Uso de animaciones para la enseñanza de algoritmos de búsqueda en Inteligencia Artificial*

L. Mandow, F. Villalba, J. Coego
Departamento de Lenguajes y Ciencias de la Computación
ETSI Informática
Universidad de Málaga
{lawrence,villalba,jcoego}@lcc.uma.es

Resumen

En este trabajo se describe el uso de una herramienta gráfica de libre disposición y código abierto desarrollada como apoyo en la enseñanza de algoritmos de búsqueda heurística en Inteligencia Artificial (A*, IDA* y búsqueda frontera). Se pretende facilitar la comprensión de los mismos y sus propiedades desde una perspectiva activa y participativa por parte de los alumnos, incrementando sus oportunidades para la exploración y el descubrimiento. La herramienta permite editar cuadrículas con obstáculos y definir nodos de inicio y fin. A continuación se puede visualizar paso a paso la ejecución de diversas combinaciones de algoritmos y heurísticos, así como mostrar datos relativos a su rendimiento. Esto permite desarrollar una serie de prácticas y tareas amenas para el alumnado, que a su vez evitan algunas dudas y malentendidos frecuentes en el aprendizaje de esta materia. La herramienta se emplea con éxito desde el curso académico 2007/2008 en la ETSI Informática de la Universidad de Málaga.

1. Introducción

La búsqueda heurística es la principal hipótesis de la Inteligencia Artificial (IA) simbólica para la resolución de problemas [12]. Por este motivo se trata de una materia de caracter obligatorio virtualmente en todos los curricula internacionales sobre Informática [1] y en los planes de estudio vigentes en España para la titulación de Ingeniería Informática. En la ETSI Informática de la Universidad de Málaga, esta materia se imparte en la asignatura troncal "Inteligencia Artificial e Ingeniería del Comocimiento", de la titulación de Ingeniería Informática, y en la asignatura optativa "Introducción a la Inteligencia Artificial" de la titulación de Ingeniería Técnica en Informática de Gestión. En esta universidad la continuidad de esta materia está garantizada por la existencia de dos nuevas asignaturas en los nuevos planes de estudio: "Introducción a la IA" (obligatoria), e "IA en Juegos" (optativa). Al tratarse de una materia obligatoria en todos los estudios superiores de Informática, el desarrollo de recursos y materiales didácticos para la misma es potencialmente interesante para gran cantidad de profesores y alumnos.

Esta materia se encuentra también naturalmente presente en los principales textos docentes de IA tanto nacionales como extranjeros [18][14][11]. En estos textos se aborda la descripción de diversos algoritmos de búsqueda, se ilustra su funcionamiento mediante ejemplos, y se detallan sus propiedades formales. Se trata por tanto de una materia que puede ser presentada fácilmente al alumno siguiendo un esquema de clases magistrales. Sin embargo, la metodología docente favorecida por la implantación del Espacio Europeo de Educación Superior incide también en la necesidad de atender a los aspectos del trabajo personal del alumnado, así como a una mayor participación e interacción en clase.

En este trabajo se describe fundamentalmente el uso de una herramienta gráfica desarrollada por los autores para facilitar la enseñanza y el aprendizaje de los principales algoritmos de búsqueda en grafos de la Inteligencia Artificial. Esta herramienta se emplea por una parte como elemento dinamizador de las clases, facilitando la comprensión de la materia en el laboratorio desde una perspectiva activa y participativa por parte de los alumnos. Por otra parte, la herramienta permite también el desarrollo de diversas actividades prácticas que los alumnos deben trabajar por su cuenta, incrementando las oportunidades para la exploración y el descubrimiento. Los aspectos básicos del desarrollo de esta herramienta se enmarcan en un proyecto de innovación educati-

*Trabajo parcialmente financiado por la ETSI Informática de la Universidad de Málaga, y el proyecto de innovación educativa para la convergencia en EEES de la Universidad de Málaga (PIE07/112 - convocatoria 2007/08): Herramienta software para la enseñanza de algoritmos de búsqueda.

va de la Universidad de Málaga [10].

