Agente Inteligente Resolvedor de Laberintos (Airla)

Agent Intelligent Solver Of Labyrinth (Airla)

Nelson Becerra Correa
Edgar Altamirano Carmona

Fecha recepción: 13 de enero 13 de 2013

Fecha aceptación: 4 de mayo de 2013

Resumen

Dentro del campo de la Inteligencia Artificial, los Sistemas Inteligentes, se han caracterizado por ofrecer una técnica para la solución de problemas complejos con características distribuidas.

A la hora de abordar el desarrollo de sistemas inteligente es indudable un notable aumento de complejidad, así como la necesidad de adaptar técnicas ya existentes o, en ocasiones, el desarrollo de técnicas y herramientas nuevas.

Para desarrollar el software utilizamos algunos y herramientas de la inteligencia artificial, que nos facilitaron la obtener una solución fiable.

En este artículo, se presentan los principales resultados de un proyecto que implementa mediante un agente inteligente el problema de resolución de laberintos.

Palabras clave

GAIA, modelamiento, juego de estrategia, Metodologías, Agentes, Sistemas Multiagente, Tiempo Real, Agente Inteligente.

* Docente Facultad Tecnológica Universidad Distrital FJC – Colombia, Investigador Fundación FABBECOR.ONG.

** Docente Universidad Autónoma de Guerrero, México. Investigador Fundación FABBECOR.ONG.

Abstract

Within the field of artificial intelligence, intelligent systems have been characterized by offering a technique for solving complex problems with distributed characteristics.

When addressing the development of intelligent systems is undoubtedly a significant increase in complexity, and the need to adapt existing techniques or, sometimes, the development of new tools and techniques.

To develop the software and tools we use some artificial intelligence that facilitated us to obtain a reliable solution.

In this article, we present the main results of a project implemented by an intelligent agent problem solving mazes.

Key words

GAIA, modeling, strategy game, Methodologies, Agents, Multi-Agent Systems, Real-Time Intelligent Agent.

1- Introducción

El paradigma conocido como agentes y sistemas multiagentes inteligentes, constituye actualmente un área de creciente interés dentro de la Inteligencia Artificial, entre otras razones, por ser aplicable a la resolución de problemas complejos no resueltos de manera satisfactoria mediante técnicas clásicas.

Numerosas aplicaciones basadas en este nuevo paradigma vienen ya siendo empleadas en infinidad de áreas [Jennings98], tales como control de procesos, procesos de producción, control de tráfico aéreo, aplicaciones comerciales, gestión de información, comercio electrónico, aplicaciones médicas, juegos, etc.. [1]

Uno de los problemas en el área de agentes es el hecho que cada vez son más necesarios métodos, técnicas y herramientas que faciliten el desarrollo de aplicaciones basadas en dicho paradigma.

La mayor dificultad, en el aprendizaje de agentes inteligentes, es lograr diferenciar claramente entre programación por objetos, estructurada, sistemas expertos y agentes inteligentes. Esto al momento de su implementación.

Muchos alumnos cuando se ven enfrentados a construir sistemas inteligentes logran aplicaciónes bonitas y funcionales pero si se observan en detalla es simple y llanamente programación tradicional, ayudados por alguna metodología de agentes. Lo cual no implica que sean agentes inteligentes.

Este proyecto pretende describir de manera sencilla, la construcción de un agente inteligente, conocido como débil con una o dos características inteligentes (reactividad, au-

tonomía), para solucionar el problema de un laberinto.

2. El problema

- En un ambiente abierto, hay árboles y otro tipo de obstáculos, ubicados aleatoriamente. Estos obstáculos impiden transitar libre a un agente inteligente.
- En este parque, además, hay un río que divide el parque en dos partes. Para cruzar el rio, es necesario hacerlo por un puente construido sobre él.
- La salida está en un extremo del parque.
- El agente inteligente se encuentra ubicado en algún lugar del parque (ubicación aleatoria) y deberá encontrar la salida. El Agente debe recorrer el parque sin chocar con ningún árbol u obstáculo y además no debe caer al rio (moriría).

