Universidad de Málaga

MEMORIA ROBOCODE

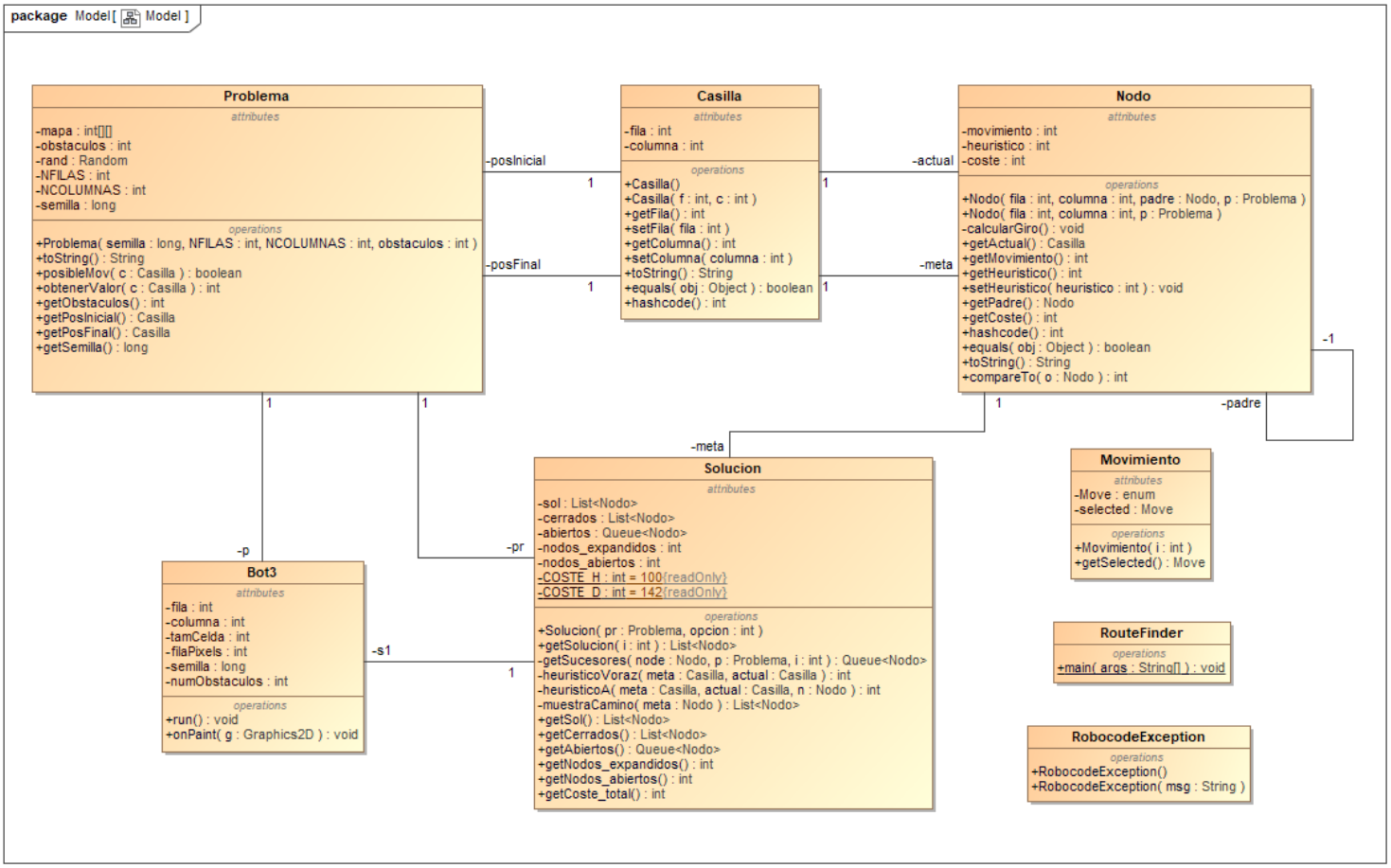
Sistemas Inteligentes

Profesor: José Antonio Montenegro Montes

Víctor Ramírez Mármol

Antonio Lara Gutiérrez

2º Ingeniería Informática A

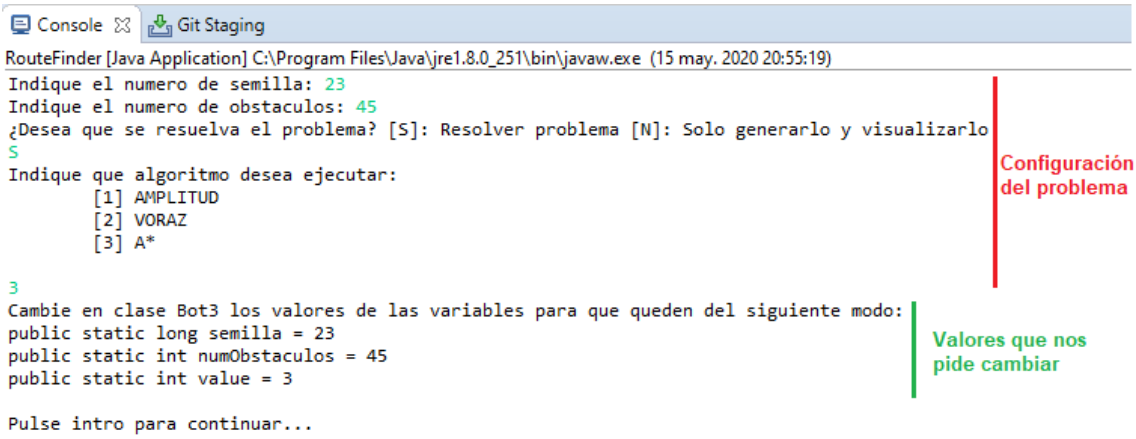


Aquí presentamos el diagrama de clases correspondiente a nuestra implementación del Robocode. Vamos a enumerar las clases con sus principales métodos:

- Clase Problema: El constructor crea un problema que viene representado por una matriz bidimensional. Asigna tantos obstáculos como se requieran en casillas aleatorias, así como la posición inicial y la posición final a la que se aspira llegar. El método *posibleMov(Casilla c)* verifica que una casilla del problema está libre para ser visitada por nuestro agente.
- Clase Casilla: representa una casilla, que tiene una fila y una columna.
- Clase Nodo: almacena información para la búsqueda, esto es, el heurístico, el coste acumulado y el movimiento que realiza con respecto a su padre. El constructor calcula el coste acumulado y asigna el correspondiente padre. El método *calcularGiro()* asigna un número al nodo que corresponde con un Movimiento
- Clase Movimiento: aloja el tipo enumerado Move con el literal de todos los movimientos posibles (hasta 8) y un método para obtenerlo
- Clase RobocodeException: manejador específico de excepciones
- Clase RouteFinder: clase principal, crea el tablero y ejecuta las implementaciones
- Clase Bot3: modelo de Robot que ejecuta la solución. En su método *run()* crea una versión del problema y su respectiva solución para proceder a ejecutarla visualmente. En su método *onPaint()* incluye información importante en pantalla: nodos cerrados, abiertos, solución, nodo inicial y final…
- Clase Solución: clase que administra la solución del Problema. El método *getSolucion()* implementa un algoritmo genérico que se modifica según el algoritmo especificado. Este método se complementa con *getSucesores()* que devuelve todos los posibles sucesores dado un nodo (verificando que cumplan todas las restricciones). Los métodos heurísticos calculan y asignan el heurístico requerido a cada nodo (diferente para A* y Voraz). Por último, el método *muestraCamino()* devuelve el camino que ha seguido el robot hasta la meta.

