Introducción a la Inteligencia Artificial
Proyecto Aplicativo 2

Profesor:
Ing. Julio Omar Palacio Niño, M.Sc.
palacio_julio@javeriana.edu.co

Temática:
Problema de satisfacción de restricciones.

Integrantes:
Grupos máximo de 4 personas

Problema
El objetivo del siguiente proyecto es la construcción de un programa que permita encontrar la solución de un sudoku, para lo cual se emplearía las estrategias de modelamiento de un problema de satisfacción de restricciones (CSP).

En el proyecto anterior de problemas de búsqueda del laberinto el objetivo era encontrar una secuencia de acciones para alcanzar un objetivo predefinido, donde la secuencia (camino) era importante. Por el contrario, en este proyecto de CSP necesitamos encontrar un objetivo (estado del mundo que satisfaga las restricciones), pero el camino hacia la solución no es importante.

Sudoku
El objetivo es rellenar una cuadrícula de n x n celdas dividida en subcuadrículas de n/3 x n/3 con las cifras del 1 al 9 partiendo de algunos números ya dispuestos en algunas de las celdas. No se debe repetir ninguna cifra en una misma fila, columna o subcuadrícula. Un sudoku está bien planteado si la solución es única.

Lectura
Para la lectura del sudoku se ha de implementado por medio de una matriz en Python de tamaño n x n , los espacios son 0 son espacios vacíos

0	0	3	0	2	0	6	0	0
9	0	0	3	0	5	0	0	1
0	0	1	8	0	6	4	0	0
0	0	8	1	0	2	9	0	0
7	0	0	0	0	0	0	0	8
0	0	6	7	0	8	2	0	0
0	0	2	6	0	9	5	0	0
8	0	0	2	0	3	0	0	9
0	0	5	0	1	0	3	0	0

```
tablero=[
[0,0,3,0,2,0,6,0,0],
[9,0,0,3,0,5,0,0,1],
[0,0,1,8,0,6,4,0,0],
[0,0,8,1,0,2,9,0,0],
[7,0,0,0,0,0,0,0,8],
[0,0,6,7,0,8,2,0,0],
[0,0,2,6,0,9,5,0,0],
[8,0,0,2,0,3,0,0,9],
[0,0,5,0,1,0,3,0,0]]
```

Solución 1

Uno de los primeros enfoques para solucionar un sudoku es la implementación de fuerza bruta, ingresando números y probando todas las posibles combinaciones, dadas las reglas del juego.

```
def solve_sudoku_FB(tablero):
   # Construccion de la función que dado un tablero devuelva verdadero o falso si
encuentra o no una solución al tablero entregado
   #en caso de ser verdadero la solución queda almacenada en "tablero"

    return respuesta  # respuesta puede ser True o False
```

Solución 2

Dado que probar todas las posibles combinaciones de números es un método muy ineficiente, es mejor afrontar un método basado en satisfacción de restricciones, y el primero de ellos es el uso del algoritmo *backtrakking*, el algoritmo es el siguiente

```
función backtracking(problema):
    si el problema está resuelto:
        devolver la solución
    para cada opción en las posibles opciones del problema:
        si la opción es válida:
            aplicar la opción al problema
            resultado = backtracking(problema actualizado)
            si resultado es válido:
                devolver resultado
            deshacer la opción aplicada al problema
    devolver "No hay solución"
```

Solución 3

A pesar que el backtrakking puede ofrecer un método efectivo para encontrar una solución a partir de un proceso sistemático y ordenado basado en las restricciones del problema, en ciertas ocasiones es necesario prevenir la generación de soluciones que serán invalidas en el futuro. Este proceso se realiza analizando las restriciones y el dominio del problema. Por lo cual son muy útiles para mejorar la capacidad de solución del problema.

```
1. Inicializar una estructura de datos para representar el problema
2. Inicializar una lista de variables no asignadas
3. Mientras haya variables no asignadas:
   a. Seleccionar una variable no asignada.
   b. Para cada valor en el dominio de la variable seleccionada:
      i. Asignar el valor a la variable.
      ii. Realizar comprobaciones de consistencia hacia adelante:
          - Para cada variable vecina:
            * Eliminar el valor asignado del dominio de las variables vecinas si
viola las restricciones.
      iii. Si ningún dominio se vacía, continuar con la siguiente variable no
asignada.
      iv. Si todos los dominios son no vacíos, continuar con la siguiente variable
no asignada.
   c. Si no se pudo asignar ningún valor a la variable, retroceder (backtrack).
4. Si se han asignado valores a todas las variables sin violar las restricciones,
se ha encontrado una solución.
5. Si se ha recorrido todo el espacio de búsqueda sin encontrar una solución, no
hay solución.
```

Proyecto

El código del proyecto deberá estar totalmente desarrollado en Python para lo cual deberá estar compuesto en los siguientes componentes

- lectura del tablero
- función solucionador fuerza bruta
- función solución backtrakking basico
- función backtrakking con comprobación hacia delante
- impresión del tablero solución

Para la construcción del proyecto se entregará como base un código en Python donde deberá agregar lo necesario para el funcionamiento adecuado del proyecto.

Análisis

Debido a que se tienen tres estrategias para solucionar el problema que conclusiones se pueden obtener, cual rendirá más, que dificultades se presentaron, además que otras estrategias podrían ser implementadas para encontrar una solución, recuérdese que el problema de satisfacción de restricciones es un campo amplio, además se pueden emplear otras técnicas para encontrar una solución al problema.

Entregables

- Documento tipo informe con el desarrollo del problema, como afrontaron cada uno de los desafíos
- Código fuente debidamente comentariado
- Todo se entregará en campus virtual

Sustentación

- Se procederá a realizar una pequeña demostración del funcionamiento del proyecto al resto de la clase en una fecha designada en clase