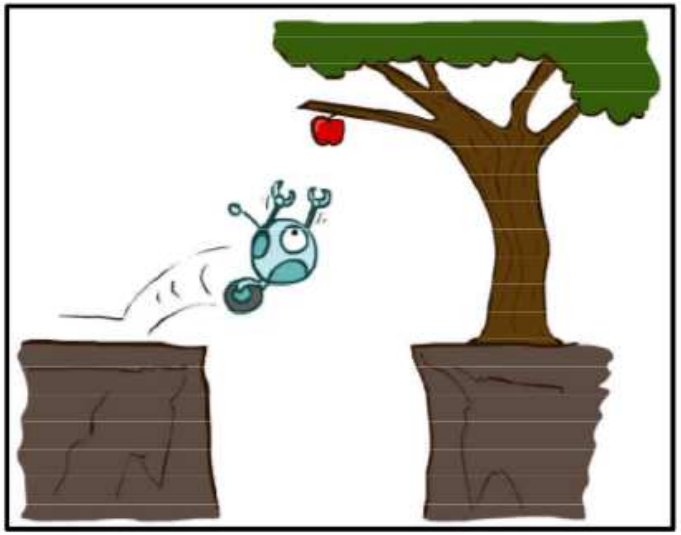
February 24, 2016

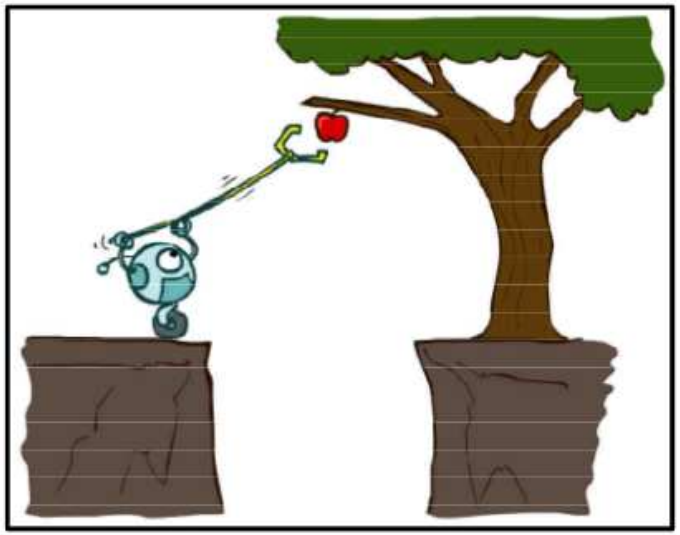
Agentes que planifican su Futuro



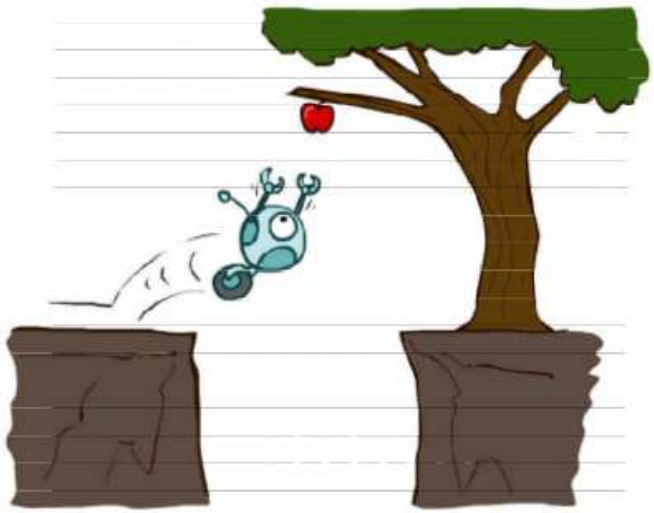
- Agentes que planifican su futuro
- Problemas de Búsqueda
- Métodos de búsqueda no informada
  - Búsqueda en profundidad
  - Búsqueda en amplitud
  - Búsqueda de costo uniforme

Agentes que planean vs Agentes Reflexivos



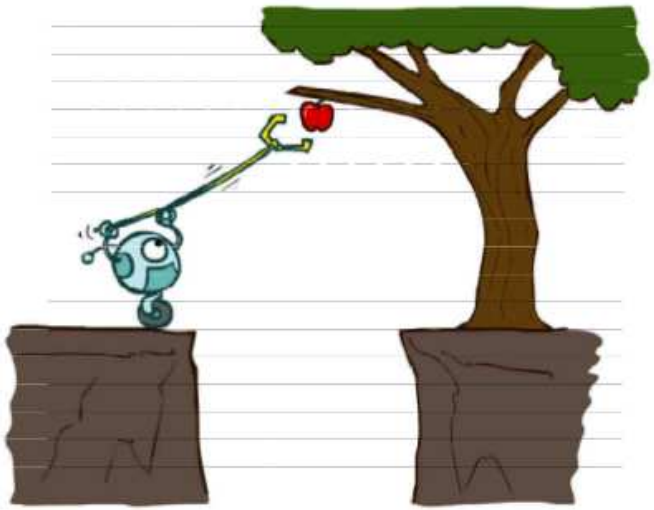


Agentes que planean



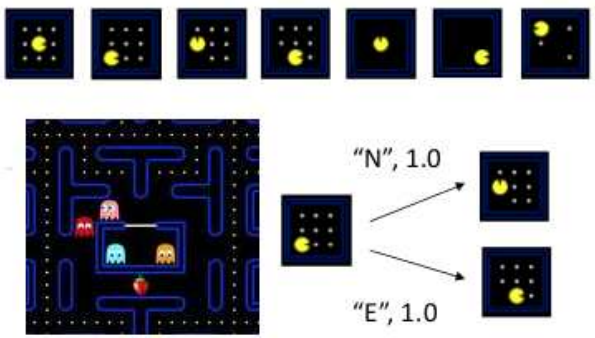
- Agentes que Reflexionan
  - Seleccionan su acción basado en una percepción actual (y quizas su memoria)
  - Puede tener memoria o un modelo del mundo de estado actual
  - No considera futuras consecuencias de sus acciones
  - Considera como el mundo ES.
- Puede un agente reflexivo ser racional?

