La Torre de Babel, por Pieter Bruegel el viejo (1563)

Proyecto N° 1

Solucionador de la Torre de Babilonia

IC-6200 | Inteligencia Artificial | 24 de Agosto de 2018

Especificación de Proyecto Práctico Grupal N°1

Descripción general del proyecto

Objetivo didáctico: poner en práctica la técnica de solución de problemas mediante búsqueda en un espacio de estados usando A*.

Objetivo general del proyecto: Existe un juguete llamado (en Internet) "Torre de Babilonia", que consiste en una especie de matriz cilíndrica de bolitas, donde cada fila puede desplazarse hacia izquierda o derecha alrededor de su eje, y un campo libre en la parte superior del juguete permite que una sola de las columnas pueda ser desplazada hacia arriba. En la fotografía se puede ver claramente que es posible desplazar 1, 2, 3 o las 4 bolitas a elección del jugador. El juego consiste en que usando los desplazamientos horizontales y verticales es posible poner el juguete en una configuración arbitraria, y, a partir de ella, el jugador debe, usando ese mismo repertorio de movimientos, colocar las bolitas en alguna configuración dada.

El objetivo de este proyecto es el de crear un programa que sea capaz de recibir la disposición inicial de bolitas, recibir también la configuración final que se desea, e indicarle al usuario qué movimientos hay que hacer para lograr esta a partir de aquella.

Objetivos específicos

1. Ejercitar a los estudiantes en las técnicas de solución de problemas como búsqueda en un espacio de estados
2. Ejercitar a los estudiantes en la representación de conocimiento de una ambiente específico, y su manipulación desde y hacia un usuario

Descripción de la ejecución para el usuario: Al usuario se le solicita la posición inicial de las bolitas en el juguete (tome en cuenta que la bolitas no están marcadas, así que no se le puede pedir al usuario que las individualice), y también se le pedirá la configuración final; a continuación, el programa lo "pensará" un rato para, finalmente, decirle al usuario, paso a paso, cómo debe manipular el juguete para llegar a la configuración final. Es muy importante tomar en cuenta que vamos a suponer que la sección gruesa sin bolitas (en donde está la muesca vacía) es la parte de arriba. Los estudiantes deberán crear asimismo un pequeño lenguaje formal para describir el juguete y su manipulación, pues se exigirá también que ya sea la configuración inicial o la meta, o ambas, puedan ser dadas mediante un archivo tipo txt, o

mediante la solicitud de una hilera en una ventana o cuadro de diálogo. Asimismo, si el usuario lo desea, las configuraciones inicial y final, y las instrucciones de armado deberán quedar en un archivo de texto.

Requerimientos técnicos obligatorios: El programa deberá resolver el problema planteado mediante la técnica A*, puesto que este es el verdadero objetivo en este proyecto; la ausencia de esta exigencia volvería inaceptable el trabajo de los estudiantes. No obstante, la interfaz de usuario será también muy tomada en cuenta, puesto que, en general, esta suele ser un factor crítico en la usabilidad del software.

Documentación: La interfaz de usuario deberá ser lo más explícita posible, de manera que las instrucciones aparezcan en ella. Si estas son muy extensas, deberán estar disponibles mediante el uso de un menú pull-down. En otras palabras, no deberá haber manual de usuario en forma impresa.
La documentación escrita del proyecto será un documento con las siguientes partes:

- Portada: Figurarán
    a. el nombre del trabajo
    b. Institución (el TEC) y carrera
    c. la materia (nombre y código)
    d. el semestre, y los autores, en orden alfabético por apellido (nombre completo) y con su número de carné.
- Índice
- Gramática del lenguaje de descripción, con ejemplos. Se deberá emplear diagramas de ferrocarril o expresiones regulares.
- Plan de pruebas efectuadas: cada módulo ha de ser probado por aparte, diseñando casos de prueba específicamente para cada uno. Se supone que cada módulo implementa una funcionalidad específica, que se aplica a un cierto universo de entradas, produciendo un bien definido conjunto de salidas. En ese conjunto de entradas habrá casos mínimos, casos máximos (tal vez), casos típicos y casos excepcionales; en universos más complejos se puede esperar que el universo de entradas se divida en clases (en ocasiones en varios niveles). Si esto último es el caso, entonces los casos mínimos, máximos, usuales y excepcionales ocurrirán dentro de cada subdivisión. Así pues, esta sección deberá documentar, para cada módulo:
    a. una descripción del universo de entradas
    b. los casos mínimos, máximos (si los hubiere), típicos, excepcionales
    c. entradas no pertenecientes al universo (que el módulo debe detectar y rechazar)
    d. las pruebas, esto es, los ejemplos de cada caso con sus correspondientes resultados.
- Análisis A*
    a. Función de costo de transición
    b. Función heurística de costo futuro
    c. Demostración de admisibilidad de la función de costo total estimado

Elementos de evaluación

Requisitos indispensables

1. El programa deberá compilarse in situ en el momento de la presentación
2. El programa deberá tener interfaz gráfica
3. El programa deberá ejecutar.
4. El programa deberá obtener la solución mediante búsqueda A*.
5. El programa correrá en el SO Linux