Ejecución de los algoritmos

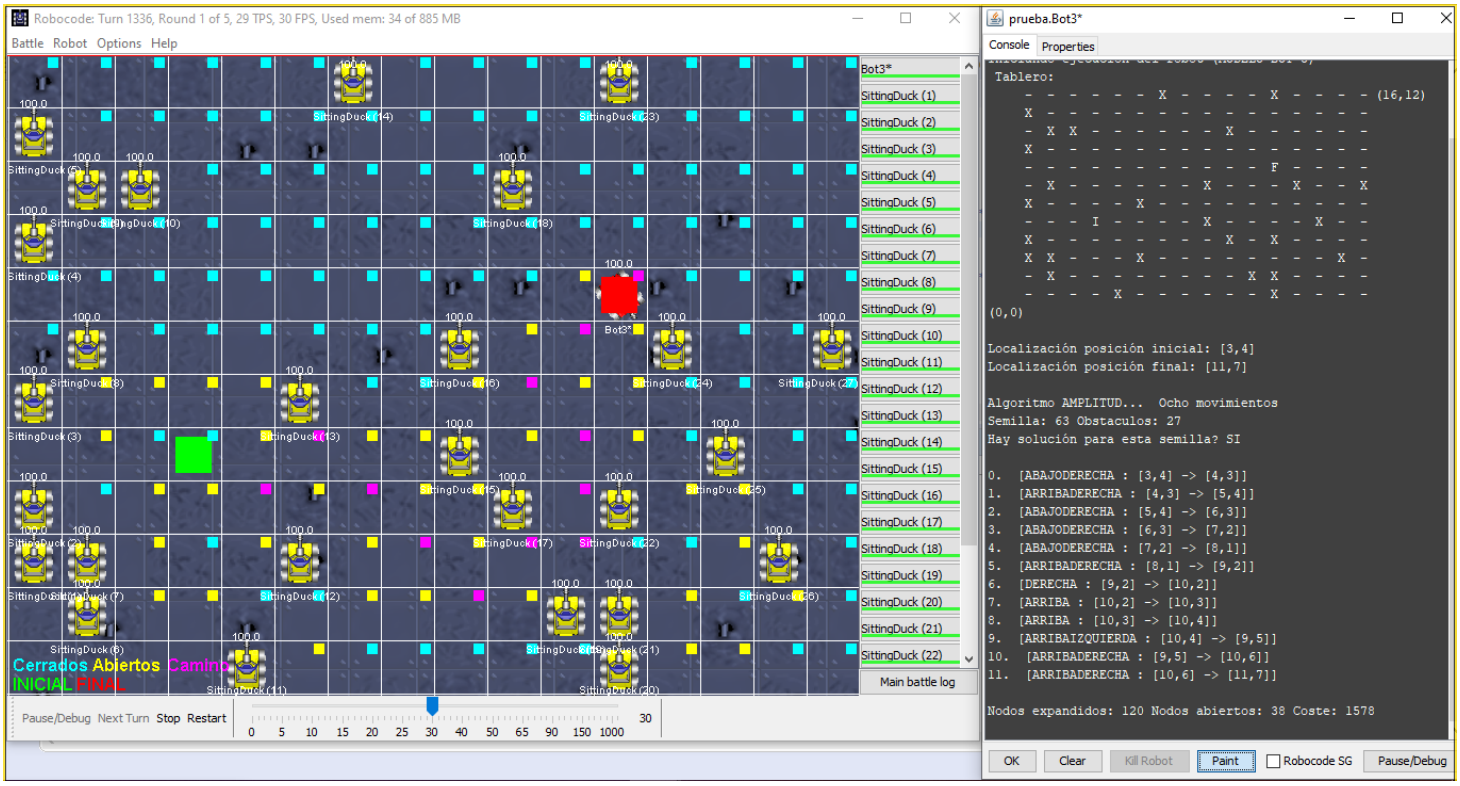
Para ejecutar cualquier algoritmo se debe lanzar la clase RouteFinder. En la consola aparecerá un configurador para seleccionar los diferentes parámetros a la hora de ejecutar el algoritmo. Entre ellos, el número de obstáculos y la semilla a usar, si se desea ejecutar el hito2 (creación del mapa) o el hito3 (creación y solución), así como el tipo de algoritmo que se quiere seguir en la solución. Además, la pantalla informa de los cambios que se tienen que hacer en la clase Bot3/Bot2 para su correcto lanzamiento. En esta captura podemos ver una configuración válida del problema.