En el planteamiento de la experiencia descrita se tuvieron en cuenta las recomendaciones vigentes en el desarrollo de animaciones para la enseñanza de algoritmos. Un buen resumen puede encontrarse en [5]. El uso de animaciones para la enseñanza de algoritmos no siempre ha gozado de evaluaciones favorables [21]. Las conclusiones de diversos estudios apuntan a que las animaciones son útiles desde el punto de vista pedagógico únicamente si se emplean en conjunción con materiales y prácticas adecuados, se apoyan en las explicaciones del profesor, o permiten una actitud participativa por parte del alumnado [8] [6]. Se tuvieron en cuenta por ello las recomendaciones de [3] sobre animación de algoritmos: elaboración de un modelo claro y sencillo, ilustración visual de las medidas de eficiencia, control de velocidad en la animación, representación uniforme, simplicidad en el control y creación de las animaciones, y facilidad para la interacción, la exploración y el descubrimiento.

El artículo está organizado de la siguiente forma. En primer lugar se discuten los conceptos fundamentales en la enseñanza de algoritmos de búsqueda, con especial atención a su enseñanza en la Universidad de Málaga. En segundo lugar se discute el uso de herramientas de simulación en la enseñanza de estos conceptos, así como algunas herramientas previas. A continuación se describen los aspectos más relevantes de la herramienta utilizada, y se aborda la elaboración de prácticas y ejercicios para evitar los principales malentendidos que surjen en el aprendizaje de esta materia. Por último se presentan algunas conclusiones y trabajos futuros.

2. Conceptos fundamentales en la enseñanza de algoritmos de búsqueda heurística

El cuadro 1 muestra el temario de la asignatura troncal *Inteligencia Artificial e Ingeniería del Conocimiento* (4º curso ETSI Informática, Universidad de Málaga). En los temas de búsqueda heurística se persigue un doble objetivo. En primer lugar, introducir a los alumnos en los conceptos y algoritmos de búsqueda heurística comúnmente encontrados en los textos docentes, fundamentalmente A* e IDA*. Por otra parte, se intentan presentar los conceptos más innovadores y de importancia demostrada que aún no se encuentran en estos textos docentes. Concretamente, en los últimos cuatro años se han introducido en el temario dos técnicas que han demostrado un importante salto cualitativo en la resolución de problemas sobre las técnicas tradicionales. Se trata del uso de *bases de datos de patrones* [2] para el cálculo de heurísticos, y las técnicas de *búsqueda frontera* [7].

1. INTRODUCCIÓN
2. LENGUAJE LISP
3. BÚSQUEDA Y RESOLUCIÓN DE PROBLEMAS
   - Espacios de estados
   - Búsquedas con árbol
   - Búsquedas con retroceso e irrevocables
4. AGENTES
   - Problemas de generación de planes
   - Problemas con adversario y juegos
   - Problemas de decisión markovianos
5. PERCEPCIÓN

Cuadro 1: Temario de la asignatura Inteligencia Artificial e Ingeniería del Conocimiento, curso 2009/2010.

Los estudios de búsqueda heurística en Inteligencia Artificial se apoyan en la idea de que muchos problemas pueden representarse simbólicamente mediante el formalismo del *espacio de estados*. Un espacio de estados se compone de una descripción simbólica capaz de representar todas las posibles situaciones del problema, así como de un conjunto de reglas o acciones, que describen cuándo puede pasarse de una situación a otra. Una instancia de problema viene definida por la descripción de una situación inicial, y una condición objetivo que debe ser cumplida por toda situación deseada. En muchos casos se establecen preferencias entre distintas soluciones, asignando un coste a cada acción, y estableciendo como óptima aquella solución que minimiza la suma del coste de sus acciones.

Puesto que un espacio de estados define implícitamente un grafo, encontrar la solución a un problema equivale a encontrar un camino en un grafo. Los algoritmos de búsqueda en grafos son por tanto de importancia fundamental en el estudio de la IA. La

Hipótesis de la Búsqueda Heurística va má allá, señalando que la búsqueda no puede ser ciega, sino que "debe haber información en el espacio del problema y [el resolutor] debe ser capaz de extraerla y utilizarla" [12]. Esta información, normalmente incompleta, imprecisa, y a veces incierta, es la que se denomina *heurística*.