Agentes que planean



- Agentes que Planifican
  - Se preguntan "Que pasa si...."
  - Decisiones basadas en las (hipotéticas) concecuencias de sus acciones
  - No considera futuras consecuencias de sus acciones
  - Se formulan un objetivo - un test.
  - Consideran como el SERÍA el mundo
- Planificación óptima vs Planificación Completa
- Planificación vs Replanificaciíon

Representación de Problemas



- Un Problema de Búsqueda conciste de
  - Un espacio de Estados
  - Un estado inicial
  - Una función que cambie de un estado a otro (sucesor)
  - Un estado objetivo
  - Y una función de prueba
- Una solución será una secuencia de acciones (un plan), mediante el cual, se transforme del estado inicial, al estado objetivo.

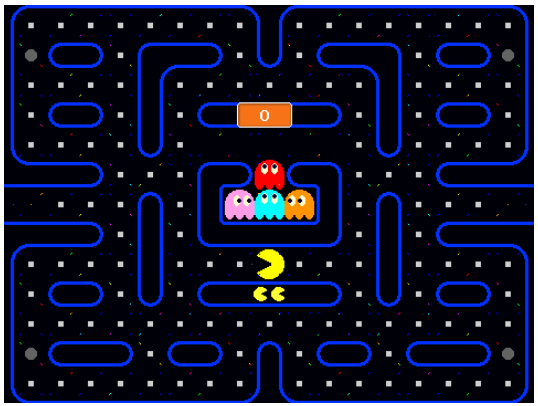
Los Problemas de Búsqueda son modelos

- Para que el Agente logre resolver el problema:
  - Debe tener un modelo del entorno
  - Una represetnación simplificada de la realidad
  - Sobre el planificará sus acciones
- Para realziar una secuencia de acciones (plan), debe tener conocimiento del entorno (modelo)

Ejemplo: Viajando por Bolivia

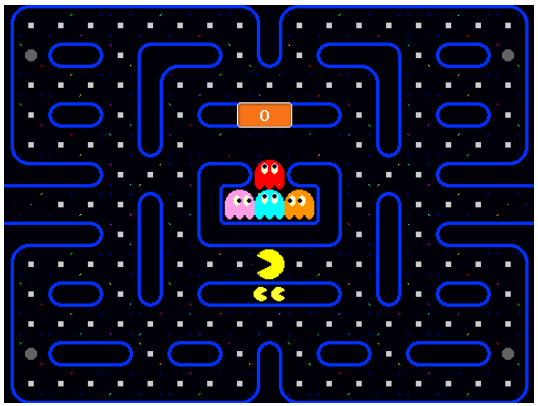
- Espacio de Estados: Ciudades de Bolivia
- Función Sucesor: Viajar de una ciudad a otra concecutiva por carretera
- Estado inicial: En Cobija
- Estado Final: En Yacuiba
- Test Objetivo: Estamos en Yacuiba?
- Solución?

Que hay en un espacio de Estados?



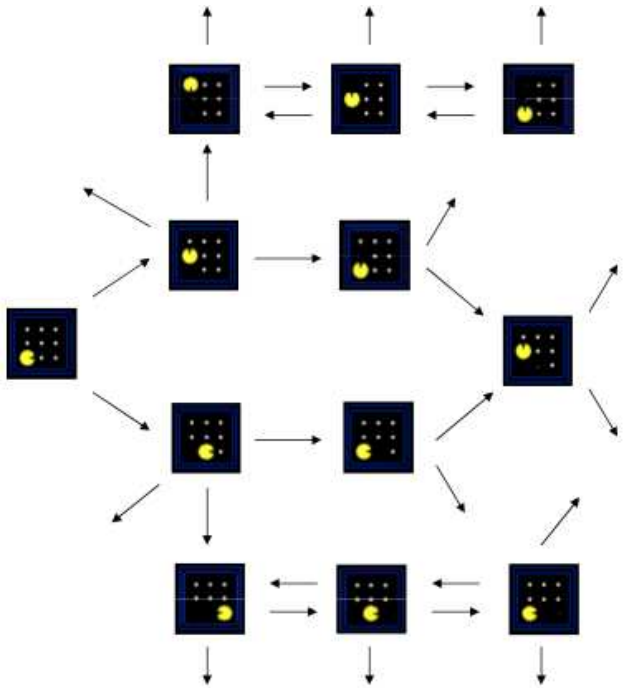
- El estado del mundo incluye hasta el último detalle del entorno
- Un estado de búsqueda conserva sólo los detalles necesarios para la planificación (abstracción)
- Problema: Camino
  - Estado: localidad (x,y)
  - Acciones: Movimientos: arriba, abajo. izquierda, derecha
  - Sucesor: actualziar localidad siguiente
  - Test Objetivo: es (x,y) = objetivo?

Que hay en un espacio de Estados?



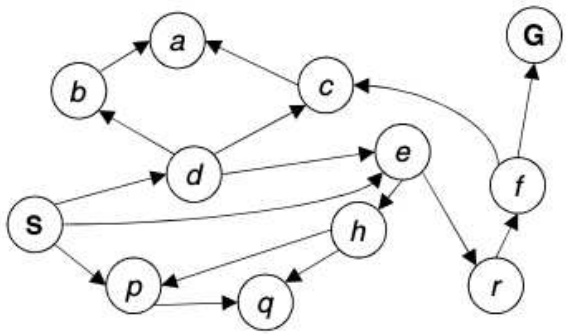
- Problema: Comerce los Puntos
  - Estado: (x,y), puntos (true-false)
  - Acciones: Movimientos
  - Sucesor: actualziar localidad siguiente, comerce punto (de true a false)
  - Test Objetivo: todos los puntos tienen valor false?

Espacio de Estados como Grafo



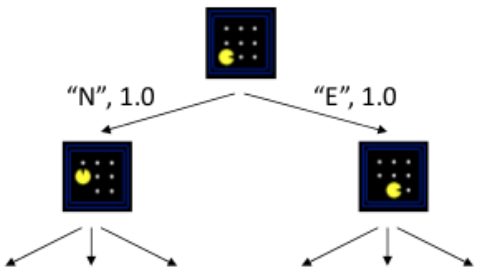
- Grafo de Estados: Una representación matemática de un problema de búsqueda
  - Los nodos son los posibles estados del mundo (abstracciones)
  - Los arcos representan los sucesores (resultado de las acciones)
  - El test objetivo evalua cada nodo verificando si es el estado deseado
- En un grafo de búsqueda cada estado ocurre solo una vez.

