Instituto Tecnológico de Ciudad Juárez

Carrera: Ingeniería en Sistemas Computacionales

Tema 3

Alumno:
Víctor Manuel Nava **Nc:**16112052

Reglas y búsquedas.

Búsqueda Heurística

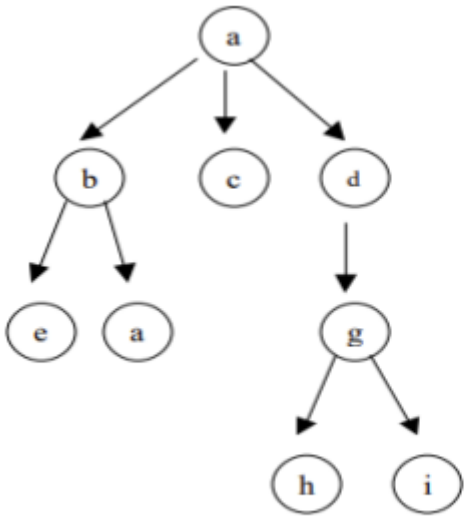
La búsqueda heurística es utilizada para ordenar y ver primero los nodos más prometedores y para controlar el ancho de la búsqueda. Como ejemplo para este tipo de búsqueda podemos encontrar los mapas usados para encontrar el camino óptimo y menos costoso en base a la lo logística, dado que esta búsqueda analiza todos los resultados posibles y todos los caminos existentes, por lo tanto podemos conocer los costos totales de cada uno de los caminos para al finar tomar el menos costoso.

Uno de los grandes problemas de este algoritmo de búsqueda es que deja mucho que desear en múltiples casos, por ejemplo en videojuegos no se puede usar para inteligencia artificial dado que cuando un personaje enemigo tiene la destreza de perseguirnos por un mapa usa un mapa especial de colisiones para así ver cuáles son los caminos que no están bloqueados. Si se llega a usar la búsqueda heurística desde el enemigo hasta el punto donde se encuentra el personaje principal y suponiendo que está cada uno detrás de la pared, el enemigo que usa esa búsqueda intentará traspasar la pared porque es el camino más corto para llegar.

Búsqueda Ciega

Búsqueda ciegas : estrategias de búsqueda de soluciones que no explotan información adicional que pueda guiar el proceso. las estrategias básicas son la búsqueda de anchura y la búsqueda de profundidad.

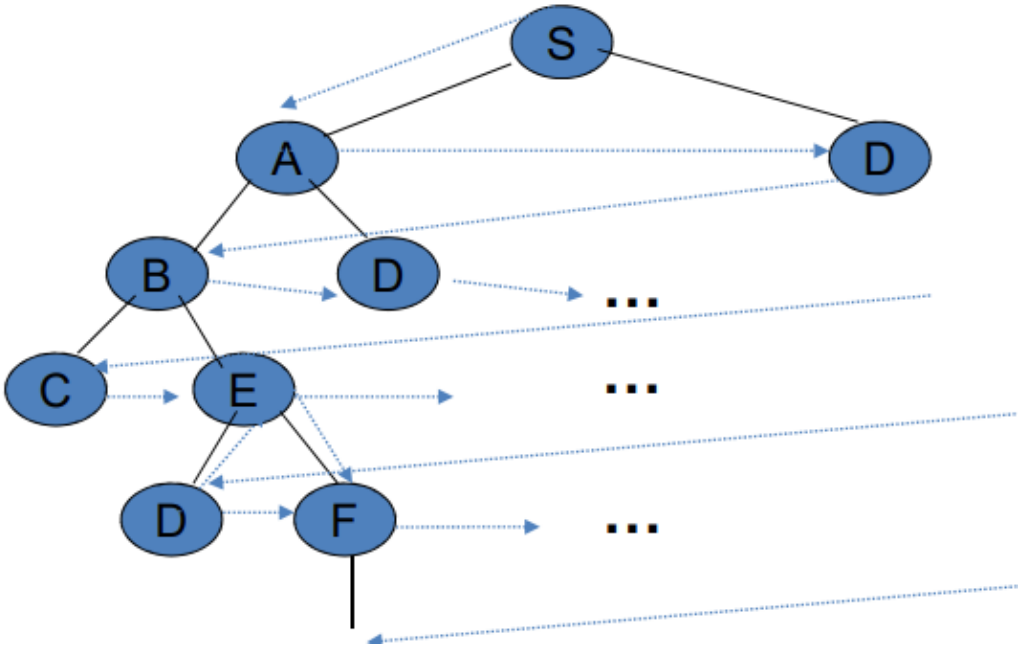
Busqueda en profundida:



Las ramas son: a, b, e a, b, a a, c a, d, g, h a, d, g, i.

Para este proceso se debe tener en cuenta que una hoja de un árbol es parte de un nodo que no tiene un sucesor, y una rama de árbol es un camino que comienza desde un nodo raíz y termina en una hoja. Para lo cual la imagen de arriba lo demuestra.

Búsqueda en anchura:



Lo que se trata este busqueda es primero expandir el nodo menos profundo que no haya sido expandido.el algoritmo detrás es

crear: una lista con solo elemento consiste en una trayectoria o camino de longitud cero(nodo raíz), hasta que el primer camino de la lista llegue al nodo objetivo o se llegue a la lista vacía, si hacía vacía debemos hacer

- ☐ Extraer el primer camino de la lista
- ☐ expandir el nodo final de este camino a todo los vecinos del nodo terminal
- ☐ eliminar los ciclos de los caminos expandidos
- ☐ insertar estos nuevos caminos al final de la lista.

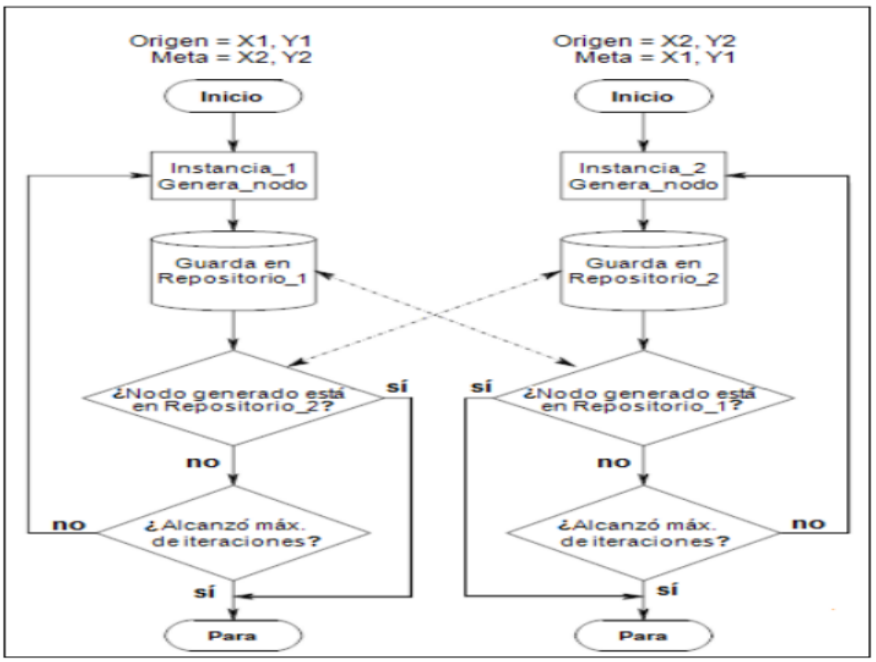
después de cumplir de hallar el nodo meta es cuando termina pero si no hay fracaso en la búsqueda.

Bidireccional

En la búsqueda bidireccional, el problema consiste en trazar una ruta para ir desde el origen hasta la meta; mientras de forma paralela y simultánea, se intenta trazar otra ruta que conduzca de la meta al origen.

Puede darse el caso en el que la ruta que se está construyendo de origen a meta, se cruce con la que se está construyendo para ir de la meta al origen, caso en el cual, se habrá establecido un camino para unir los dos puntos.