Los conceptos de estimación heurística del coste, y su introducción en los algoritmos de búsqueda mediante funciones de evaluación estáticas son sin duda los conceptos clave más importantes en esta materia.

Los principales algoritmos de búsqueda heurística son A*, que puede entenderse como la versión heurística del algoritmo de coste uniforme (Dijkstra), e IDA*, que puede entenderse como la versión heurística del algoritmo de profundización progresiva, a su vez una variante del algoritmo Backtrack. La mayor parte de textos docentes inciden invariablemente en el análisis del rendimiento de los algoritmos heurísticos, y en cómo bajo determinadas condiciones el uso de heurísticos mejora los requisitos de espacio e iteraciones sobre las versiones no informadas.

Sin embargo, el estudio aislado de estas propiedades formales lleva con frecuencia a diversos malentendidos en relación al funcionamiento práctico de los algoritmos [4]. La herramienta y las prácticas descritas más adelante pretenden precisamente dar un sentido más práctico a esta materia y evitar en lo posible cualquier malentendido.

3. Consideraciones sobre la enseñanza de los conceptos fundamentales

La enseñanza de los algoritmos de búsqueda heurística se ha apoyado tradicionalmente en el uso de puzzles deslizantes como el N-puzzle [14][9], o en el análisis de determinados grafos con una representación gráfica intuitiva, como las redes de carreteras o las mallas cuadradas [15, cap. 5].

En nuestra Escuela, el estudio de los algoritmos de búsqueda combina las explicaciones teóricas y la resolución de problemas con prácticas de programación en laboratorio, y otras de realización como trabajo del alumnado. Nuestra experiencia es que la introducción de los conceptos básicos, la comparación de heurísticos básica, y las diferencias en el comportamiento práctico de los algoritmos (A*, IDA*, búsqueda frontera) pueden asimilarse mejor mediante el uso de simulaciones. Una vez que estos conceptos se han afianzado, se puede seguir avanzando mediante la programación de puzzles y problemas sencillos, sobre los que se puede probar con facilidad la eficacia del uso de bases de datos de patrones. Concretamente, se programan bases de datos de patrones para el puzzle-8, que permiten al alumnado experimentar en el laboratorio la resolución incluso de las instancias más complejas del problema [16] con recursos de tiempo y espacio más que razonables.

Existe un creciente interés en la incorporación en las asignaturas de IA de elementos relacionados con el entorno industrial y profesional en el que desarrollarán su labor los futuros titulados. En [19] se describe una interesante experiencia en la que se aprovechan herramientas de desarrollo de Google para las interfaces de las prácticas de búsqueda. En [23] se describe también un enfoque encaminado a preparar al alumnado para ser productivo de manera inmediata en la industria del desarrollo de juegos. La futura implantación de la asignatura *IA en Juegos* en la Universidad de Málaga hace especialmente interesante en nuestro caso la exploración de esta última oportunidad.

En este trabajo consideramos principalmente nuestra experiencia con el uso de simulaciones para la mejora del proceso de enseñanza/aprendizaje, aprovechando una herramienta previamente desarrollada como parte de un proyecto de innovación educativa de la Universidad de Málaga [10]. El modelo empleado para la simulación es el uso de laberintos en mallas cuadradas.

El uso de este modelo para la simulación de algoritmos de búsqueda tiene un especial atractivo, pues son fáciles de representar, visualizar y modificar. Además, diversos problemas reales, como el movimiento de manipuladores robóticos, el alineamiento de secuencias de ADN, o la búsqueda de caminos en juegos de estrategia en tiempo real (de especial interés en nuestro caso), pueden representarse como búsquedas en mallas. La combinación de un modelo sencillo con aplicaciones reales actúa sin duda como un elemento motivador en el interés del alumnado.

