

IAAAAAA

Las definiciones de Inteligencia artificial pueden agruparse en 4 categorías: Pensar humanamente| Pensar racionalmente| Actuar humanamente |Actuar racionalmente,| La búsqueda puede ser informada, si el agente conoce el resultado de la transición entre cada estado (por ejemplo, conocer el mapa de la ciudad y el tiempo estimado para cruzar de una calle a otra), **o desinformada** si no la conoce. | **Formulación del problema de búsqueda = Formular, buscar, ejecutar | open loop =** Se refiere a un sistema de control donde el agente ejecuta acciones sin considerar las percepciones actuales del ambiente.|**Infraestructura de algoritmos de búsqueda=** Para cada nodo del árbol, tenemos una estructura que contiene cuatro componentes: □ **.ESTADO:** el estado en el espacio de estado al que corresponde el nodo. □ **.PADRE:** el nodo en el árbol de búsqueda que generó este nodo. □ **.ACCION:** la acción que se aplicó al padre para generar el nodo. □ **.PATH-COST:** el costo de la ruta desde el estado inicial hasta el nodo . Tradicionalmente denotado por $g(n)$.|**Manejando la frontera :** Las colas se caracterizan por el orden en que almacenan los nodos insertados. □ Las variantes más comunes son: □ Primero en salir o **FIFO**, que hace aparecer el elemento **FIFO QUEUE** más antiguo de la cola. □ **Cola LIFO** último en entrar, primero en salir o también conocida como pila, que muestra el elemento más nuevo. □ Cola de prioridad, que muestra el elemento de la cola con la prioridad más alta según alguna función ordenadora| Las colas se caracterizan por el orden en que almacenan los nodos insertados. □ Las variantes más comunes son: □ Primero en salir o **FIFO**, que hace aparecer el elemento **FIFO QUEUE** más antiguo de la cola. □ **Cola LIFO** último en entrar, primero en salir o también conocida como pila, que muestra el elemento más nuevo. □ **Cola de prioridad**, que muestra el elemento de la cola con la prioridad más alta según alguna función ordenadora |**Criterios de desempeño de un algoritmo de búsqueda** □ Podemos evaluar el rendimiento de un algoritmo en cuatro maneras: □ **Integridad o Completitud:** ¿se garantiza que el algoritmo encuentre una solución cuando la hay? □ **Optimalidad:** ¿La estrategia encuentra la solución óptima? □ **Complejidad del tiempo:** ¿Cuánto tiempo lleva encontrar una solución? □ **Complejidad del espacio:** ¿Cuánta memoria se necesita para realizar la|**tipos de estrategias de busqueda desinformadas =** □ **Breadth-first search** (Búsqueda en amplitud) □ **Uniform cost search** (Búsqueda de costo uniforme) □ **Depht-first search** (Búsqueda en profundidad) |**Breadth-first search =** La búsqueda en amplitud es una estrategia simple en la que el nodo raíz se expande primero, luego todo se expanden todos los sucesores del nodo raíz, luego sus sucesores, y así sucesivamente. el mejor ejemplo para este metodo de busqueda es una cola del supermercado (**¿Es completo?:** Si, ya que si el nodo final más superficial está a profundidad , BFS lo encontrará cuando genere todos los nodos hasta dicha profundidad. □ **¿Es óptimo?:** No necesariamente el nodo final más superficial generará la solución con menor costo (más óptima). La función de costo tendría que ser uniforme (todas las acciones con el mismo costo) para alcanzar el óptimo)| **uniform cost search** En lugar de expandir el nodo más superficial, búsqueda de costo uniforme expande el nodo con el costo del camino más bajo . Esto se hace almacenando la frontera como un cola de prioridad ordenada por g |**¿Es completo?:** Si, ya que solo se prueban los nodos finales que se expanden. Y solo se expandirá el nodo si es un camino de costo mínimo. Considerando por supuesto que los costos son todos positivos. Ya que si hay costos cero, el algoritmo podría caer en loop infinito|**Depth-First Search =** baja por la primera rama que encuentre sin importar el costo hasta que no recorra todos los nodos de dicha rama no pasara a la otra **¿Es completo?:** No, ya que puede caer en ramas de largo infinito si es que cae en loops. □ Para evitarlo hay que revisar que el estado no se repita dentro del camino. □**¿Es óptimo?:** Tampoco es óptima ya que si podría encontrar una solución en otra rama diferente la de menor costo