1.- El concepto de Agente. Agentes Racionales vs. Agentes Inteligentes. Arquitecturas de Agentes.

Un agente es un ente que percibe su entorno mediante unos sensores y procede en consecuencia mediante unos actuadores, estos actuadores se activan por los estímulos del medio.

Un agente racional es aquel que actúa con la intención de obtener el mejor resultado posible, si tiene incertidumbre actúa para obtener el mejor resultado esperado.

Un agente inteligente son los únicos capaces de pasar el test de Turing.

- Arquitectura reactiva: actúa en función al entorno que perciben a través de sus sensores, a lo denominado estímulo-respuesta. Operan rápido y eficazmente sin la necesidad de una representación interna del entorno ni de hacer uso de decisiones pasadas. Aunque también existen los agentes reactivos con memoria, que estos sí que almacenan acciones tomadas en el pasado y pueden llegar a llevar una pequeña representación del mundo en el que se mueven para poder así tomar una decisión más correcta la próxima vez que se vaya a tomar una.

- Arquitectura deliberativa: este tipo de agentes cuentan con un modelo del entorno donde se encuentran y lo utilizan para planificar sus acciones para conseguir su objetivo impuesto.

- Arquitectura híbrida: una mezcla entre reactivo y deliberativo, para muchos problemas es la recomendada, ya que si por ejemplo, durante el desarrollo de un plan de un agente deliberativo ocurren problemas se necesitan decisiones reactivas para poder esquivar el problema y continuar con la ejecución normal del agente.

- Arquitectura horizontal: recorre su lista de acciones posibles por capas hasta llegar a la última y entonces ejecutar la acción correspondiente.

- Arquitectura vertical: pueden ser de una pasada que entonces actuarían como las arquitecturas horizontales o de dos pasadas, éstas recorren su lista de posibles acciones por capas hasta llegar a la última y después el flujo baja hasta la capa inferior de nuevo.

2.- Características de los Agentes reactivos y deliberativos. Similitudes y diferencias. Arquitecturas.

Un agente reactivo se caracteriza por tomar las decisiones guiado únicamente por sus sensores, no poseen un modelo simbólico ni pueden recordar acciones hecha anteriormente.

Un agente deliberativo se caracteriza por tener el entorno en un modelo simbólico y actúa lógicamente mediante equiparaciones de patrones y manipulación simbólica. Pueden

actuar de esta manera porque tienen un conocimiento que aplica sobre su modelo simbólico, permitiéndole modificarlo.

Similitudes:

- Tienen como objetivo alcanzar una meta.
- Tienen que partir con un conjunto de reglas o acciones con las que interactuar con el entorno.

Diferencias:

- El agente deliberativo necesita tener un modelo simbólico del entorno, mientras que el reactivo no.
- El agente reactivo basa sus acciones en lo que percibe en ese momento del entorno, mientras que el deliberativo procesa su acción con más información y conocimiento.
- El agente reactivo suele realizar sus acciones más rápidas que el agente deliberativo.

3.- Métodos de búsqueda no informada.

Los algoritmos de no informada no dependen de información propia del problema a la hora de resolverlo, sino que proporcionan métodos generales para recorrer los árboles de búsqueda asociados a la representación del problema, por lo que se pueden aplicar en cualquier circunstancia. Hay 6 tipos:

Búsqueda en profundidad: utilizado para recorrer todos los nodos de un grafo o árbol de manera ordenada, pero no uniforme. Su funcionamiento consiste en ir expandiendo todos y cada uno de los nodos que va localizando, de forma recurrente, en un camino concreto. Cuando ya no quedan más nodos que visitar en dicho camino, regresa de modo que repite el mismo proceso con cada uno de los hermanos del nodo ya procesado.

Búsqueda en anchura: utilizado para recorrer todos los nodos de un grafo o árbol. Intuitivamente, se comienza en la raíz y se exploran todos los vecinos de este nodo. A continuación, para cada uno de los vecinos se exploran sus respectivos vecinos adyacentes, y así hasta que se recorra todo el árbol.

Búsqueda en profundidad limitada: es una variante de la búsqueda en profundidad, donde se establece un límite de profundidad.

Búsqueda en profundidad iterativa: es una variante de la búsqueda en profundidad limitada, donde la cota de profundidad elegida va aumentando si no encuentra una solución.

Búsqueda en coste uniforme: es una variante del método "Primero el Mejor", utilizado para recorrer sobre grafos el camino de costo mínimo entre un nodo raíz y un nodo destino. La búsqueda comienza por el nodo raíz y continúa visitando el siguiente nodo que tiene

menor costo total desde la raíz. Los nodos son visitados de esta manera hasta que el nodo destino es alcanzado.