El uso de herramientas para la simulación de algoritmos de búsqueda no es nuevo. Quizá el antecesor más llamativo entre estas herramientas sea Theseus, el ratón electromecánico presentado por Claude Shannon en 1951 [20]. Se trata de un dispositivo

en el que un pequeño ratón se mueve por un laberinto cuadrado de 5×5 celdas, que se puede configurar colocando obstáculos metálicos en 40 pequeñas ranuras. Mediante el uso de relés telefónicos, el aparato implementa una búsqueda con retroceso que permite al ratón encontrar y recordar el camino desde una posición cualquiera a otra objetivo (un "queso" metálico). El aparato, que en su día despertó el interés mediático, se encuentra actualmente en el museo del MIT [1].

El principal precedente de la herramienta descrita en este artículo es la herramienta PathDemo [2] [22], que ilustra diversos algoritmos empleados en búsqueda de caminos en juegos de estrategia mediante una malla con obstáculos. Esta herramienta reune los principales elementos necesarios para una buena simulación de los algoritmos de búsqueda, pero presenta también algunos inconvenientes para su uso como recurso didáctico. En primer lugar, el código no es abierto, con lo cual no se puede modificar, adaptar ni ampliar. En segundo lugar sólo está disponible para plataformas Windows. Los algoritmos disponibles no se corresponden claramente con la clasificación normalmente empleada en clase y, dada su antiguedad, carece de una técnica de gran importancia y desarrollo reciente, como es la búsqueda frontera [7], ya mencionada anteriormente. Por último, el editor de mallas no permite guardar ni cargar problemas ya editados, lo que limita considerablemente su uso para la realización de prácticas por parte del alumnado.

4. La herramienta de simulación

Una descripción de las motivaciones y proceso de desarrollo de la herramienta de simulación se encuentra en [10]. Detallaremos aquí los aspectos básicos relativos a las prácticas y tareas presentadas más adelante. La herramienta se ha bautizado recientemente con el nombre de AIDA, ya que A* e IDA* son los algorimos de referencia más utilizados en búsqueda heurística. El programa está disponible públicamente en Google Code a través de la dirección `http://code.google.com/p/aida-uma/`.

La herramienta se ha desarrollado en Common Lisp, empleando el paquete gráfico de la plataforma LispWorks[3]. Esta plataforma dispone de una versión personal de libre disposición, que permite la ejecución del programa en plataformas Windows, Linux, o Mac sin necesidad de ningún cambio en el código. Existe también un prototipo desarrollado en el lenguaje Java.

El simulador permite editar, guardar, y cargar cuadrículas con obstáculos, y definir nodos de inicio y fin. Las cuadrículas siguen una topología de 4-vecinos. El coste de las transiciones es unitario. A continuación se puede visualizar paso a paso la ejecución de diversas combinaciones de algoritmos y heurísticos. En las prácticas descritas más adelante se consideran los algoritmos heurísticos A*, IDA*, y búsqueda frontera. En cuanto a los heurísticos, se dispone de búsqueda a ciegas, distancia Euclídea, y distancia Manhattan.

En relación a la implementación de los algoritmos de búsqueda en Lisp, el uso del paradigma funcional ha tenido tradicionalmente predicamento en la comunidad docente [13] [17]. Sin embargo, en nuestro caso hemos optado por una implementación orientada a objetos (CLOS). Los algoritmos están implementados siguiendo un estilo recursivo y se ejecutan como hebras o procesos ligeros (light processes). La llamada recursiva de cada iteración se difiere mediante una llamada al planificador de procesos de LispWorks (scheduler). De este modo, es fácil modificar en tiempo real la velocidad de simulación, simplemente cambiando el tiempo de espera que se proporciona al planificador para la ejecución de la siguiente llamada recursiva.

5. Evaluación

La acogida de la herramienta por el alumnado ha sido buena. La realización de encuestas arrojó una valoración media global de 4,26 sobre 5 sobre su utilidad para el aprendizaje de los algoritmos. La figura 1 muestra en detalle las respuestas proporcionadas por el alumnado de varios grupos, asignaturas y cursos.

Recientemente se ha incluido en las encuestas una valoración específica para cada algoritmo de búsqueda. La figura 2 resume los resultados.