Espacio de Estados como Grafo



- Grafo de Estados: Una representación matemática de un problema de búsqueda
  - Los nodos son los posibles estados del mundo (abstracciones)
  - Los arcos representan los sucesores (resultado de las acciones)
  - El test objetivo evalua cada nodo verificando si es el estado deseado
- Rara vez podemos construir un grafo completo (memoria), pero es una idea útil

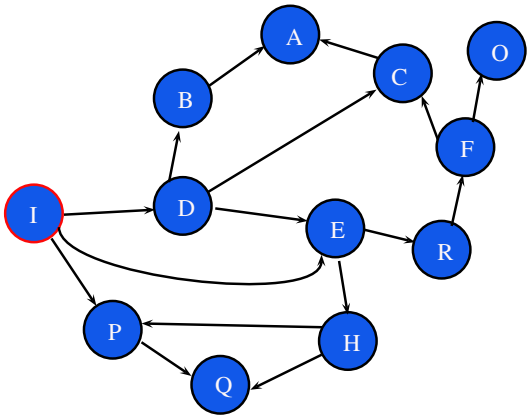
Árbol de Búsqueda



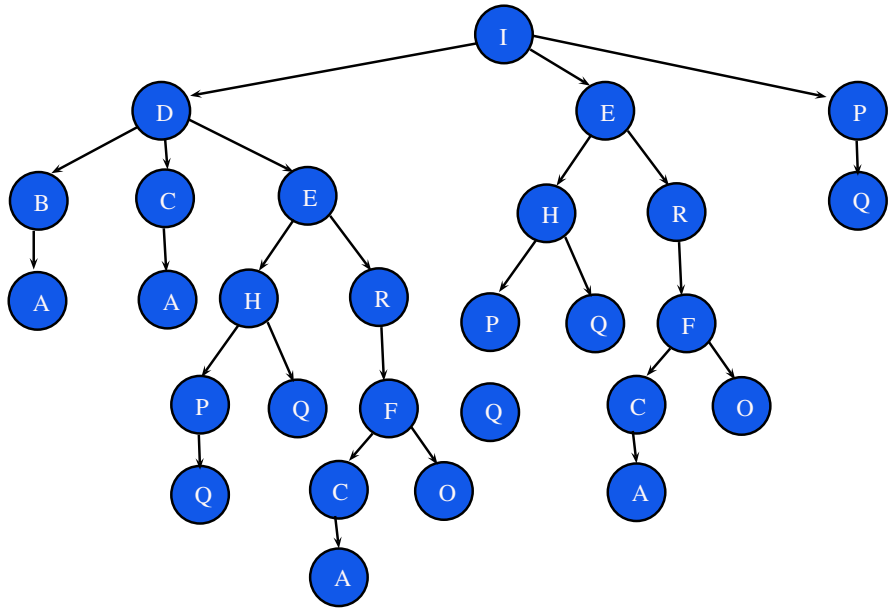
- El estado inicial es el nodo raíz
- Los nodos hijos corresponden a los sucesores.
- Los nodos son estados, pero corresponden a los planes para alcanzar dichos estados
- Para la mayoria de los problemas, nunca podremos construir todo el árbol.