Búsqueda bidireccional: Es un método en el cual se realiza simultáneamente una búsqueda desde el estado inicial y otra búsqueda desde el estado objetivo.

4.- El concepto de heurística. Como se construyen las heurísticas. Uso de las heurísticas en IA.

La heurística trata de la resolución de problemas aplicando soluciones parciales, a menudo intuidas. Se evalúan los resultados intermedios obtenidos para aproximarse poco a poco al resultado o solución final. También se aplican atajos que funcionan, aunque no se sepan exactamente por qué. Existen estrategias, reglas y silogismos que ayudan a encontrar soluciones heurísticas. Una simplificación del concepto serían los métodos iterativos o de aproximaciones sucesivas. Algunas búsquedas de información en bases de datos interactivas online se desarrollan haciendo varias iteraciones hasta hallar la terminología adecuada que permite recuperar toda la información. Los métodos heurísticos se basan en la experiencia más que en la razón. Heurística es la capacidad de un sistema para realizar cambios o innovaciones positivas para sus fines de forma inmediata. La capacidad heurística es un rasgo característico de nuestra especie, desde cuyo punto de vista puede describirse como el arte y la ciencia del descubrimiento y de la invención o de resolver problemas mediante la creatividad y el pensamiento lateral.

Las heurísticas son criterios, métodos, o principios para decidir cuál puede ser la solución más efectiva hacia un objetivo, por tanto, se caracterizan en por tener criterios simples y poder diferenciar buenas de malas decisiones.

La construcción de funciones heurísticas puede considerarse un proceso de descubrimiento, algunos lo consideran un arte por ser tan difícil, una buena opción de construir una es a partir de una simplificación del problema. Una heurística admisible puede derivar de una versión simplificada del problema, o por información desde un patrón de base de datos que almacene la solución exacta de un problema, o usando aprendizaje inductivo.

Si se tiene de una heurística admisible, la que mejor funcione va a ser la de mayor valor, o sea, la que mejor aproxime la solución óptima real sin sobrepasar su valor. De ahí si tenemos varias heurísticas admisibles y no sabemos cuál elegir, podemos crear una nueva que sea el máximo de todas las otras.

Mientras que todas las heurísticas consistentes son admisibles, no todas las heurísticas admisibles son consistentes. Una heurística H(N) es consistente si, para todo nodo N y todo sucesor N' de N generado por cualquier acción A, el costo estimado de alcanzar el objetivo desde N no es mayor que el costo de obtener N’ más el costo estimado de obtener el objetivo desde N’.

Una heurística encapsula el conocimiento específico que se tiene sobre un problema y sirve como guía para resolverlo, aunque no garantice una solución óptima, en promedio

produce resultados satisfactorios. Se usa en juegos, recorridos de mapas y optimización de recursos entre otros.

5.- Los métodos de escalada. Caracterización general. Variantes.

Los métodos de escalada son un tipo de búsqueda con información basados en la búsqueda en profundidad. Son métodos de búsqueda local porque parten de un nodo inicial y profundizan en la búsqueda seleccionando el mejor hijo, realizando esta misma acción para el hijo y sus descendientes sólo si resulta ser una mejor opción que el padre. Por esta razón puede no encontrar una solución ya que se puede quedar en extremos locales, siempre que la función no sea monótona.

Este método intenta optimizar una función objetivo (f(x)), donde x es un conjunto de valores discretos y/o continuos. En cada iteración, el método de escalada ajustará un único elemento en x y determinará si el cambio mejora el valor de la función. Cualquier cambio que mejore la función es aceptado, y el proceso continúa hasta que no pueda encontrarse un cambio que la mejore. Se dice entonces que es "óptima localmente".

En la **escalada simple** se escoge el primer nodo cercano, mientras que en **escalada de ascenso pronunciado** todos los sucesores son comparados y la solución más cercana es elegida. Ambas formas fallan si no existe un nodo cercano, lo cual sucede si hay máximos locales en el espacio de búsqueda que no son soluciones. La escalada de ascenso pronunciado es similar a la búsqueda en anchura, la cual intenta todas las posibles extensiones del camino actual en vez de sólo una.

La **escalada estocástica** no examina todos los vecinos antes de decidir cómo moverse. En vez de eso, selecciona un vecino aleatorio, y decide (basado en la cantidad de progreso en ese vecino) si moverse a él o examinar otro.

El **descenso coordinado** realiza una búsqueda en línea a lo largo de una dirección de coordenadas a partir del punto actual en cada iteración. Algunas versiones del descenso coordinado eligen aleatoriamente una dirección coordinada diferente en cada iteración.

