**Sistemas Inteligentes. 2020-21**

**Evaluación Práctica-1 Grupo 3-LB1**

**Nombre:** David Arnal García

*Al realizar esta prueba de evaluación ACEPTO la "cláusula de veracidad" por la que no recibiré ni daré ayuda en esta prueba y garantizo la autoría del 100% de los resultados.*

*El incumplimiento por mi parte de los deberes derivados de las buenas prácticas de honestidad académica podrá dar lugar a la adopción de las medidas contenidas en la Normativa de Integridad Académica del alumnado de la UPV.*

**Importante:**

* Debe subirse a PoliformaT el código inicial realizado en la práctica
* Contestad a las preguntas en los recuadros correspondientes.
* Al final del examen, subid el código modificado.

**Tiempo: 1h:05’. La Tarea se cerrará automáticamente a las 16:15.**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **(Tiempo estimado 15’, 2 puntos) Considerando el problema propuesto,**
2. Modificad la regla-meta para que imprima también el número de disparos que quedan.

(defrule goal

(declare(salience 10))

(pos ?x ?y box enemy $?bb shots ?z level ?l)

=>

(printout t "Solucion Encontrada en el Nivel: " ?l ",

Numero de Nodos expandidos: "?\*nod-gen\* crlf)

(printout t "Los disparos de que quedan son:" ?z crlf)

(halt)

)

b) Suponed la siguiente situación inicial:

El tamaño del grid es de (x,y): 13 columnas y 4 niveles.

Las cajas están en las posiciones (x,y): (2,1) (12,3) (13, 2) (7, 2)

Hay huecos en las posiciones (x,y): (4, 2) (3, 3), (6, 3) (5 4)

Hay enemigos en las posiciones (x,y): (4,2) (8,2) (5,2) (6, 2)

Hay escaleras en las posiciones (x,y): (2,3) (2, 2) (1, 3) (10, 2) (7,1) (11, 3)

Se disponen de 4 disparos iniciales

Inicialmente, el robot está en la posición (7,2)

|  |
| --- |
| **Indicad la nueva expresión deffacts correspondiente a esta situación inicial:**  (deffunction start()  (reset)  (printout t "Introduce la Profundidad Maxima:= ")  (bind ?prof (read))  (printout t "Introduce el Tipo de Busqueda " crlf " 1.- Anchura" crlf " 2.- Profundidad" crlf)  (bind ?a (read))  (if (= ?a 1)  then (set-strategy breadth)  else (set-strategy depth))  (printout t " Ejecuta run para poner en marcha el programa " crlf)  (assert (pos 7 2 box b 2 1 b 12 3 b 13 2 b 7 2 enemy e 4 2 e 8 2 e 5 2 e 6 2 shots 4 level 0))  (assert (ladder l 2 3 l 2 2 l 1 3 l 10 2 l 7 1 l 11 3))  (assert (gap g 4 2 g 3 3 g 6 3 g 5 4))  (assert (tam 13 4))  (assert (max-level ?prof))  ) |

Lanzad el programa y obtened el nivel en que se encuentra la solución, número de nodos generados, disparos restantes y posición final del robot, para una búsqueda en profundidad y en anchura:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Nivel en que se encuentra la solución: | Nodos generados: | Disparos Restantes |
| Anchura | 29 | 21578 | 3 |
| Profundidad **(máx. prof= 40)** | 39 | 17878 | 3 |

1. **(Tiempo estimado 10’, 2 puntos)** Modificad o añadid el código necesario para que, en la situación final, además de las condiciones previas, el robot deba haber matado a todos los enemigos.

|  |
| --- |
| (defrule goal  (declare(salience 10))  (pos ?x ?y box enemy shots ?z level ?l)    =>  (printout t "Solucion Encontrada en el Nivel: " ?l ",    Numero de Nodos expandidos: "?\*nod-gen\* crlf)  (printout t "Los disparos de que quedan son:" ?z crlf)  (halt)  ) |

1. **(Tiempo estimado 15’, 3 puntos)** Añadid una nueva regla para que el robot pueda ‘saltar’ un hueco que se encuentra a su derecha. Es decir, si el robot está en la posición (x, y) puede ‘saltar’ a la posición (x+2, y), siempre y cuando haya un hueco a su derecha y no haya hueco ni enemigo en la posición destino.

|  |
| --- |
| (defrule right\_jump  (pos ?x ?y box $?aa enemy $?bb shots ?z level ?l)  (gap $?gg)  (tam ?x1 ?y1)  (max-level ?max)  (test (< ?l ?max))  (test (<= (+ ?x 2) ?x1))  (test (member$ (create$ g (+ ?x 1) ?y) $?gg))  (test (not (member$ (create$ e (+ ?x 2) ?y) $?bb)))  (test (not (member$ (create$ g (+ ?x 2) ?y) $?gg)))  =>  (bind ?nod-gen (+ ?nod-gen 1))  (assert (pos (+ ?x 2) ?y box $?aa enemy $?bb shots ?z level (+ ?l 1)))  ) |

1. **(Tiempo estimado 5’, 1 puntos)** Con las modificaciones realizadas en los puntos 2 y 3, y el estado inicial descrito en el punto 1, lanzad el programa y obtened el nivel en que se encuentra la solución y número de nodos generados, para una búsqueda en profundidad y en anchura:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Nivel en que se encuentra la solución: | Nodos generados: | Disparos Restantes |
| Anchura | 36 | 36762 | 0 |
| Profundidad **(máx. prof = 40)** | 40 | 31027 | 0 |

1. **(Tiempo estimado 10’, 2 puntos)** Responder **breve, pero razonadamente, a las siguientes preguntas:**

¿Es posible que en profundidad encuentre alguna vez una solución más corta que en anchura?

En profundidad no encontrará una solución más corta, ya que, en anchura, por definición, siempre se encontrará la solución más corta.

¿Es posible que, en profundidad, con la profundidad limitada al nivel de la solución obtenido en anchura, genere alguna vez más nodos que en anchura?

No, no es posible, ya que por anchura se recorrerán todos los nodos del nivel y, por tanto, en anchura siempre serán más nodos generados.

¿Puede ser mayor el número de reglas disparadas que el número de hechos que aparecen en la ventana de hechos? ¿Y menor? ¿Por qué?

Para un hecho puede haber distintas reglas que hagan pattern-matching, pero se podría dar el caso que un hecho no hiciera pattern-matching con ninguna regla, por lo que se podría dar el caso que no se hubiera ninguna regla disparada, pero sí varios hechos. También se puede dar el caso contrario, en el que un hecho haga pattern-matching con distintas reglas y, entonces, se ejecuten más reglas que los hechos que haya, pero teniendo en cuenta que las reglas instanciadas solo se pueden ejecutar una vez.