3. Desarrollo teórico

3.1 El agente

Un agente**, "Es todo aquello que percibe su ambiente mediante sensores y que responde o actúa mediante *efectores." Russell and Norvig , The AIMA Agent, 1995.***[1]

El modelo abstracto de una agente inteligente lo podemos describir de la siguiente manera:

El entorno: S={s1, ...,sn} , las Acciones A= {a1,...an} (capacidad de actuar del agente), la Acción: S^k ® A, la Interacción Agente-entorno (historia) h: $S_0 \rightarrow^{a0} S_{1\ .....} \text{®}\ S_0$ y Observación del entorno: ver:S $\rightarrow$P.

3.2 Diseño del agente

Para el desarrollo del proyecto se decidió utilizar un agente con memoria [6].

función *AGENTE-CON-MODELO (percepción)* ***retorna*** *una acción*

estático: *reglas, conjunto de reglas condición-acción*

estado ←*ACTUALIZAR-ESTADO (estado, percepción)*

regla ←*REGLA-COINCIDENCIA (estado, reglas)*

acción←*REGLA-ACCION [regla]*

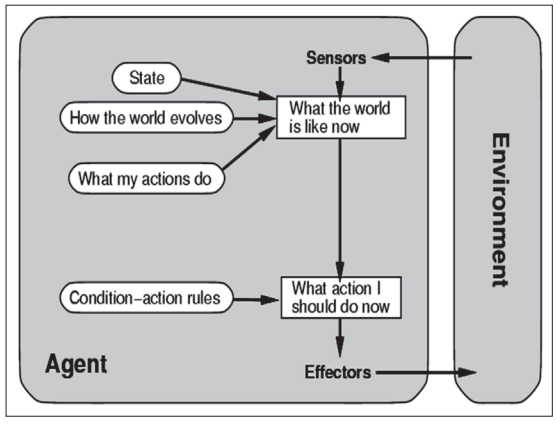
estado ← *ACTUALIZAR-ESTADO (estado, acción)*

retorne *acción*

Características del agente: Un agente inteligente lo podemos clasificar según sus características como agente débil y fuerte. Un *agente débil* tendrá como mínimo las siguientes características: Autonomía, Reactiviad, Proactividad, Habilidad social. La autonomía se refiere a que su comportamiento depende de las experiencias adquiridas. Un agente reactivo, responde a los cambios en su ambiente. La pro actividad tiene que ver con el aprovechamiento de las oportunidades que se le presentan si y solo si favorecen sus propios intereses. Si un agente se comunica con otros decimos que es social [2],[3].

Adicionalmente a estas características si una agente tiene deseos, intenciones, creencias,

Figura 1. Agente con memoria interna



Fuente: http://eisc.univalle.edu.co/~oscarbed/IA/Tema2.pdf marzo de 2013)

Tabla 1. Tabla percepción de agente

Código	Percepción	Valor 0	Valor 1	Símbolo grafico
OR	Obstáculo Rio	No hay obstáculo	Hay obstáculo	
OO	Obstáculo Otro	No hay obstáculo	Hay obstáculo	
OA	Obstáculo Árbol	No hay obstáculo	Hay obstáculo	
CO	Origen	El agente no está en el origen	El agente está en el origen	
ME	Meta	El agente No está en la meta	El agente llego a la meta	

Fuente: elaboración propia.

racionalidad, veracidad y adaptabilidad decimos que es un *agente fuerte*. [4]

Otra manera de clasificar a los agentes inteligentes tiene que ver con su estructura, como está organizada su información. Tenemos agentes de reflejo simple, agentes basados en metas, agentes bien informados de los que pasa (con memoria), basados en utilidad [7].

4. Desarrollo experimental

En nuestro proyecto, trabajamos con un agente bien informado de lo que pasa también conocido como agente con memoria [5]. A continuación describimos nuestro agente.

