INTRODUCCIÓN A LA INTELIGENCIA ARTIFICIAL

MÓDULO 4: BÚSQUEDAS CON INFORMACIÓN

4.1. INTRODUCCIÓN

La búsqueda de solución a un problema se puede formalizar por un par (S; f) donde el espacio solución S denota el conjunto finito de todas las posibles soluciones y f denota la función de costo:

$f : S \rightarrow R$

Se trata (en minimización) de encontrar una solución $i_{opt} \in S$ tal que

$f(i_{opt}) \leq f(i)$ para toda $i \in S$.

En general, el espacio de solución S no esta dado en forma explícita, porque no se conoce de antemano o es muy grande como para almacenarlo. Una forma más compacta de describir el espacio de soluciones de un problema se logra por medio de la representación de un estado inicial, un conjunto de reglas de transformación y la descripción del estado solución o meta. Al aplicar las reglas de transformación al estado incial y a sus sucesores, se genera en forma explícita el grafo de estados del problema.

Formalmente, un grafo consiste de un conjunto de nodos, no necesariamente finito, donde ciertos pares de nodos se conectan por medio de arcos dirigidos desde un nodo al otro. Los grafos así definidos, son estructuras que permiten mantener el rastro de los efectos de la aplicación de varias secuencias de reglas a una base de datos inicial. Cada nodo del grafo representa un estado del problema, esto es una nueva base de datos, alcanzado como resultado de la aplicación de una regla. Las reglas que permiten evolucionar de un nodo a otro se representan con un arco. El nodo al que se le aplica la regla se denomina padre y es el origen del arco. El nuevo nodo obtenido como resultado se denomina hijo, y el arco debe etiquetarse con el nombre de la regla aplicada.

Los árboles son un caso particular de grafo.

Los grafos permiten implementar diferentes estrategias de búsqueda, estas estrategias pueden usar procedimientos generales que se aplican a todos los problemas o conocimiento específico del problema que permite limitar el espacio de búsqueda, acelerando la solución. Entendiendo por solución del problema a la "Secuencia de reglas que transforma el estado inicial en estado meta"

4.2. PROCEDIMIENTO GENERAL DE BÚSQUEDA EN GRAFO

A continuación se describe un procedimiento general para generar un grafo **G** de manera explícita a partir de un grafo definido implícitamente.

1- Cree un grafo de búsqueda **G** y una lista **ABIERTA** y ponga a **S** en cada uno de ellos.
2- Cree una lista vacía **CERRADA**.
3- Si **ABIERTA** esta vacía retorne con **Falla**.
4- Remueva el 1er nodo = **n** de **ABIERTA** y colóquela en **CERRADA**.
5- Si **n** = **Meta** retorne con Éxito.
6- Genere los **M** sucesores de **n** que no sean sus ancestros y póngalos en **G**.
7- Establezca punteros a **n** desde sus descendientes que no estén ni en **ABIERTA** ni en **CERRADA**, agregue los **M** sucesores a la lista **ABIERTA** y en caso que ya estuvieran redireccione los punteros si es un camino mas corto.
8- Reordene **ABIERTA**.
9- Vaya al punto 3

Cuando el procedimiento termina exitosamente, el resultado se obtiene siguiendo el camino de los punteros desde **n** hasta **s** en **G**.

4.3. USO DE FUNCIONES DE EVALUACIÓN

La forma general de la función de evaluación es

$$f(n) = g(n) + h(n)$$

donde g(n) es es costo de llegar hasta el nodo n desde el nodo de salida s y h(n) es el costo estimado para llegar al nodo meta. Entonces f(n) es el costo estimado total restringido a pasar por el nodo n

$f^*(n) = g^*(n) + h^*(n)$ camino óptimo de s hasta la meta pasando por n

$g^*(n) = K(s,n)$ costo óptimo desde s hasta n

Si en el paso 8 del procedimiento de búsqueda general en grafo, usamos una función heurística como la descripta entonces a ese algoritmo se lo llama Algoritmo A

Si además se le agrega la restriccion de estimación minorante,

$$h(n) \leq h^*(n)$$

Recibe el nombre de Algoritmo A* (se lee 'A estrella') que tiene propiedades interesantes que pasaremos a mostrar:

Resultado 1: Busqueda en grafo simpre termina para grafos finitos.