Árbol de Búsqueda vs Arbol de Búsqueda

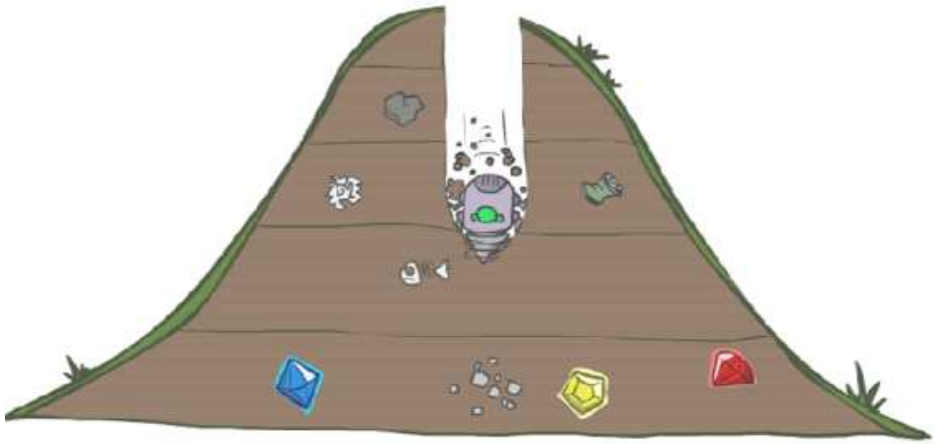
Grafo de Búsqueda



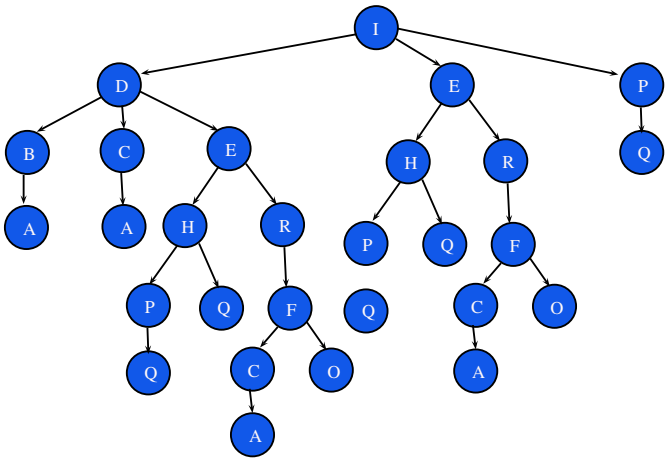
Árbol de Búsqueda



Búsqueda Primero el mas Profundo

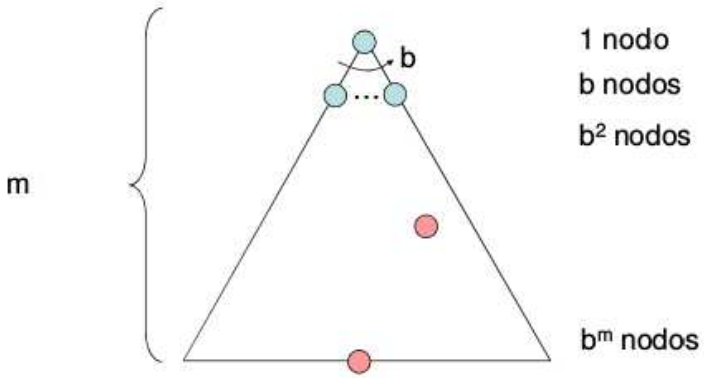


Algoritmo Primero en Profundidad



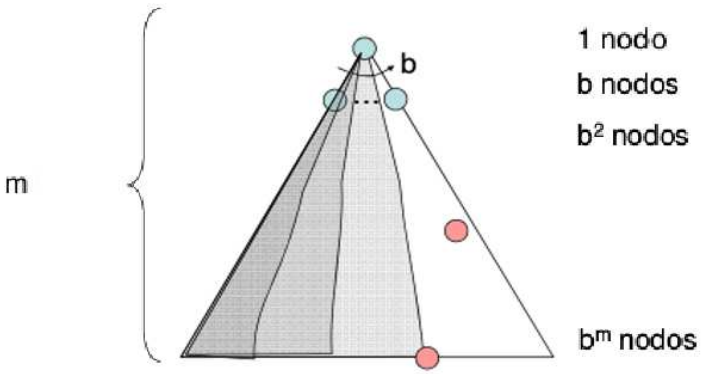
1. Estado inicial, pasa a la frontera.
2. Estado actual $\rightarrow$ sacar primero el ultimo elemento ingresado a frontera.
3. Añadir estado a Visitados
4. Verificar si estado actual es objetivo.
5. Si es, lo encontramos.
6. Si no aplicamos funcin sucesor al estado actual
7. Si el sucesor no se encuentra en Visitados, encolamos en la frontera.
8. Si frontera NO vaco y NO encontramos meta, volver al punto 2.
9. Devolver Ascendentes del estado Actual

Algoritmo Primero en profundidad



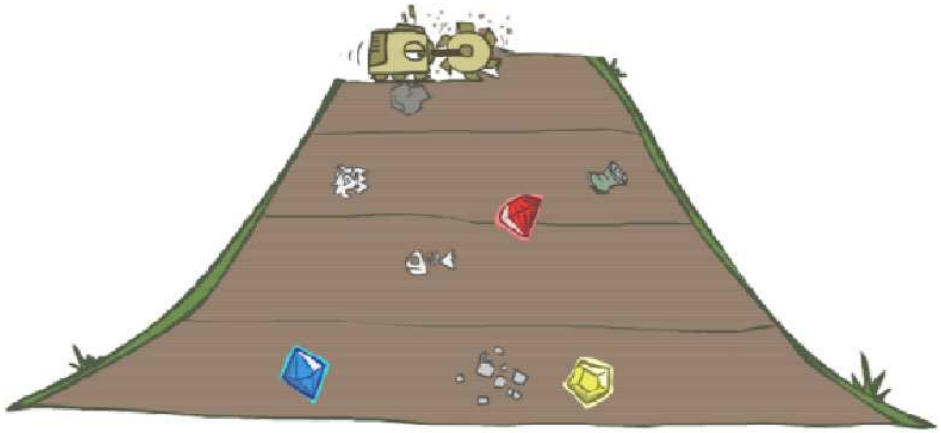
- Completo: Garantiza encontrar el objetivo si este existe?
- Óptimo: Garantiza encontrar el camino mas corto?
- Complejidad de tiempo
- Complejidad de espacio

Algoritmo Primero en profundidad

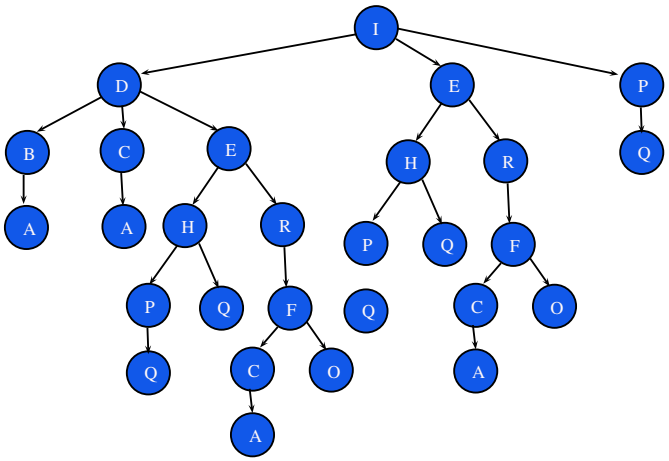


- Que nodos expande el algoritmo?
  - Podría empezar a la izquerda
  - Luego avanzar hasta lo mas profundo del árbol (m)
  - Si m es finito, tomaria un tiempo $O_{(bm)}$
- Cuanto espacio requerirá el algoritmo?
  - Solo el largo que pueda llegar $O(bm)$
- Es Completo?
  - m podría ser infinito, y no llegar a una solución aunque esta sea superficial
- Es óptimo?
  - No!.. Encuentra la solución mas a la izquierda, no necesariamente la solución mas superficial.