El agente con memoria además de tratar de alcanzar el objetivo en base a una tabla de percepción acción. Tiene en cuenta los siguientes elementos:

- Estado
- Cómo evoluciona el mundo
- Lo que hacen mis acciones
  - Agente está constituido por
  - Sensores
  - Actuadores
  - Metas

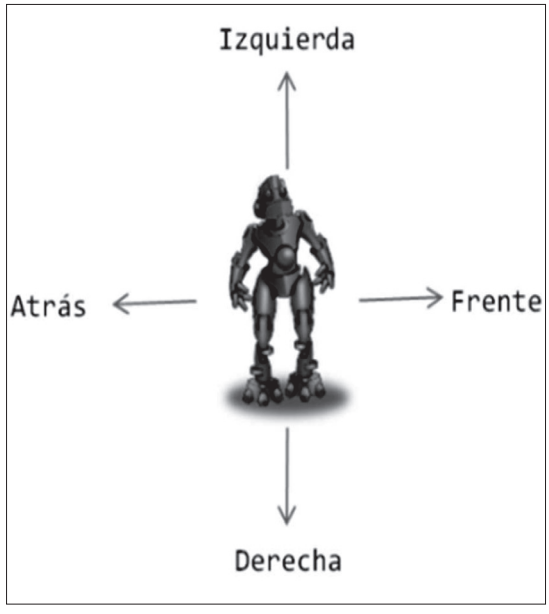
Percepciones: el agente tendrá un sensor que identificara el estado en que se encuentra una casilla adjunta. La representación mediante una tabla es mostrada en la figura 1.

Acciones: Las acciones emprendidas por el agente son:

- Avanzar derecha
- Avanzar izquierda
- Avanzar arriba
- Avanzar a bajo

La representación de las acciones mediante una tabla, se muestran en la tabla 2.

Figura 2. Elemento gráfico



Fuente: elaboración propia.

Tabla 2. Tabla acción de agente

Código	Acción	Símbolo grafico
AA	Avanza Arriba	Avanzar una casilla a arriba
AB	Avanza Abajo	Desplazarse una casilla a bajo
AD	Avanza Derecha	Avanzar una casilla a la derecha
AI	Avanzar Izquierda	Avanzar una casilla a la izquierda

Fuente: elaboración propia.

Ambiente: El ambiente al que se enfrente nuestro agente es:

- Accesible: Pues los sensores proporcionan todo lo que hay que saber sobre el estado completo del ambiente.
- Determinístico: El estado siguiente del ambiente está determinado plenamente por el estado presente del mismo, y por la acción del agente.
- Episódico: Es episódico ya que implica que los episodios siguientes no dependen de las acciones que ocurrían en episodios previos.
- Estático: El ambiente que no cambia mientras el agente está pensando.
- Discreto: Discreto - con escaso número de percepciones y acciones en el ambiente.

Tabla de percepción acción: La tabla 1, muestra el número de entradas, las cuales dependen de la dimensión del parque el cual es de nxm, se considera que el máximo número para el cual fue hecho el software es de 20 x 20.

Estado: Un estado está conformado por el tablero y la percepción actual, en tabla 4. El * significa que el agente puede tomar cualquier orientación.

El desempeño se toma como -1000, si el agente se cae al rio, -100 si se encuentra con

Tabla 3: percepción - acción

Percepciones						Acciones	Desempeño
Entrada	Obstáculo			Meta	Orientación (↑→↓←)		
	OR	OA	OO				
0	0	0	0	0	*	Av	-1
1	1	0	0	0	*	Av ↑	-1
2	1	0	0	0	*	Av ↓	-1
3	1	0	0	0	*	Av →	-1.000
4	1	0	0	0	*	Av ←	-1

Fuente: elaboración propia.

Tabla 4. Desempeño general de agente

Ambiente	Percepción	Acción	Desempeño Distancia Manhattan	Desempeño acumulado
AG / Meta	No hay obstáculo árbol al frente y No hay obstáculo otro al frente y No hay rio al frente (0,0,0) →	Avanzar →	-1	0
AG / Meta	No hay obstáculo árbol al frente y No hay obstáculo otro al frente y No hay rio al frente (0,0,0) ↑	Avanzar ↑	100	99
AG / Meta	No hay obstáculo árbol al frente y No hay obstáculo otro al frente y No hay rio al frente (0,0,0) ↓	Avanzar ↓	-1	98

Fuente: elaboración propia.

un obstáculo diferente, si se cae al rio el agente muere. Para cualquier otro obstáculo, la acción puede ser reversible.