[1] `http://webmuseum.mit.edu/detail.php?t=exhibitions&type=exh&f=&s=3&record=5`

[2] `http://www.programmersheaven.com/download/1001/download.aspx`

[3] `http://www.lispworks.com`

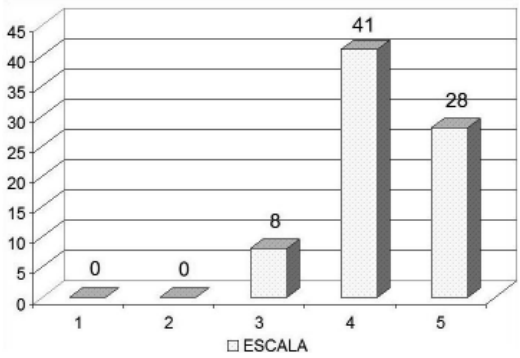


Figura 1: Respuestas a la cuestión "Valore el programa para el aprendizaje de los algoritmos de búsqueda" (1- Insuficiente, 3- Bien, 5- Excelente).

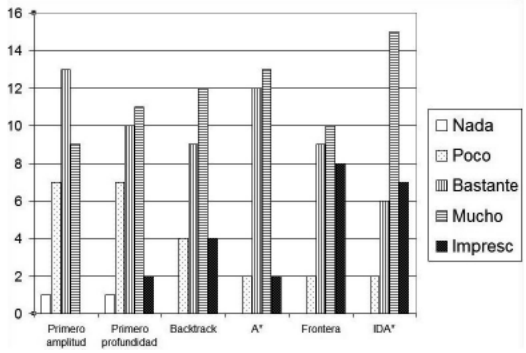


Figura 2: Respuestas a la cuestión "El uso de la herramienta me ha permitido mejorar mi comprensión de los siguientes algoritmos".

6. Trabajos para el alumnado

6.1. No siempre conviene un heurístico "mejor"

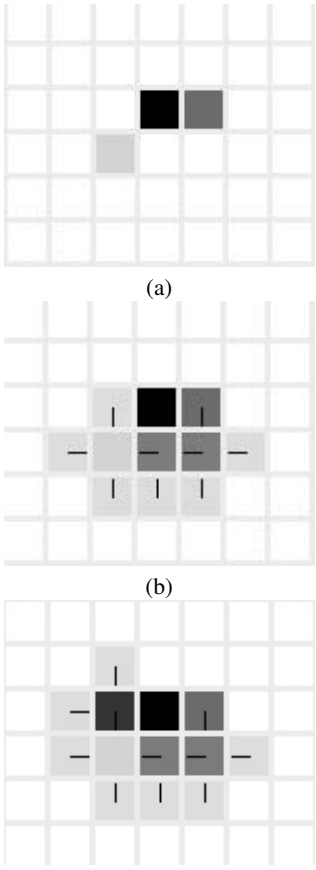
Este es uno de los malentendidos más frecuentes identificado en [4]. Dados dos heurísticos optimistas para un mismo problema $h_1(n)$ y $h_2(n)$, es posible demostrar que si $\forall n \notin \Gamma \quad h_2(n) > h_1(n)$ (i.e. h_2 está *más informado* que h_1), y ambos son monótonos, entonces A* con h_2 nunca realizará más iteraciones que con h_1. Por tanto, dado cualquier heurístico monótono $h(n)$ tal que $\forall n \notin \Gamma \quad h(n) > 0$, lo razonable es usar A* frente al algoritmo de Dijkstra. Sin embargo, para un mismo problema es difícil encontrar dos heurísticos no triviales tales que uno esté más informado que otro. En el caso de las mallas cuadradas, si h_1 mide la distancia Euclídea al objetivo, y h_2 mide la distancia Manhattan, es fácil comprobar que $\forall n \notin \Gamma \quad h_2(n) \geq h_1(n)$, con lo que no se cumple la desigualdad estricta. Si un nodo está en la misma fila o columna que el objetivo, las dos estimaciones serán idénticas.

