Proyecto 2: Nonograma

Descripción general

En la primera parte del proyecto desarrollamos una aplicación web del juego lógico nonograma.
En esta segunda parte, agregamos la posibilidad que el jugador pueda revelar una celda y revelar todo el tablero resuelto. La generación de la solución del juego la realizamos completamente en Prolog.

Parte I: Prolog

Estrategia de resolución:
La estrategia que utilizamos consiste en generar todas las soluciones posibles para cada una de las filas según sus pistas (e.g. si las pistas de la fila 3 con longitud 5 son [1, 2] se genera una lista de soluciones

- [#, _, #, #, _]
- [#, _, _, #, #]
- [_, #, _, #, #])

Cuando ya tenemos esta lista para cada una de las filas, construimos un tablero generando configuraciones de una solución de entre todas por cada fila. Dejamos que Prolog pruebe todas las combinaciones de filas necesarias para encontrar una que también satisface las pistas de las columnas. Cuando se cumpla esta condición se encontró un tablero válido que es solución al conjunto de pistas.

Si el algoritmo encuentra múltiples soluciones dadas las pistas (por ejemplo, si las pistas son [[2], [3]] y [[1], [1], [1], [1], [1]] se encuentran dos soluciones), entonces se toma la primera encontrada.

Predicados principales:

- **generarSolucionDeTodasLasFilas/3:** Dadas las pistas de las filas y la longitud de las mismas, genera todas las posibles soluciones de cada fila (llamando a otro predicado) y las acumula para luego devolver una lista con las mismas.
- **generarSolucionesDeUnaFila/3**: Para una fila, dada su longitud y sus pistas, encuentra todas las soluciones que conforman los grupos disjuntos de # especificado por sus pistas. Se usa findall/3 sobre el predicado

validarEspaciado/2 para dar con todas las filas que respetan la configuración de grupos y tienen uno o más espacios blancos entre ellos.
- **grupos/2:** Dadas las pistas de una fila, genera los grupos de #.
- **testearCombinaciones/3:** Dadas todas las soluciones encontradas anteriormente se busca en el espacio de posibles tableros los que además satisfacen todas las pistas de las columnas, generando y testeando de a un tablero. Esta búsqueda se basa en el uso del predicado configuracion/2. El testeo de tableros se hace rotando cada tablero 90° (con transpose/2), y verificando las pistas de las columnas con testearSolucion/2, el cual usa predicados de la parte anterior del proyecto.
- **configuracion/2**: Dada una lista de listas de soluciones genera un tablero tomando de a un elemento de cada lista con member/2 y agregandolos a una lista de filas que constituye el tablero

---

Parte II: React

Manual de usuario:
La interfaz consiste de dos partes interactivas: el selector de modo # o X, y el tablero de juego.
Cuando se está en el modo # y se hace click en un recuadro del tablero, si ese recuadro contiene # se vacía el recuadro. Si ese recuadro está vacío o contiene X, se pinta con #.
Además se incluyen dos botones, uno para revelar celda (el cual muestra la solución correcta para dicha celda) y otro para revelar el tablero (el cual muestra la solución de todo el juego). Para salir de cada modo revelar, se deberá volver a seleccionar el botón correspondiente.

Las pistas de las filas y columnas se activan cuando se satisface esa fila y columna, y el juego termina cuando el jugador satisface todas las filas y columnas a la vez. Cuando se gana el juego se desactiva la interacción con el tablero y el selector de modo.

Componentes:
Game.js:
- State: A los elementos descritos en el primer informe, agregamos los siguientes:
    - grillaEnJuego: mantiene la grilla actual del jugador mientras está revelando el tablero para que posteriormente vuelva a ver su propia grilla.
    - grillaResuelta: mantiene la grilla con la solución.

    - estadoRevelandoCelda: mantiene si actualmente se está revelando celdas.
    - estadoRevelandoTablero: mantiene si actualmente se está revelando el tablero.
    - filasActual y colsActual: análogo a grillaEnJuego para las líneas correctas.

- Funciones: A las funciones descritas en el primer informe, agregamos las siguientes:
    - obtenerSolucion: lógica para obtener la solución de la grilla. Se comunica con Prolog a través del predicado solve/3 enviando la información de las pistas de las filas y columnas.

      ```
      queryS = `solve(${filas}, ${columnas}, Solucion)`
      ```

      Solve/3 devuelve la grilla correcta y luego se modifica el estado de grillaResuelta.
    - revelarTablero: lógica del botón de revelar tablero.
    - revelarCelda: lógica del botón de revelar celda.