A continuación, se representa de forma simplificada el algoritmo de la búsqueda bidireccional, donde el origen en el diagrama de flujo de la izquierda, se convierte en la meta para el de la derecha y de forma contraria, la meta en el de la izquierda es el origen en el de la derecha.



Métodos de inferencia en reglas

Modus Ponendo Ponens

- “Si quieres pasar el examen de mañana, entonces debes estudiar mucho.”
    - P: Quieres pasar el examen de mañana.
    - Q: Por lo tanto, debes estudiar mucho.
- “Si quieres llegar rápido a la escuela, entonces debes tomar ese camino.”
    - P:Quieres llegar rápido a la escuela.
    - Q: Por lo tanto, debes tomar ese camino.
- “Si quieres comer pescado, entonces debes ir a comprar en el mercado.”
    - P: Quieres comer pescado.
    - Q: Por lo tanto, debes ir a comprar en el mercado.

Modus Ponendo Tollens

- “Alejandra y Bárbara no pueden ganar ambas la carrera.”P: Alejandra ganó la carrera. Q: Por lo tanto, Bárbara no puede haber ganado la carrera. Modus.
- Ponendo Tollens 2: Premisa 1: Es culpable o Inocente. Premisa 2: Es inocente. Conclusión: Por lo tanto, no es culpable.
- Modus Ponendo Tollens 3: Premisa 1: Si hay nubes va a llover, si hay sol no lloverá. Premisa 2: Hay nubes. Conclusión: Por lo tanto, va a llover.

Modus Tollendo Ponens

- “O el incumplimiento es una violación de seguridad, o no está sujeto a multas.”
    - P: El incumplimiento no es una violación de seguridad.
    - Q: Por lo tanto, no está sujeto a multas.
- “Elegiré sopa o elegiré ensalada.”
    - P: No voy a elegir sopa.
    - Q: Por lo tanto, voy a elegir ensalada.
- “Es de color rojo o azul.”
    - P: No es azul.
    - Q: Por lo tanto, es de color rojo.
- “Si tiene luz propia, entonces el astro es una estrella.”
    - P: El astro no es una estrella.
    - Q: Por lo tanto, no tiene luz propia.

Reglas de producción

La importancia en la IA de los sistemas basados en reglas es grande, la mayoría de los sistemas expertos deterministas (que veremos más adelante) están basados en este tipo de representación. Los sistemas basados en reglas consisten de un conjunto de reglas de la forma if-then, es decir se utiliza la implicación como principal forma de representación.

Sintaxis:

Las reglas expresan las transformaciones que pueden efectuarse sobre el estado actual de resolución. Inicialmente sobre el estado inicial y a medida que realizamos acciones sobre los nuevos estados. Las reglas entonces son:

IF condición **THEN** acción

A diferencia de la programación clásica, donde implementamos un algoritmo, en un sistema basado en reglas la ejecución puede ser vista como la búsqueda en un espacio de estados.

Las reglas deben ser manejadas por un intérprete. El mismo puede tener diversas estrategias de control. Debería existir una base de datos global que contenga toda la información relevante al problema.

Forward chaining: Es denominado también data-driven y bottom-up (manejado por datos y de abajo hacia arriba). Parte de los datos iniciales, aplica las reglas al estado inicial y a los sucesores de éste hasta llegar al estado final.