Aunque en general en esta circunstancia lo razonable sería usar la distancia Manhattan, la mayoría del alumnado tienen dificultad en reconocer que el uso de la distancia Euclídea puede ser más beneficioso en casos especiales. El sutíl ejemplo de [4] no se puede reproducir en nuestro modelo, pero se puede mostrar que, para una regla de desempate dada, podemos encontrarnos con la misma situación. La figura 3 muestra uno de tales ejemplos, en el que A* con la distancia Euclídea debe realizar tan sólo 4 iteraciones, mientras que A* con la distancia Manhattan debe realizar 5. El ejemplo consta únicamente del nodo de salida en la parte inferior (verde), un obstáculo (negro), y el nodo objetivo a la derecha (rojo). La ejecución con la distancia Euclídea sólo expande los nodos del camino solución (magenta). Por el contrario, la ejecución con la distancia Manhattan debe expandir también un vecino adicional del nodo de salida. Los restantes nodos (azul claro) quedan en abiertos sin explorar.

Buscar la explicación a este fenómeno es un trabajo interesante para la reflexión del alumnado. También lo es encontrar otras instancias análogas del problema.

6.2. El esfuerzo de búsqueda depende de la dirección

Otro de los malentendidos usuales en los problemas de búsqueda heurística es considerar que, dado un problema, el esfuerzo necesario para encontrar una solución que vaya desde el punto A al punto B es el mismo encontrar un camino que vaya del punto B al punto A. Por ello, en clase se suelen proponer ejercicios como el que se ilustra a continuación para que el alumnado reflexione sobre esta paradoja. La importancia de este hecho quedará de manifiesto al abordar más adelante los temas de generación de planes y los conceptos de búsqueda por objetivos y



(a) (b) (c)

Figura 3: (a) Instancia de problema en la que la distancia Euclídea es mejor heurístico que la distancia Manhatan. (b) Traza de A* con la distancia Euclídea. (c) Traza de A* con la distancia Manhattan.

análisis medios/fines.

El problema consiste en encontrar un camino óptimo, usando el algoritmo A* y el heurístico de distancia de Manhattan sobre la figura 4(a). El origen de la búsqueda es el punto verde y el destino el rojo. Al realizar una búsqueda con las condiciones especificadas obtendremos la expansión de nodos mostrada en la figura, en la que los nodos en magenta representan el camino encontrado, los nodos azules los nodos expandidos y los azul claro los generados pero no expandidos. En este primer caso la memoria utilizada son 68 nodos y el número de iteraciones para encontrar la solución 43.

Sin embargo, intercambiando origen y destino obtendremos como resultado la figura 4(b) en la que el camino encontrado tiene la misma longitud que en el caso anterior (16) pero el esfuerzo de búsqueda es claramente superior con 88 nodos en memoria y 54 iteraciones. Situaciones similares pueden encontrarse también empleando el algoritmo IDA*.

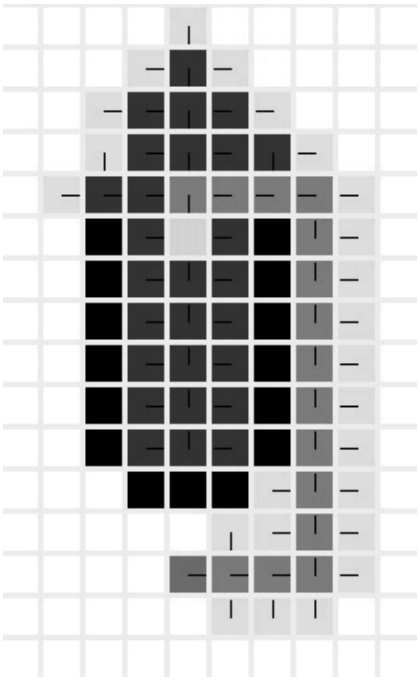
6.3. Rendimiento de la búsqueda frontera

La búsqueda frontera tiene gran aplicación en problemas donde A* sería la opción idónea, pero no es práctico debido a su gran consumo de memoria. Prácticamente en cualquier problema con mallas es posible ilustrar las ventajas en consumo de memoria de la búsqueda frontera. La figura 5 muestra la traza de un problema resuelto mediante búsqueda en amplitud (a) (con 134 iteraciones y 169 nodos), y en dos instantes de la búsqueda frontera-amplitud (con 453 iteraciones y sólo 52 nodos), concretamente en la primera iteración (b) y en un instante de la recuperación recursiva de la solución (c).