Resultado 2: En cualquier momento antes de terminar En la lista ABIERTA hay un nodo **n'** que esta en el camino óptimo con:

$$f(n') \leq f^*(s)$$

Resultado 3: Si hay camino entre **s** y la meta A* termina.

Resultado 4: A* es admisible (siempre termina con el camino óptimo, cuando este existe)

Resultado 5: Cualquier nodo seleccionado para expansión por A* $f(n) \leq f^*(s)$

Resultado 6: Si A1 y A2 son dos versiones de A* y A2 es mas informada que A1 entonces cualquier nodo expandido por A2 tambien lo será por A1.

Resultado 7: si se cumple la restricción monotónica entonces A* ya encontró el camino optimo a todo nodo que es seleccinado para expansión es decir

$$g(n) = g^*(n)$$

La restriccion monotónica se puede expresar como:

$h(ni) - h(nj) \leq c(ni,nj)$ donde c(ni,nj) es el costo para ir del nodo i al j

y además $h(t) = 0$ donde t es un nodo terminal o meta

Resultado 8: Si se satisface la condición de monotonicidad de h entonces la función de evaluación f de los nodos expandidos por A* es no decreciente.

Puede verse que cuando h=0 para todo nodo n la búsqueda es totalmente desinformada y el algoritmo de búsqueda se reduce a abrir primero.

Dadas dos heurísticas, si h1>h2 para todo n, se dice que h1 es mas informada que h2

4.4. PROBLEMAS PROPUESTOS

4.4.1 Problema 1

Especificar una base de datos global, reglas y una condición de terminación para un sistema de producción que resuelva el siguiente problema de la jarra de agua:

Dada una jarra de 5 litros llena con agua y una jarra de 2 litros vacía. ¿Cómo uno puede obtener precisamente 1 litro en la jarra de dos litros?
El agua puede ser descartada o volcada de una jarra a otra; sin embargo no se pueden utilizar más agua que los 5 litros iniciales.

4.4.2 Problema 2

Especificar una base de datos global, reglas y una condición de terminación para un sistema de producción que resuelva el problema de los misioneros y los caníbales:

Tres Misioneros y tres Caníbales vienen a un río. Hay un bote de su lado del río que puede ser usado por una o dos personas. ¿Cómo deben usar este bote para cruzar el río, tomando en cuenta que nunca y en ninguna de las orillas los Caníbales pueden superar numéricamente a los Misioneros?

4.5. REFERENCIAS BIBLIOGRÁFICAS

Inteligencia Artificial – Russell and Norvig – Cap.4.
Principles of Artificial Intellingence – Nils Nilsson Ch.2

Busquedas Heurísticas Eureka

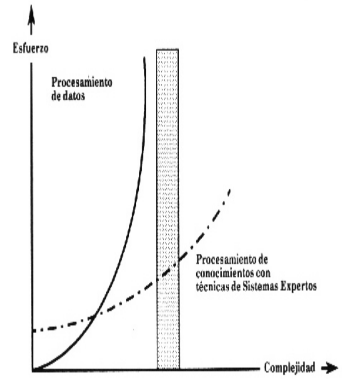

Figura 2.1: El procesamiento de conocimientos permite superar la barrera de la complejidad

Procedimiento general de Busqueda en grafo

1- Cree un grafo de búsqueda G y una lista ABIERTA y ponga a S en cada uno de ellos
2- Cree una lista vacía CERRADA
3- Si ABIERTA esta vacía retorne con Falla
4- Remueva el 1er nodo =n de ABIERTA y colóquela en CERRADA
5- Si n = Meta retorne con Éxito
6- Genere los M sucesores de n que no son ascendientes y póngalos en G
7- Establezca punteros de a sus descendientes que no estén ni en ABIERTA ni en CERRADA, agrega los M sucesores a la Lista ABIERTA en caso que estuvieran redireccione los punteros si es un camino mas corto,
8- Reordene ABIERTA
9- Vaya a 3