Búsqueda Primero en Amplitud

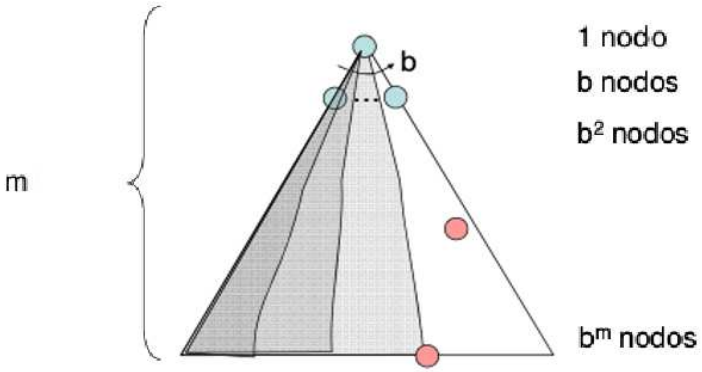


Búsqueda Primero en Amplitud



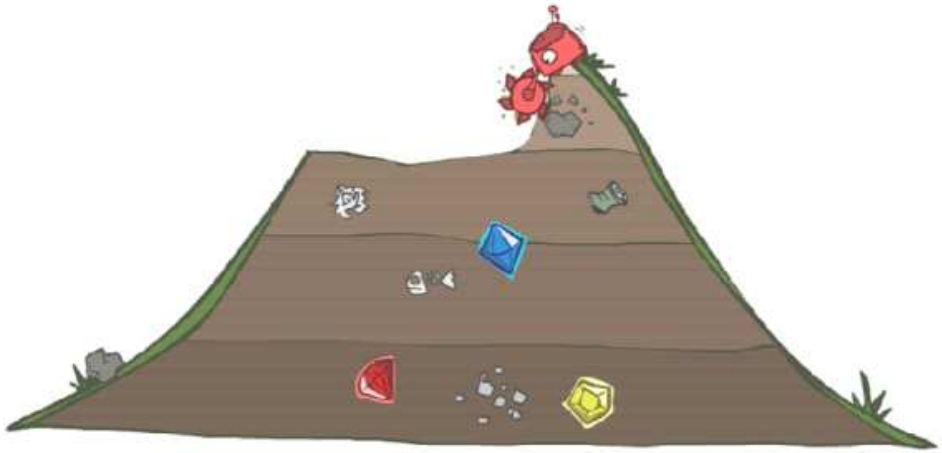
1. Estado inicial, pasa a la frontera.
2. Estado actual $\rightarrow$ sacar primero el primer elemento ingresado a frontera.
3. Añadir estado a Visitados
4. Verificar si estado actual es objetivo.
5. Si es, lo encontramos.
6. Si no aplicamos funcin sucesor al estado actual
7. Si el sucesor no se encuentra en Visitados, encolamos en la frontera.
8. Si frontera NO vaco y NO encontramos meta, volver al punto 2.
9. Devolver Ascendentes del estado Actual

Algoritmo Primero en Amplitud

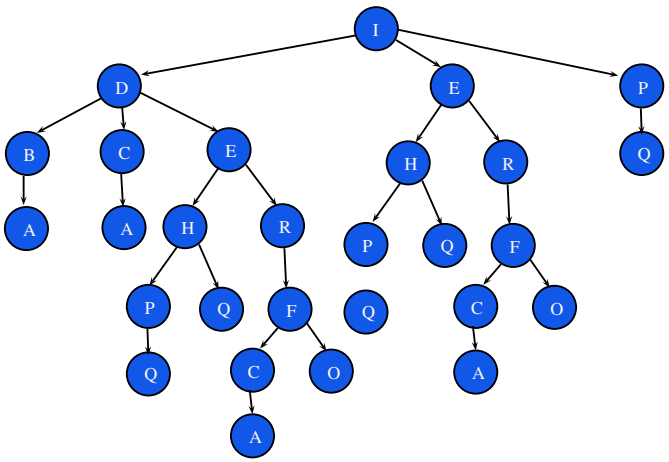


- Que nodos expande el algoritmo?
  - Empieza por un extremo
  - Luego avanzar hasta lo mas amplio del nivel s
  - La búsqueda tomaría un tiempo $O_{(b^s)}$
- Cuanto espacio requerirá el algoritmo?
  - Hasta lo mas alto del nivel que se encuentra $O(b^s)$
- Es Completo?
  - Si la solución existe, s es finito
- Es óptimo?
  - Si!.. Si el costo de cada transición de estados siempre es 1

Búsqueda Costo Uniforme

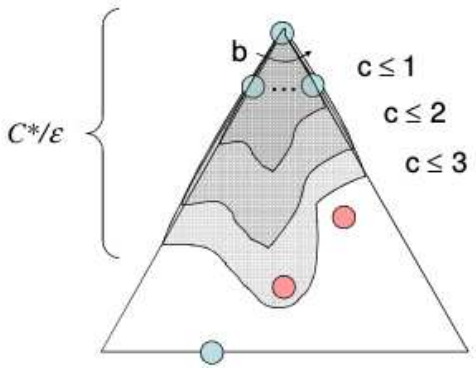


Búsqueda Costo uniforme



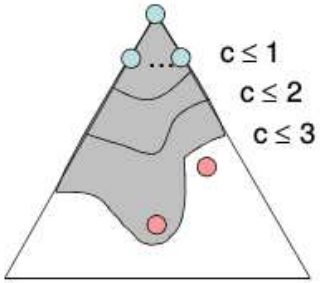
1. Estado inicial, pasa a la frontera.
2. Estado actual $\rightarrow$ sacar elemento frontera cuya sumatoria de costos sea el menor.
3. Añadir estado a Visitados
4. Verificar si estado actual es objetivo.
5. Si es, lo encontramos.
6. Si no aplicamos funcin sucesor al estado actual
7. Si el sucesor no se encuentra en Visitados, encolamos en la frontera.
8. Si frontera NO vaco y NO encontramos meta, volver al punto 2.
9. Devolver Ascendentes del estado

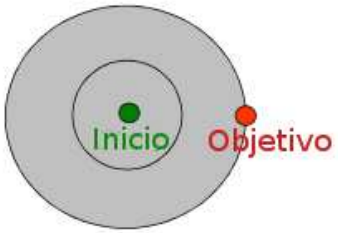
Algoritmo Costo Uniforme



- Que nodos expande el algoritmo?
  - Empieza nodo con el costo mas bajo
  - Si la solución tiene un costo de C* y el arco ϵ, entonces la profundidad efectiva es $C*/\epsilon$
  - La búsqueda tomaría un tiempo $O_{(b^{C*/\epsilon})}$
- Cuanto espacio requerirá el algoritmo?
  - Avanza hacia la sumatoria mas barata, $O(b^{C*\epsilon})$
- Es Completo?
  - Si la solución existe, s es finito
- Es óptimo?
  - Si! definitivamente!

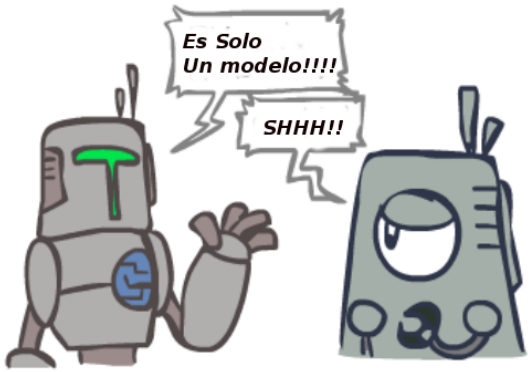
Algoritmo Costo Uniforme





- Recuerda: el algoritmo calcula la sumatoria de costos para cada acción.
- Lo bueno: Es es completo y óptimo
  - Avanza hacia la sumatoria mas baja, $O(b^{C*\epsilon})$
- Lo malo: Explora los nodos en todas las direcciones.