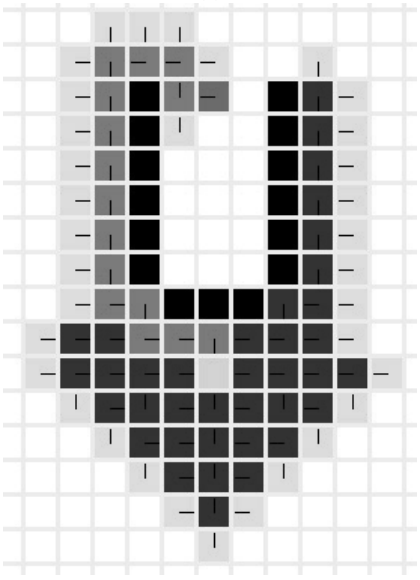
7. Conclusiones y trabajo futuro

Este trabajo describe el uso de una herramienta de simulación para la enseñanza de algoritmos de búsqueda heurística en asignaturas de Inteligencia Artificial. Las encuestas de satisfacción del alumnado revelan que su uso ha recibido una valoración muy positiva. Al tratarse de una materia prácticamente obligatoria en los planes de estudio superiores de Informática la herramienta puede ser interesante para profesores y alumnos.

La herramienta permite editar mallas cuadradas y simular paso a paso la ejecución de los algoritmos,

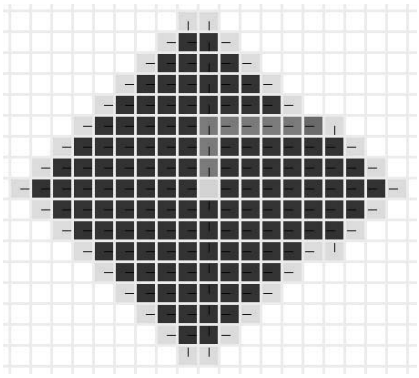


(a)

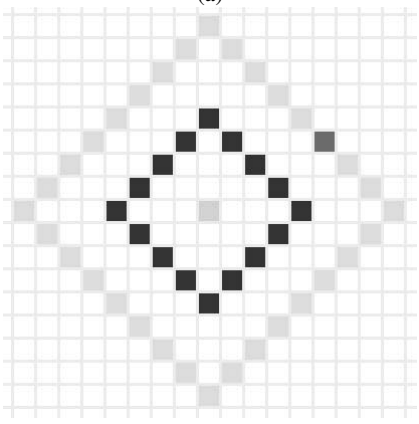


(b)

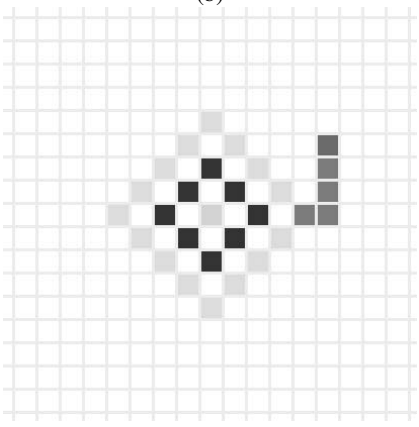
Figura 4: (a) Resultado de la búsqueda A* en un problema empleando la distancia Manahattan. (b) Resultado en el mismo problema invirtiendo origen y destino.



(a)



(b)



(c)

Figura 5: (a) Traza de la búsqueda en amplitud sobre un problema. (b) Un instante de la búsqueda frontera inicial sobre el mismo problema. (c) Un instante de la recuperación recursiva de la solución.

proporcionando información sobre su rendimiento. La herramienta favorece la participación del alumnado, aumenta sus posibilidades de exploración, y ofrece la oportunidad de plantear prácticas en las que se aclaran de manera gráfica e intuitiva aspectos relacionados con la materia.

