

**SISTEMAS INTELIGENTES.**

**PRÁCTICA DE BÚSQUEDA 2**

**1.** Crear tres heurísticas admisibles para la aplicación de cálculo de rutas (*RouteApp*). Para ello, se deberá implementar el método *h* de la clase *RouteHeuristicFunc en RouteFunctionFactory.java*.

a. Heurística Distancia Euclídea.

**private** **static** **class** RouteHeuristicFunc **implements** HeuristicFunction

{

@Override

**public** **double** h(Object state)

{

RouteBoard board= (RouteBoard) state;

//h(n) = sqrt((n.x-goal.x)^2 + (n.y-goal.y)^2)

**double** h = Math.*sqrt*(Math.*pow*((board.CurrentX-board.GoalX), 2) +

Math.*pow*(board.CurrentY-board.GoalY, 2));

**return** h;

}

}

b. Heurística Distancia Manhattan

**private** **static** **class** RouteMHeuristicFunc **implements** HeuristicFunction

{

@Override

**public** **double** h(Object state)

{

RouteBoard board= (RouteBoard) state;

//m(n) = (abs(n.x-goal.x) + abs(n.y-goal.y)).

**double** m = Math.*abs*((board.CurrentX-board.GoalX)) +

Math.*abs*(board.CurrentY-board.GoalY);

**return** m;

}

}

c. Heurística Distancia Diagonal

**private** **static** **class** RouteDHeuristicFunc **implements** HeuristicFunction

{

@Override

**public** **double** h(Object state)

{

RouteBoard board= (RouteBoard) state;

// d(n) = max(abs(n.x-goal.x), abs(n.y-goal.y))

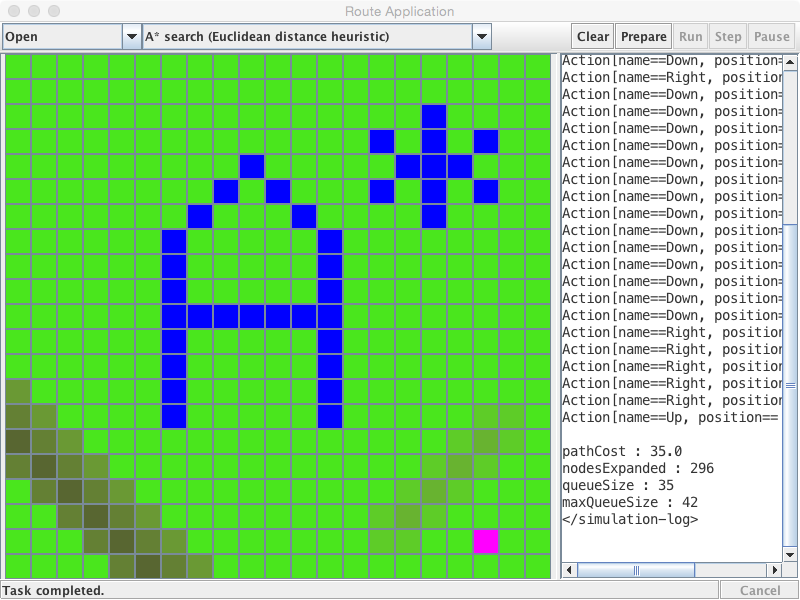
**double** d = Math.*max*(Math.*abs*(board.CurrentX-board.GoalX),

Math.*abs*(board.CurrentY-board.GoalY));

**return** d;

}

}

2. El despegable de los algoritmos solo contiene una entrada para el algoritmo A\* con la distancia Euclídea. Para que las demás heurísticas puedan ser ejecutadas, es necesario crear dos entradas más en el despegable.

2.1. En la clase RouteApp añada dos entradas

* addSearchAlgorithm("A\* search (Manhattan heuristic)",new AStarSearch(new GraphSearch(), RouteFunctionFactory.getMHeuristicFunction()));
* addSearchAlgorithm("A\* search (Distancia Diagonal heuristic)",new AStarSearch(new GraphSearch(), RouteFunctionFactory.getDHeuristicFunction()));

2.2. Implemente los métodos:

RouteFunctionFactory.getMHeuristicFunction()));

**public** **static** HeuristicFunction getMHeuristicFunction ()

{ *\_heuristicFunction* = **new** RouteMHeuristicFunc();

**return** *\_heuristicFunction*

}

RouteFunctionFactory.getDHeuristicFunction()));

**public** **static** HeuristicFunction getDHeuristicFunction ()

{ *\_heuristicFunction* = **new** RouteDHeuristicFunc();

**return** *\_heuristicFunction*;

}

3. Anotar en la tabla los resultados de ejecutar los algoritmos con el mapa *mapa.dat.* Anota también las conclusiones que extraes de los resultados.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Secuencia Acciones** | **Coste** | **Nodos Expandidos** | **Nodos Frontera Máx** |
| **A\* H\_E** | Down, Right, Right, Right, Right, Right, Right, Right, Right, Down, Right, Down, Right, Down, Right, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Right, Right, Right, Right, Right, Up | 35 | 296 | 42 |
| **A\* H\_M** | Down, Down, Down, Down, Down, Down, Right, Down, Down, Down, Down, Right, Right, Down, Down, Down, Right, Right, Right, Right, Right, Right, Down, Right, Right, Down, Down, Down, Right, Right, Right, Right, Right | 35 | 227 | 65 |
| **A\* H\_D** | Down, Right, Right, Right, Right, Right, Right, Right, Right, Down, Right, Down, Right, Down, Right, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Right, Right, Right, Right, Up, Right | 35 | 319 | 39 |
| **Amplitud** | Right, Right, Right, Right, Right, Right, Right, Right, Right, Right, Right, Down, Down, Down, Down, Right, Right, Down, Right, Right, Right, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down | 37 | 373 | 23 |
| **Profundidad** | Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Down, Right, Right, Right, Right, Right, Right, Right, Right, Right, Right, Right, Right, Right, Right, Right, Right, Up | 67 | 332 | 138 |

**Conclusiones**

Las mejores heurísticas según su coste son la distancia Euclídea, la distancia Manhattan y la distancia Diagonal, ya que esas tienen un coste de 35, el menor de todos.

Si las comparamos según el número de nodos expandidos la mejor sería la heurística distancia Manhattan, ya que con 227 es la que menos nodos expande.

La mejor según los nodos frontera máximos es la amplitud, porque tiene el menor número , el cual es 23.