Ejemplo que ilustra como el agente se desempeña en el parque, el color corresponde al puente y el color azul corresponde al rio

5. Características del agente

Nuestro agente inteligente, tiene las siguientes características:

- Autonomía, reactivo y proactivo.
- La autonomía, entienda como, la capacidad del agente de tomar sus propias decisiones basado más en sus experiencias que en su conocimiento dado por el programador.
- El agente solucionador de laberintos, almacena sus experiencias en forma de reglas e infiere sus conocimiento, utilizando el encadenamiento adelante.
- El agente es reactivo, ya que una vez modificado el entorno cambia de posición en el parque. Este inmediatamente genera un proceso nuevo para la solución el problema.
- El agente es proactivo, pues aprovecha cada oportunidad presentada en el ambiente para lograr sus objetivos.
- Nuestro agente, no implementa más características por el momento.

5.1 Medida de rendimiento

Nuestro agente, utiliza como medida de rendimiento, un valor de 1 si logra avanzar un cuadro en el parque en forma segura.

Recibirá una penalización de 1000 punto si este se cae al lago, ya que en este caso el agente morirá y una penalización de 100 puntos si choca con algún objeto pues esta acción es reversible.

5.2 Implementación

El algoritmo general para la implementación del agente de reflejo simple es:

Tabla 5. Algoritmo general de agente

1. Leer archivo de configuración inicial (.ini)
2. Generar ambiente aleatorio
3. Iniciar agente
4. Si condición de parada = 1.
a. Termina
b. Si no ir a 5
5. Leer percepciones del estado actual
6. Realizar acción según tabla. Ir a 4

Fuente: elaboración propia.

Tabla 6. Algoritmo general de agente con memoria interna

1. Leer archivo de configuración inicial (.ini)
2. Generar ambiente aleatorio
3. Iniciar agente
4. Si condición de parada = 1.
a. Termina
b. Si no ir a 5
5. Leer percepciones del estado actual
6. Realizar acción según reglas
7. Calcular estadísticos
8. Actualizar estado interno. Ir a 4.

Fuente: elaboración propia.

Se implementó el agente inteligente en lenguaje C++.

5.3 Rendimiento

- Diagrama general de la implementación
- Basado en metas

Complejidad del agente:

Bloque principal :hace n operaciones y entra en un ciclo del cual llama a un método de 1 operación

Percepción p = ambiente.Percepcion() que llamada a una función con 20 operaciones

Acción a = agenteConMemoria(ambiente.Percepcion());

este a su vez , llamada a un método: 1 operación ambiente.realizarAccion(a);

Complejidad Bloque principal: n*(1+12+1)= 10*(1+20+1)=220 operaciones

Bloque percepción: se actualiza la descripción interna del estado que mantiene el agente hace un llamado a la Acción

agenteConMemoria(ambiente.Percepcion()); este a su vez , llamada a un método: 1 operación ambiente.realizarAccion(a);