Existen numerosas posibilidades para continuar esta experiencia, integrando en la herramienta simulación de problemas análogos, como mallas de 8-vecinos, alineamiento de secuencias, así como ilustrar problemas relacionados con en ámbito de los juegos de estrategia en tiempo real. Estos incluyen el uso de tablas de transposición, el movimiento coordinado o la búsqueda jerárquica.

Referencias

[1] The Joint Task Force on Computing Curricula IEEE Computer Society Association for Computing Machinery. *Computing Curricula 2001. Computer Science. Final Report (december 15, 2001)*.

[2] Culberson, J., Schaeffer, J. *Pattern databases*, Computational Intelligence 14(3) (1998) 318-334.

[3] Fleischer, R., Kucera, L. *Algorithm Animation for Teaching*. En Software Visualization. LNCS 2269. Stefan Diehl (Ed.), pp. 113-128. Springer, 2002.

[4] Holte, R.C. *Common Misconceptions Concerning Heuristic Search*, Symposium on Combinatorial Search (SoCS'09).

[5] Hundhausen, C.D., Douglas, S.A., Stasko, J.T. *A meta-study of algorithm visualization effectiveness*. Journal of Visual Languages and Computing, 13(3):259-290, 2002.

[6] Kehoe, C., Stasko, J., Taylor, A. *Rethinking the Evaluation of Algorithm Animations as Learning Aids: An Observational Study*, International Journal of Human-Computer Studies, Vol. 54, No. 2, pp. 265-284, 2001.

[7] Korf, R., Zhang, W., Thayer, I., Hohwald, H. *Frontier search*. JACM, Vol 52, Issue 5, pp 715-748, 2005.

[8] Lawrence, A.W. *Empirically Evaluating the Use of Animation to Teach Algorithms* Tech. Rep. GIT-GVU-94-07, 1994.

[9] Levitin, A., Papalaskari, M.A. *Using puzzles in teaching algorithms* Proceedings of the 33rd SIGCSE technical symposium on computer science education, pp. 292-296.

[10] Mandow, L., Coego, J., Villalba, F. *Herramienta software para la enseñanza de algoritmos de búsqueda*. III Jornadas de Innovación Educativa y Enseñanza Virtual en la Universidad de Málaga, 2009.

[11] Marín, R., Palma J.T. *Inteligencia Artificial: métodos, técnicas y aplicaciones*. McGraw-Hill, 2008.

[12] Newell, A., Simon, H. *Computer Science as Empirical Enquiry: Symbols and Search*, en Mind Design II, J. Haugeland (ed.)

[13] Novig, P. *Paradigms of Artificial Intelligence Programming*. Morgan Kaufmann, 1992.

[14] Nilsson, N. *Inteligencia Artificial: una nueva síntesis*. McGraw-Hill, 2001.

[15] Pearl, J. *Heuristics*. Addison-Wesley, 1984.

[16] Reinefeld, A. *Complete Solution of the Eight-Puzzle and the Benefit of Node Ordering in IDA**. IJCAI'1993, pp. 248-253.

[17] Rubio, J., Bañares, J.A., Muro-Medrano, P.R. *Una aproximación funcional a la implementación de algoritmos de búsqueda*. CAEPIA-TTIA'99, pp 88-97.

[18] Russell, S., Novig, P. *Inteligencia Artificial: un enfoque moderno*. Prentice-Hall, 2001.

[19] Sánchez Nielsen, E. *Integrando tecnologías de desarrollo de Google en las asignaturas de Inteligencia Artificial y Sistemas Inteligentes*. JENUI'2009, pp. 233-240.

[20] Shannon, C.E. *Presentation of a maze-solving machine*, en Claude E. Shannon: Collected Papers, Wiley, 1993.

[21] Stasko, J., Bradre, A., Lewis, C. (1993). *Do Algorithm Animations Assist Learning? An Empirical Study and Analysis* ACM INTERCHI, pp.61-66, 1993.

[22] Stout, B. *Smart moves: intelligent pathfinding*. Game Developer Magazine, July, 1997.

[23] Zyda, M., Koenig, S. *Teaching Artificial Intelligence Playfully* Proceedings of the AAAI-08 Education Colloquium, 2008.