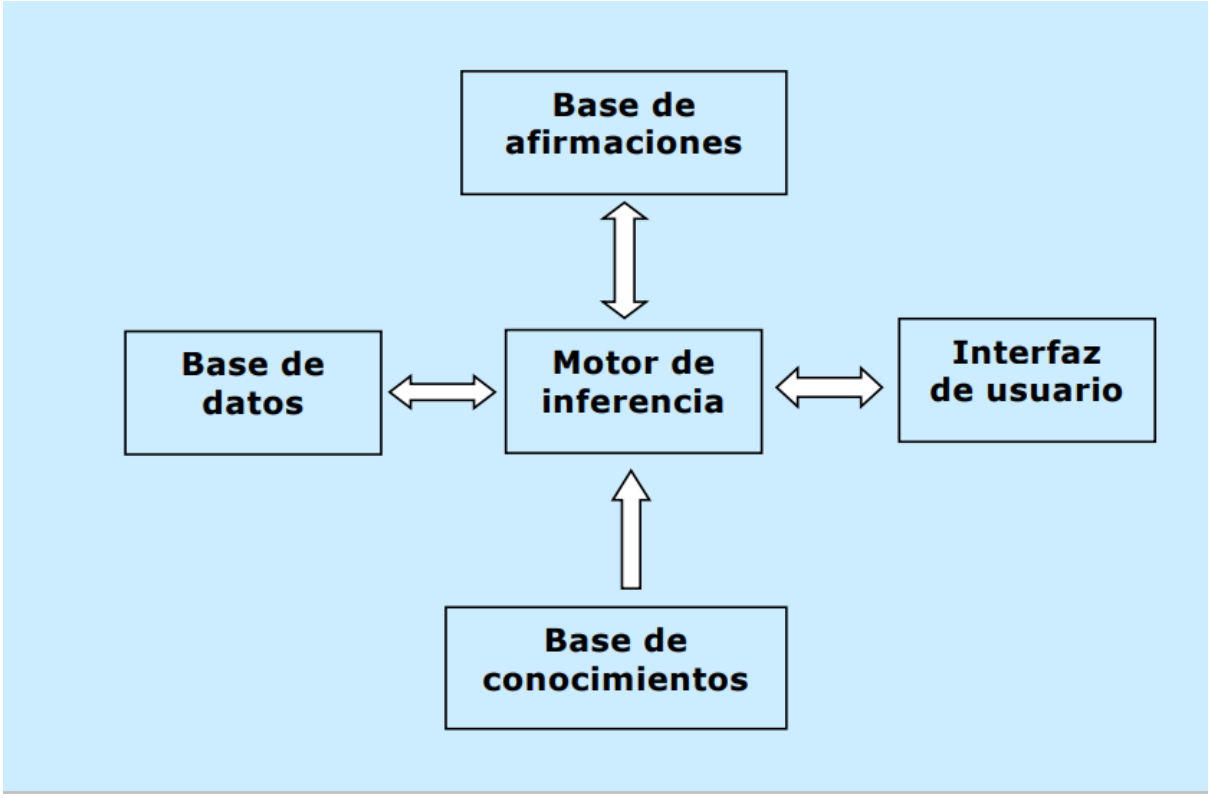
Backward chaining: Es denominado también goal-driven y top-down (manejado por el objetivo y de arriba hacia abajo). Se parte del objetivo

(una hipótesis que se quiere probar) y se busca que reglas permiten esta hipótesis. Ahora se trata de probar las condiciones que disparan esas reglas, las cuales se transforman en subobjetivos, y éstos a su vez se transforman en otros subobjetivos, así hasta llegar al estado inicial.

Extracción: obtener el conjunto de reglas plausibles de ser utilizadas, esto dependerá de la estrategia utilizada. En forward chaining se elegirán aquellas que tengan su condición satisfecha en la base de datos global, mientras que en backward chaining se elegirán aquellas cuya conclusión está en el objetivo o subobjetivos actuales.

Refinamiento: Se refinan las reglas obtenidas (y las pendientes de pasos anteriores si las hubiera) eliminado algunas. Selección: Se elige una regla del conjunto refinado (es posible elegir más de una incluso alguna estrategia plantea elegir todas y no refinar).

Componentes de Sistema de producción



Inferencia

Una regla cuando se cumple su antecedente, todas las cláusulas que lo componen. Las reglas se ejecutan hacia adelante: si se satisface el antecedente se efectúan las acciones del consecuente.

Los campos en los que más se han desarrollado sistemas expertos basados en reglas son los siguientes:

- Medicina
- Geología
- Química
- Economía
- Ingeniería Civil