Rúbrica

Interfaz con el usuario: 25%

Rubro	***Inexistente (0%)***	***Insuficiente (25%)***	***Deficiente (50%)***	***Medianamente apropiado (75%)***	***Apropiado (100%)***
Proceso evidente (2%)	El usuario no sabe por sí mismo lo que debe hacer para hacer funcionar el programa	Hay algunas indicaciones, pero el usuario hace funcionar el proceso trabajosamente	Hay indicaciones, pero el proceso es incómodo, y no hay documentación en línea, o es confusa	Hay memorización mínima, pero la documentación no está siempre disponible	Hay memorización mínima, y en caso de duda la documentación está siempre disponible
Condescendencia con la entrada manual del estado de la torre o del estado final (3%)	Si el usuario cambia de opinión no hay manera de volver atrás		Si el usuario cambia de opinión, puede volver a un estado anterior (undo) o al estado inicial (restart), pero no a ambos		Si el usuario cambia de opinión, puede volver a un estado anterior (undo) o inclusive al estado inicial (restart)
Simplicidad de manejo (3%)	El manejo del programa es complicado		El número de pasos para completar una tarea es de mediano tamaño y no es directo		El número de pasos para completar una tarea es pequeño
Buen diálogo (3%)	No hay mensajes de parte del sistema en caso de error	No salen mensajes de error en todos los casos necesarios		En caso de errores los mensajes son poco informativos	En caso de errores los mensajes son suficientemente informativos
Cuidado del usuario (2%)	El programa no advierte nada en caso de una acción irreversible		Antes de una acción seria, el programa no siempre hace la advertencia pertinente al usuario		Antes de una acción seria, el programa hace la advertencia pertinente al usuario

Navegable (3%)	El usuario no tiene ninguna orientación sobre las rutas de ejecución ni de salida rápida.	No se provee de rutas de ejecución adecuadas	Provee de rutas de ejecución, pero no se permite una salida abrupta	Provee de adecuadas rutas de ejecución y de salida pero no permite salvar trabajo incompleto	Provee de adecuadas rutas de ejecución y de salida (permite salvar trabajo incompleto)
Buenas pistas visuales (3%)	El programa no usa iconos, ni menús pulldown		El diálogo con el programa se basa en iconos y otras pistas visuales pero son difíciles de interpretar		El diálogo con el programa se basa en iconos y otras pistas visuales fácilmente reconocibles, basadas en metáforas del mundo real
Predictivilidad (3%)	El programa se comporta de manera impredecible para el usuario		Hay consistencia en la ejecución de algunas partes del programa, pero no en todo él.		Hay consistencia en la ejecución del programa
Flexibilidad de entrada (3%)	La entrada no es flexible: no existe la opción para entrada por archivo		El usuario puede elegir qué tipo de entrada quiere, pero no puede elegir la ubicación de los archivos		El usuario puede elegir qué tipo de entrada quiere, y en el caso de los archivos, puede describir fácilmente su ubicación

Ejecución: 60%

Rubro	***Inexistente (0%)***	***Insuficiente (25%)***	***Deficiente (50%)***	***Medianamente apropiado (75%)***	***Apropiado (100%)***
Ejecución (15%)	No brinda soluciones correctas		Ejecuta correctamente, pero solamente para casos típicos		Ejecuta correctamente y sin errores todas las configuraciones dadas
Compilación de entrada (10%)	No reconoce ningún tipo de hilera de entrada		Existen hileras que no son reconocidas por el parser (falso rechazo)		Hace parsing correcto de casos normales de la entrada por archivo
Detección de entradas incorrectas (10%)	No se da cuenta de las hileras que no son gramaticales		Da por buenas algunas hileras incorrectas (falsa aceptación)		Detecta errores en la entrada por archivo
Muestra salidas correctas (10%)	No hay salida por archivo de texto	Hay archivo de salida, pero no contiene la solución	Hay archivo de salida, contiene la solución, pero no contiene los		Brinda una salida correcta y completa por archivo de

			estados inicial o meta		texto para la solución
Implementación de la teoría (15%)		No se implementa apropiadamente la función heurística de costo futuro	No se implementa apropiadamente la función heurística de costo total		Se implementa un uso apropiado de A*

Documentación: 15%

Rubro	***Inexistente (0%)***	***Insuficiente (25%)***	***Deficiente (50%)***	***Medianamente apropiado (75%)***	***Apropiado (100%)***
Formalidades (3%)	El documento presentado no cumple ninguna de las formalidades solicitadas	El documento cumple solo algunas de las formalidades	El documento cumple la mayoría de las formalidades		El documento cumple con todas las formalidades exigidas
Documentación del parser (4%)	No hay documentación para el parser	Los diagramas de ferrocarril o las Reg.Exp. son incorrectas	El diagrama de ferrocarril es correcto pero no hay ejemplos		El diagrama de ferrocarril y los ejemplos son apropiados
Documentación de A (4%)*	No hay documentación de los componentes de A*	No figura la definición de función heurística de costo futuro	No figura la definición de costo de transición		La función de costo puntual y la función de costo total estimado son apropiadas
Demostración de admisibilidad (4%)	No se presentó ninguna demostración de admisibilidad	Se brinda una justificación vaga para la admisibilidad	La justificación de admisibilidad es más o menos precisa, pero no es formal		La demostración de admisibilidad es matemáticamente correcta

Formalidades de elaboración y ejecución

Número de integrantes de los grupos: hasta un máximo de 4
Fecha de revisión: 14 de septiembre de 2018