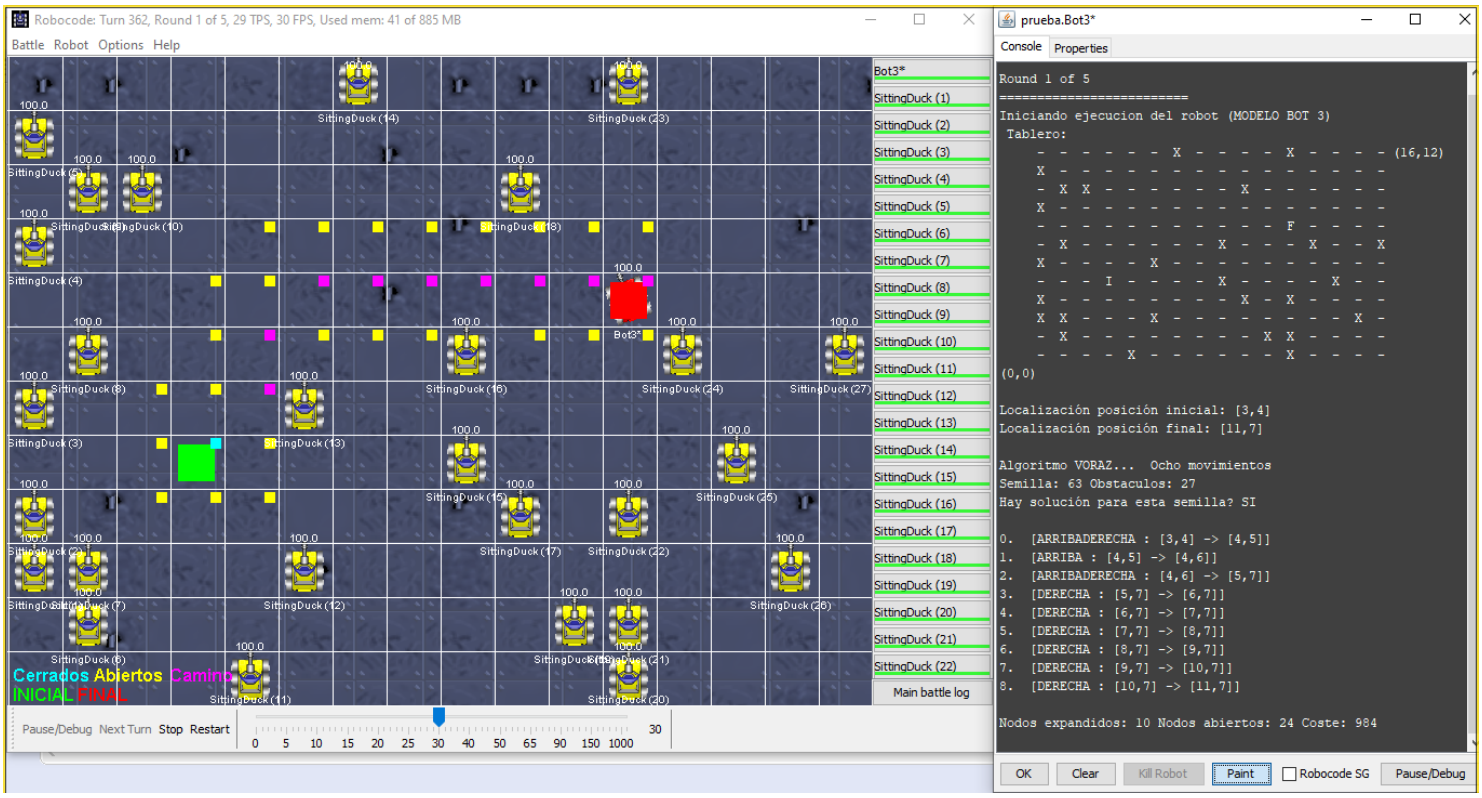
Ejemplo funcionamiento con todos los algoritmos

Vamos a mostrar las ejecuciones de un problema cuya semilla es 63, con un total de 27 obstáculos. Hemos escogido esta configuración en particular porque todos los algoritmos siguen caminos diferentes, por lo que podemos compararlos