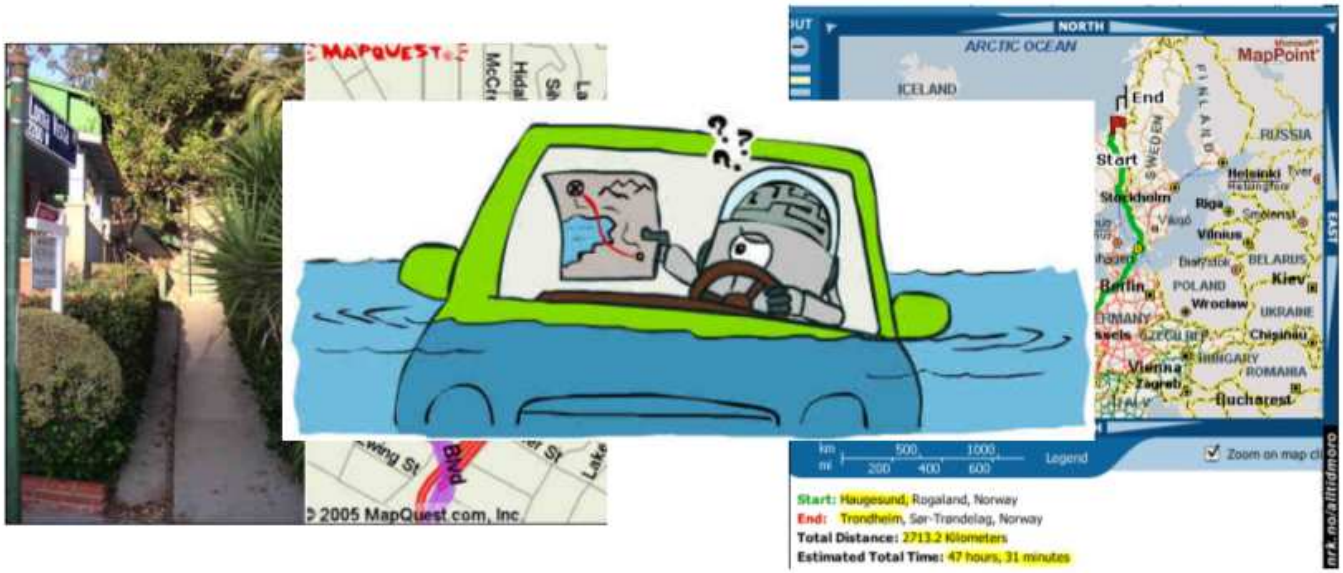
El modelo y la búsqueda



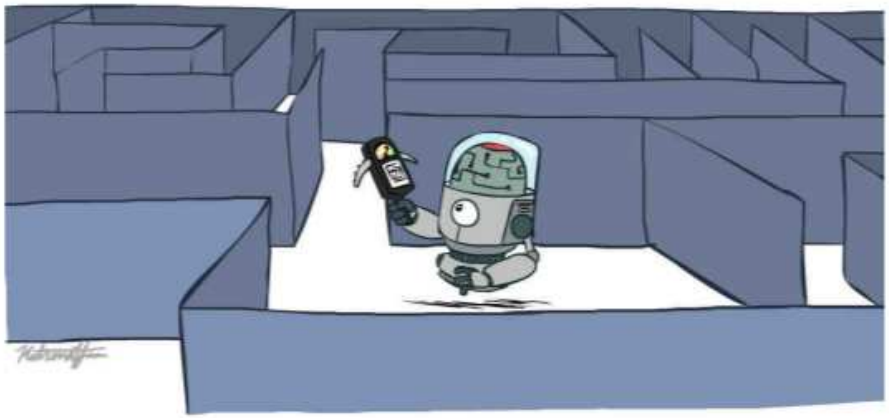
- Toda la planificación se genera en una simulación.
- La búsqueda es buena en la medida del modelo

El modelo y la búsqueda

El modelo y la búsqueda



Búsquedas con Información

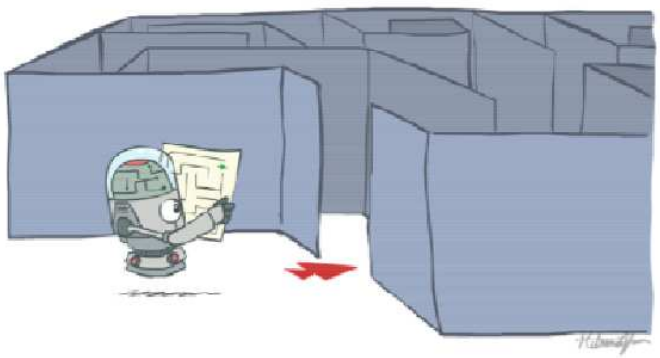


Búsqueda con Información

Búsqueda informada

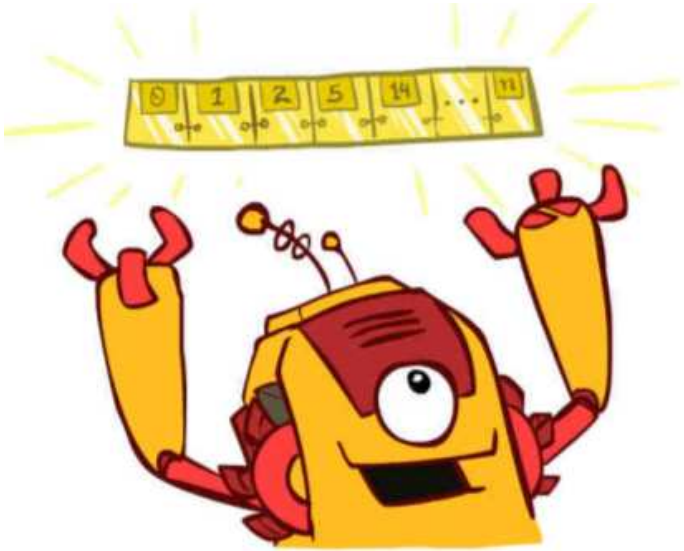
- Heurística
- Búsqueda Avara
- Algoritmo A*