La **escalada de reinicio aleatorio** comienza cada vez con una condición inicial aleatoria, x0, y se va iterando sobre el conjunto de elementos guardándose la mejor xn (siendo n veces las iteradas). Cabe destacar que es sorprendentemente efectivo en muchos casos. La conclusión de este mejor gastar tiempo de CPU explorando el espacio, que cuidadosamente optimizar desde una condición inicial.

6.- Características esenciales de los métodos “primero el mejor”.

La búsqueda voraz primero el mejor se parece a la búsqueda primero en profundidad en el modo que prefiere seguir un camino hacia el objetivo, pero volverá atrás cuando llegue a un callejón sin salida. Sufre los mismos defectos que la búsqueda primero en profundidad, no es óptima, y es incompleta (porque puede ir hacia abajo en un camino infinito y nunca volver para intentar otras posibilidades). La complejidad en tiempo y espacio, del caso peor, es O(bm), donde m es la profundidad máxima del espacio de búsqueda. Con una buena función, sin embargo, pueden reducir la complejidad considerablemente. La cantidad de la reducción depende del problema particular y de la calidad de la heurística.

Se utiliza, normalmente un árbol en el cual se va iterando, cogiendo el hijo con mejor heurística hasta que no quedan más hijos, entonces se vuelve al padre del nodo actual, se escoge el hijo que mejor tiene la heurística, sin contar por el que ya se ha pasado y se repite el proceso hasta que se haya recorrido todo el árbol.

7.- Elementos esenciales del algoritmo A*.

Se trata de un algoritmo “Primero el mejor” donde se guía por una fórmula:

- **F(x) = G(x) + H(x)**

G(x): es la distancia del mejor camino desde el nodo inicial hasta el actual, en este caso **x**.

H(x): es la heurística que se está aplicando, que es la distancia desde el nodo actual al nodo objetivo.

En cada paso se selecciona el nodo con mejor valor que se haya analizado. Cuando se elige a uno, se expanden sus hijos y son analizados, si alguno de ellos es el nodo objetivo acaba el algoritmo. En otro caso se realiza una búsqueda en anchura, si no aparece la solución por la rama que se está recorriendo, se escoge otra hasta dar con la solución.

Por último, tiene que satisfacer el Teorema de Admisibilidad:

Si h'(x) nunca sobrestima a h(x) entonces A* es admisible.

8.- Elementos esenciales de un algoritmo genético.

Los algoritmos genéticos son métodos sistemáticos para resolver problemas de búsqueda y optimización que aplican estrategias biológicas: selección basada en el más adaptado, reproducción sexual y mutación.

El fin de estos algoritmos es encontrar una solución óptima para una cierta función objetivo. Se utilizan los tecnicismos biológicos:

- Cromosoma: Representación de una solución al problema.

- Gen: Atributo concreto del vector de representación de una solución.
- Población: Conjunto de soluciones al problema.
- Adecuación al entorno o fitness: Valor de la función objetivo.
- Selección natural: Operador de selección.
- Reproducción sexual: Operador de cruce.
- Mutación: Operador de modificación.
- Cambio generacional: Operador de reemplazamiento.

Funcionamiento:

Se genera aleatoriamente la población inicial, que está constituida por un conjunto de cromosomas los cuales representan las posibles soluciones del problema.

A cada uno de los cromosomas de esta población se aplicará la función de aptitud para saber cómo de "buena" es la solución que se está codificando.

El algoritmo se deberá detener cuando se alcance la solución óptima, pero esta generalmente se desconoce, por lo que se deben utilizar otros criterios de detención. Normalmente se usan dos criterios: un número máximo de iteraciones (generaciones) o detenerlo cuando no haya cambios en la población. Mientras no se cumpla la condición de término se hace lo siguiente:

Selección: Después de saber la aptitud de cada cromosoma se procede a elegir los cromosomas que serán cruzados en la siguiente generación. Los cromosomas con mejor aptitud tienen mayor probabilidad de ser seleccionados.

Recombinación o cruzamiento: La recombinación es el principal operador genético, representa la reproducción sexual, opera sobre dos cromosomas a la vez para generar dos descendientes donde se combinan las características de ambos cromosomas padres.

Mutación: Modifica al azar parte del cromosoma de los individuos, y permite alcanzar zonas del espacio de búsqueda que no estaban cubiertas por los individuos de la población actual.

Reemplazo: Una vez aplicados los operadores genéticos, se seleccionan los mejores individuos para conformar la población de la generación siguiente.

Un algoritmo genético no garantiza la obtención del óptimo, pero, si está bien construido, proporcionará una solución razonablemente buena.