Grado en Ingeniería Informática, Universidad de Córdoba

Sistemas Inteligentes

Práctica 1: N-Reinas

Aurora Esteban Toscano
aestebant@uco.es

22 de febrero de 2022

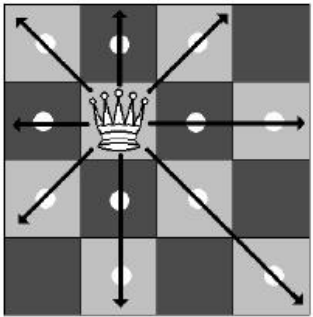
Objetivo

Poner en práctica las técnicas de búsqueda vistas en teoría sobre un problema clásico: **el puzzle de las N-Reinas**.

- Entender conceptualmente el problema de las N-Reinas.
- Entender el modelo de representación de estados que se proporciona.
- Implementar una técnica de búsqueda para resolver el problema.
- Utilizar la solución programada para responder al cuestionario **al final de clase**.

Fundamentos del problema I

El problema de las N-Reinas consiste en colocar n reinas en un tablero de dimensiones $n \times n$ de forma que no se amenace ninguna a otra $\rightarrow$ que no compartan fila, columna o diagonal.

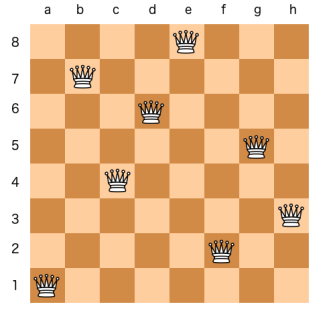


Posibles ataques de una reina en ajedrez

Fundamentos del problema II

El caso específico de las *8 reinas* fue propuesto por el ajedrecista alemán Max Bezzel en 1848. Hoy en día se usa para explicar diversas técnicas de computación.

- Definido para todos los números naturales.
- Los casos de $n = 2$ y $n = 3$ no tienen solución.
- Se sabe el número completo de soluciones hasta $n = 27$.
- Los algoritmos de búsqueda por fuerza bruta pueden manejar variantes hasta $n \approx 20$.
- Más allá, el coste computacional crece demasiado.



Una solución para $n = 8$

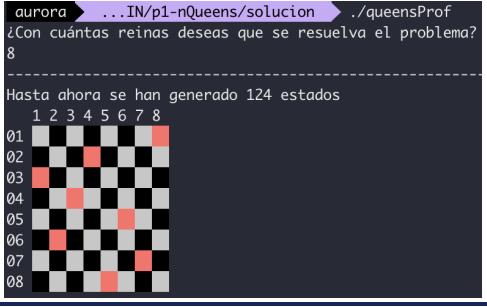
Material de la práctica

Para resolver el problema se proporciona una librería que funciona *prácticamente*. Será necesario **hacer uso de las funciones que se proporcionan** para programar la técnica de búsqueda apropiada.

- `main.cpp`: fichero con la estructura principal del código. Tiene instrucciones para completar las tareas de programación.
- `State.h`: declaración de funciones genéricas para cualquier estado. En la solución deben usarse `isObjective()` y `getSucessors()`
- `QueensState-XXX.o`: librería pre-compilada con el código de funcionamiento interno de la librería. Dependiente de la arquitectura en la que se esté realizando la práctica.

Compilación y ejecución

1. Compilar el programa según el SO:
   - Ubuntu o equivalente: `g++ -o queensProf main.cpp QueensState-ubuntu.o`
   - MacOS: `g++ -o queensProf main.cpp QueensState-macOS.o`
   - LinuxUCO: `g++ -o queensProf main.cpp QueensState-aulasUCO.o`
2. Ejecutar el archivo creado: `./queensProf`.
   - Recordar darle los permisos adecuados: `chmod u+x queensProf`



Tareas a implementar

- Haciendo uso de la clase `stack` de C++:
  1. Crear la estructura frontera.
  2. Insertar el estado inicial.
  3. Crear el bucle de búsqueda.
  4. Obtener el estado de frontera.
  5. Comprobar si es solución.
  6. Si no lo es, generar los sucesores e insertarlos en la frontera.
  7. Borrar el vector de sucesores de la memoria.
- Adicionalmente, tratar de cambiar el programa para implementar la frontera con una *cola* en lugar de una pila.

Evaluación

En los últimos 10 minutos de clase se realizará un cuestionario de cuatro preguntas en Moodle para evaluar la práctica.
Se necesitará ejecutar tanto el programa basado en pila como el basado en cola.

- Se recomienda conservar ambos ejecutables con distintos nombres para ejecutarlos rápidamente con las condiciones que se pidan.

Grado en Ingeniería Informática, Universidad de Córdoba

Sistemas Inteligentes

Práctica 1: N-Reinas

Aurora Esteban Toscano
aestebant@uco.es

22 de febrero de 2022