El Camino hasta ahora



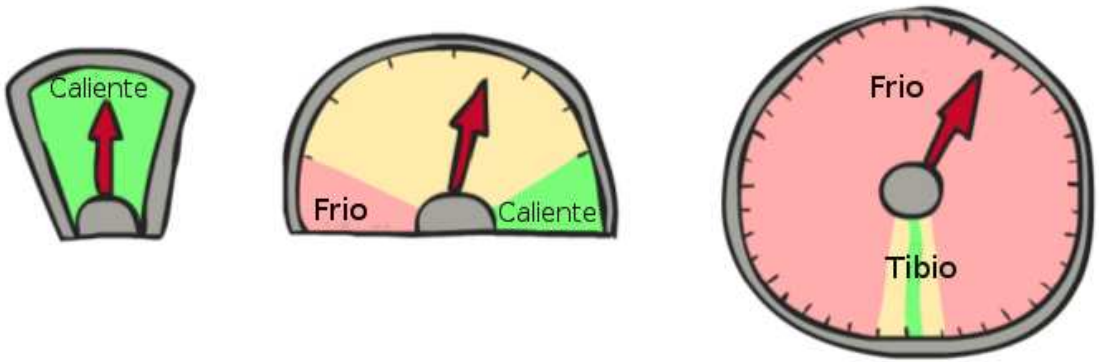
- Problemas de Búsqueda
  - Estados: Configuración del "Mundo"
  - Estado inicial y estado meta
  - Función sucesor:: Acciones y costos
  - Test Objetivo: Verificar meta
- Árbol de Búsqueda
  - Nodos: Represetnación de un plan, un estado
  - La ejecución de un plan tiene costo
- Algoritmo de Búsqueda
  - Sistemáticamente crea un árbol de búsqueda
  - Explora verificando si es el objetivo
  - Si no es genera hijos ordenandolos en una **frontera**

El Camino hasta ahora



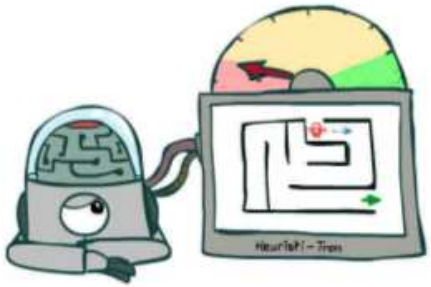
- Todos los algoritmos que vimos tienen la misma base, lo que varia es la estratetegia en el tratamiento de la frontera
  - Primero en profundidad: La frontera es una Pila
  - Primero en amplitud: La frontera es una cola
  - Costo Uniforme: La frontera es una cola con prioridad

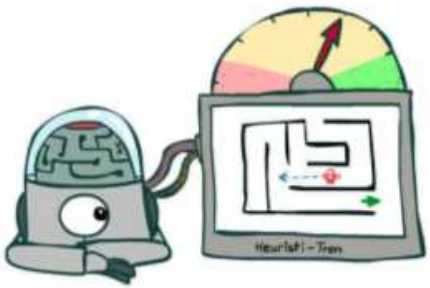
Algoritmos Informados



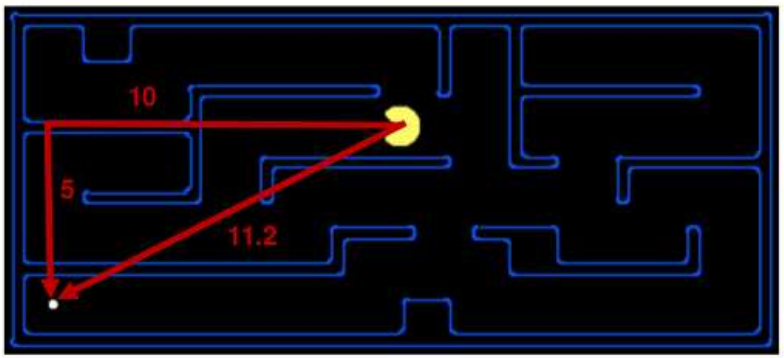
- Que tal si tuvieramos alguna información que nos guie en la búsqueda.
- Frio, Tibio, Caliente!!!!!
- Una especie de brujula que nos diga como vamos en nuetsra búsqueda.
- No nos indica exactamente, pero nos da una idea para direccionar nuestra búsqueda.
- Una Heurística!

La Heurística



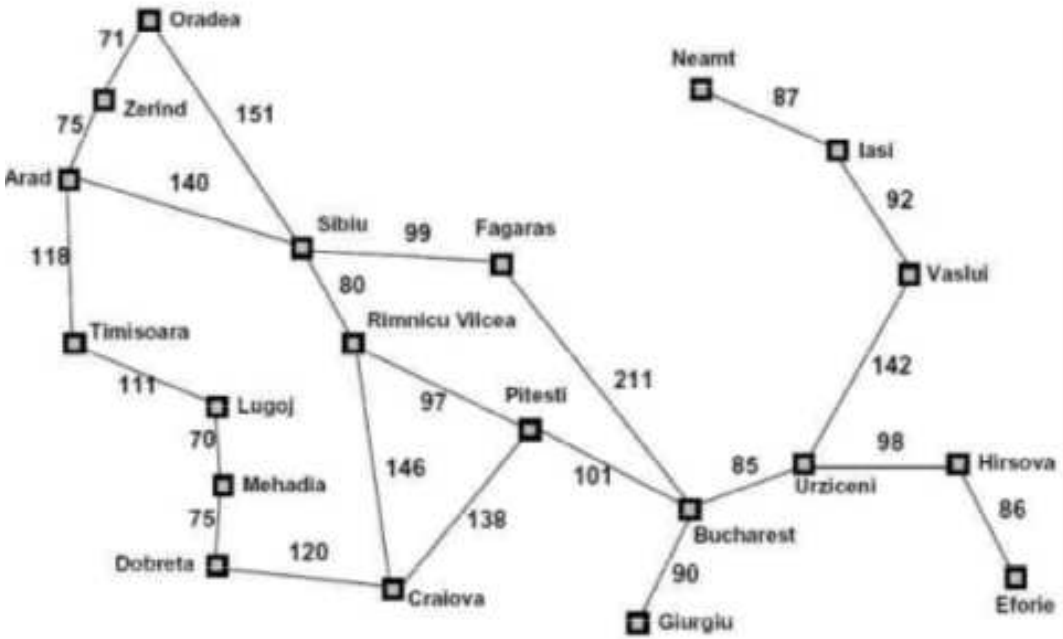


- Una Heurística es:
  - Una función que **estima** cuan cerca se encuentra un estado, del objetivo.
  - Considerado un rasgo característico de los humanos
  - Arte y la ciencia del descubrimiento o de resolver problemas mediante la creatividad.
  - Su base está en la experiencia de resolver problemas y en ver cómo otros lo hacen.
  - Ejemplos: Distancia Manhatan, Distancia Euclidiana.