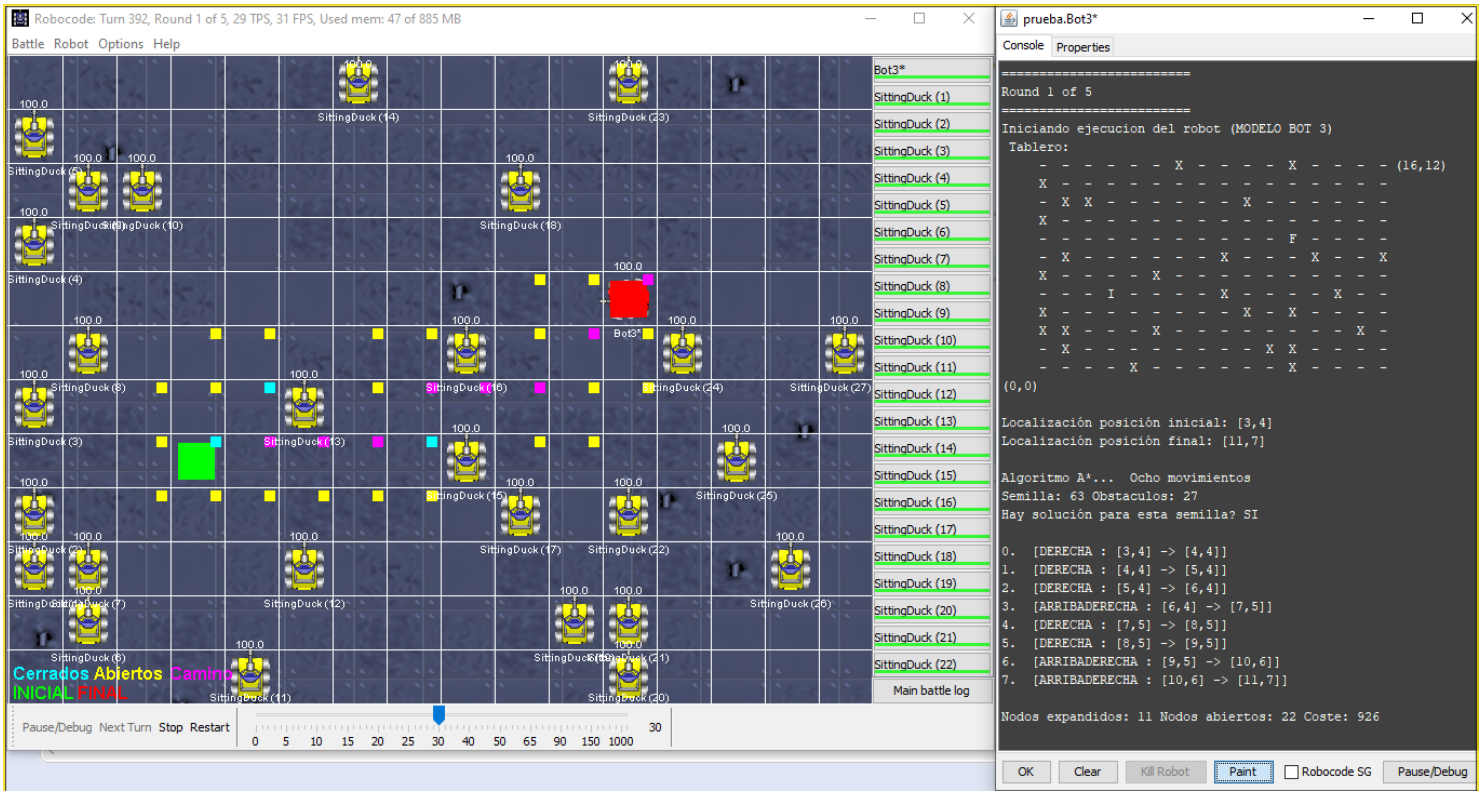
- **<u>Ejecución algoritmo en amplitud</u>**



- **Ejecución algoritmo voraz**



- **Ejecución algoritmo A***



En esta tabla y gráfica podemos ver la comparación de todas las implementaciones del experimento:

Implementación	Nodos expandidos	Nodos abiertos	Coste
Amplitud	120	38	1578
Voraz	10	24	984
A*	11	22	926

Conclusiones

En general, a ambos nos ha parecido una práctica muy interesante en cuanto a contenido, acercándonos más a la ejecución de los algoritmos de búsqueda, así como a su implementación en código. Hemos afianzado los conceptos sobre búsqueda informada y no informada, y hemos aprendido a planificar y diseñar estructuras de datos para llevar a cabo estas tareas.

Para terminar, solo hemos encontrado algunas pequeñas dificultades en cuanto a implementación, pero que hemos sabido solventar y adaptar, creemos que, correctamente.

Nota. El código de la práctica también se encuentra disponible en este proyecto **git**