FUNCIONES DE EVALUACION

La forma general
f(n) = g(n) + h(n)

g(n) costo para llegar hasta n desde salida s
h(n) costo *estimado* para llegar a t (meta)
f(n) costo *estimado* total que pasa por n

f*(n) = g*(n) + h*(n)
camino óptimo de s hasta t pasando por n

g*(n) = K(s,n) costo óptimo desde s hasta n

Si en el paso 8 se usa h entonces algoritmo A

Restricción de estimación minorante
h(n) ≤ h*(n)

Entonces es Algoritmo A*

Propiedades de A*

Resultado 1: Busqueda en grafo simpre termina para grafos finitos.

Resultado 2: En cualquier momento antes de terminar En la lista ABIERTA hay un nodo n' que esta en el camino óptimo con
f(n') ≤ f*(s)

Resultado 3: Si hay camino entre s y la meta A* termina

Resultado 4: A* es admisible (siempre termina con el camino óptimo, cuando este existe)

Resultado 5*: Cualquier nodo selccionado para expansión por A
f(n) ≤ f*(s)

Propiedades de A*

***Resultado 6*: Si A1 y A2 son dos versiones de A* y A2 es mas informada que A1 entonces cualquier nodo expandido por A2 tambien lo será por A1.**

***Resultado 7*: si se cumple la restricción monotónica entonces A* ya encontró el camino optimo a todo nodo que es seleccionado para expansión es decir**

$g(n) = g^*(n)$

La restriccion monotónica

$h(ni) - h(nj) \leq c(ni,nj)$ donde $c(ni,nj)$ es el costo para ir del nodo i al j

$h(t)=0$ donde t es un nodo terminal

Resultado 8: Si se satisface la condición de monotonicidad de h entonces la función de evaluación f de los nodos expandidos por A* es no decreciente.

Templado simulado

- **Estado inicial= S**
- **T = caliente**
- **Repita hasta que T= frio**
- **{Perturbe el sistema (genere un estado al azar con distancia proporcional a T)**
- **Calcule la energía del sistema**
- **Si ΔE es < 0**
  - **Entonces acepte el nuevo estado**
  - **Sino acepte el nuevo estado con probabilidad = exp($-\Delta E/KT$}**

  – **Si ΔE <0 decreció en las últimas x temperaturas**
  – **Entonces T=0.9*T**
  – **Sino T=frío**
  – **}**
  – **Retorne con el sistema en el último estado**

Recocido simulado

Algoritmo de Metropolis.

- Dado un estado (solución) i con energía Ei
- Genera un nuevo estado j mediante un mecanismo de perturbación,(distorsión aleatoria del estado i).
- Calcula la energía del nuevo estado Ej
- Si Ej < Ei

  Entonces acepta el estado j como estado nuevo
- Sino, acepta el estado con probabilidad:

  exp [(Ei-Ej)/KB.T]

  donde T es la temperatura del sistema

  KB es la constante de Boltzman

Recocido Simulado

- **Estado inicial= T**
- **T = caliente**
- **Repita hasta que T= frío**
- **{Perturbe el sistema (genere un estado al azar con distancia proporcional a T)**
- **Calcule la energía del sistema**
- **Si ΔE es < 0**
  - **Entonces acepte el nuevo estado**
  - **Sino acepte el nuevo estado con probabilidad = exp(-ΔE/KT}**

- **Si ΔE <0 decreció en las últimas x temperaturas**
- **Entonces T=0.9*T**
- **Sino T=frío**
- **}**
- **Retorne con el sistema en el último estado**

Algoritmo Recocido Simulado

- i := iinic ; i es la solución actual
- f(i) := f eval inicial ; f es la función de evaluación

- Repite
- For T:= Tinic to 1 do ; Tinic es la temperatura Inicial del sistema
- {Generar j ϵ Sj; (S es el espacio de soluciones)
- **Si f(j) < f(i)**

  entonces i := j
- sino, si exp[f(i)-f(j)/C.T] > random [0,1]

  entonces i := j
- T:= T-1

- donde Cϵ R+ denota el parámetro de control.
- Si f(i)= f(i_{T+n}) entonces return f(i)
- }