Búsqueda Codiciosa

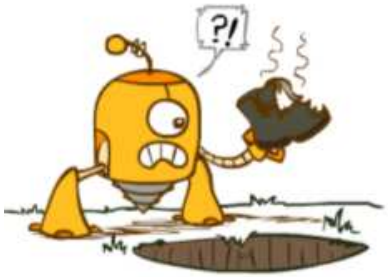
Búsqueda Codiciosa



Straight-line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

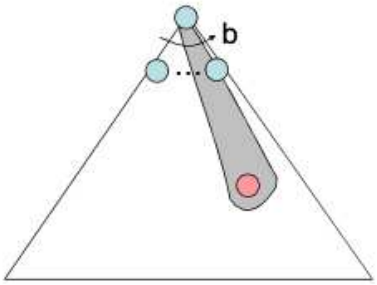
h(x)

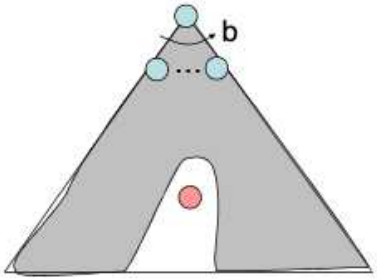
B-usqueda codiciosa



- Como usamos la Heurística :
  - Tenemos ya la guia, la heurística, porque no solo seguirla?
  - Podemos ordenar la frontera, según la heurística
  - Este algoritmo codicioso siempre expande la heurística que aparentemente este mas cerca a la meta.
  - Que puede salir mal?

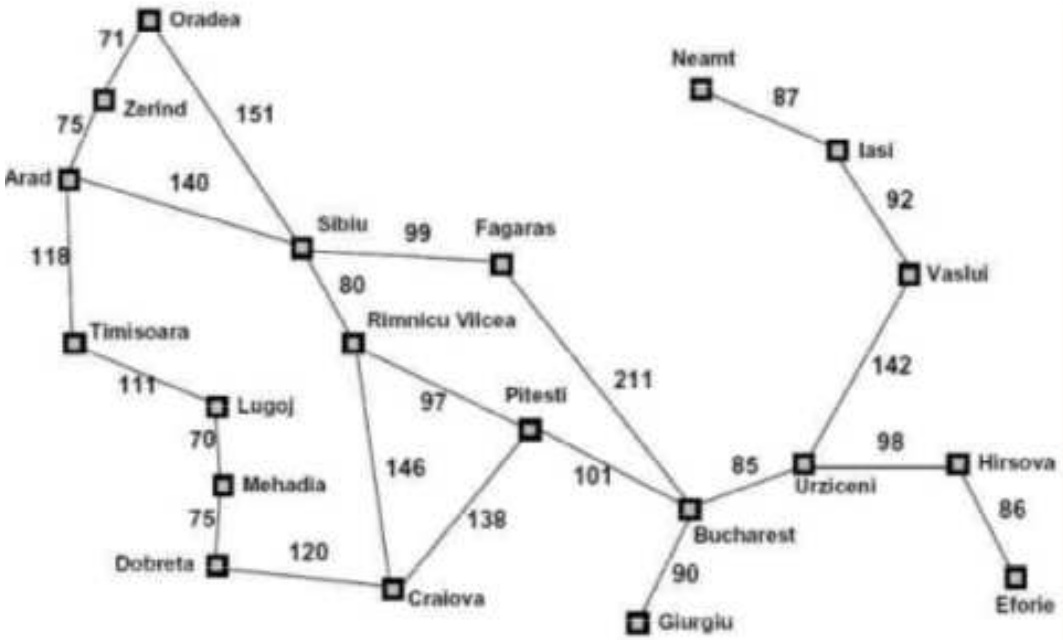
La Heurística





- Si el algoritmo expande el estado que piensa es el mas próximo a la meta...
  - La Heurística estima la distancia a la meta, para cada nodo.
  - Esta heurística baja en la medida que avancemos, lógicamente la heurística de la meta será 0.
  - No controla el costo de ir de un estado anterior al siguiente estado.
- ...el caso común...
  - Brinda soluciones no óptimas,
  - Lleva directamente a la meta de forma erronea.
- ...El peor de los casos..
  - Se convierte en un algoritmo primero en profundidad mal guiado.

Algoritmo A*



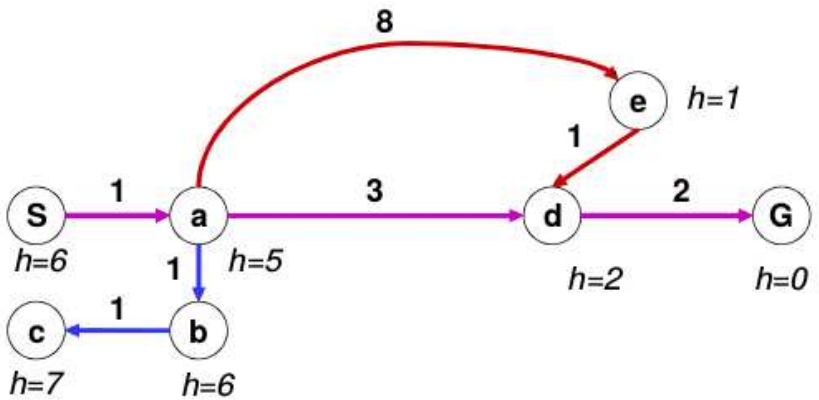
Straight-line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

h(x)

Combinando Algoritmos

Costo uniforme: Utilizamos una función para el costo del camino $g(n)$

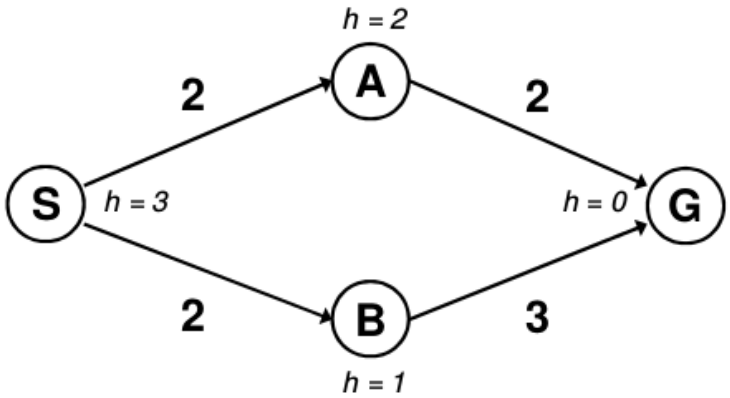
Búsqueda Codiciosa: Utilzamos la estimación de cuanto falta hacia la meta $h(n)$



Búsqueda A*: Utiliza una función f tal que : $f(n) = g(n) + h(n)$

Cuando termina el Algoritmo A* ?

El algoritmo se detiene al encolar el estado objetivo?



NO, el algoritmo termina solo al desencolar el estado objetivo.

Es el Algoritmo A* óptimo?