Complejidad Bloque principal: n*(1+12+1)= 10*(1+20+1)=220 operaciones

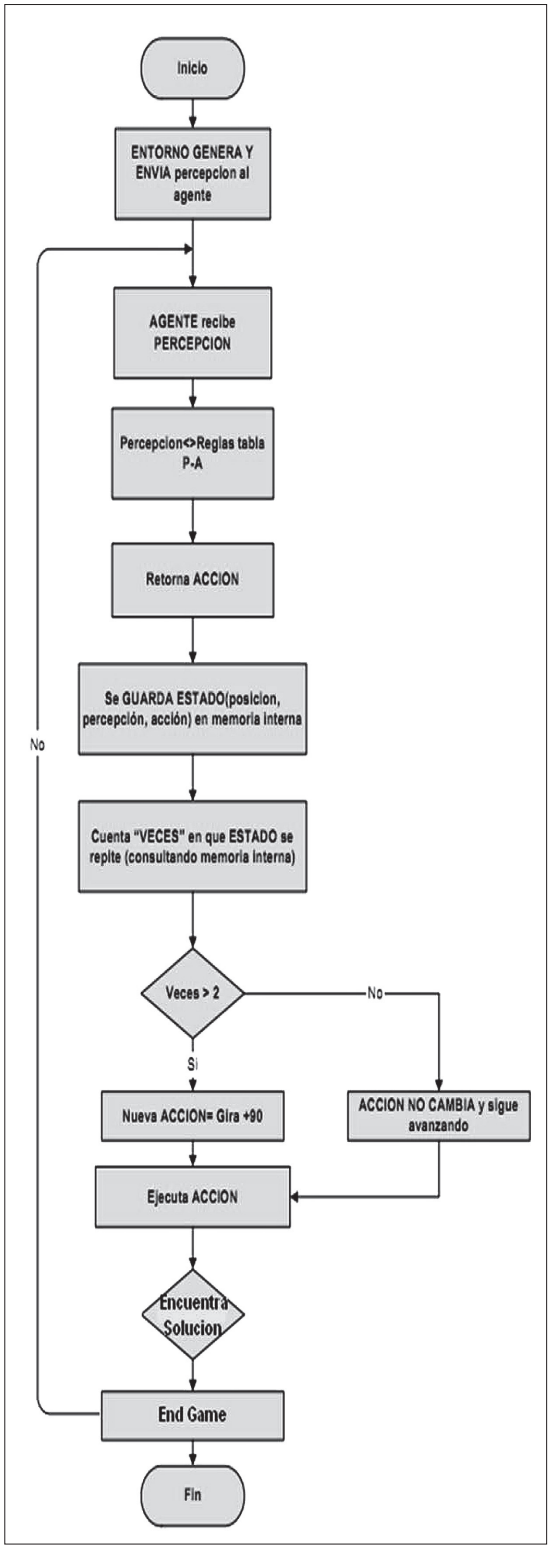
Bloque percepción: se actualiza la descripción interna del estado que mantiene el agente hace un llamado a la Acción agenteConMemoria(Percepción p) que llamada a una función con 3 operaciones, estado = actualiza_estado (estado, p) a su vez llamada a una función: 4 operaciones y regla = selecciona_regla (estado, reglas) , llamada a una función de 1 operación , la función Accion acción=aplica_regla(regla) llamada a una función que 12 operaciones, estado = actualiza_estado (estado, acción)

Tabla 7. Número de pruebas realizadas para el agente

Dimensión del parque					
NxN	**10**	**50**	**200**	**500**	**1000**
Complejidad del agente. (Número de operaciones que se requieren realizar para obtener la salida)	220	1111	4444	11111	22222

Fuente: elaboración propia.

Figura 3. Diagrama general del agente



Fuente: elaboración propia.

Complejidad Bloque percepción: 20 operaciones

Bloque actualizar estado respecto a la percepción: Estado actualiza_estado(Estado e, Percepcion p) , Solo llama a uno de las dos siguientes

Complejidad bloque : 3 operaciones

Bloque actualizar estado respecto a la acción: Llama a actualización de estado actualiza_estado(Estado e, Acción p) , en este caso estamos considerando el peor de los casos, ya que en esta última función hay varias alternativas else if", pero el bloque que contiene el mayor número de operaciones es el primero, es decir 12 operaciones

Complejidad Bloque actualizar estado: 12 operaciones O(1)

En este apartado medimos, el rendimiento del agente. Para ello tenemos en cuenta los siguientes parámetros:

- Tamaño de la matriz, la cual probaremos de una dimensión de 3 a 20.
- Numero de obstáculos iniciales
- Cantidad de objetos con los cuales choca el agente
- Probabilidad de objetos colocados = 0.5
- Probabilidad de puentes = 0.4
- Numero de iteraciones para cada corrida = 200

% Éxito = (Numero de obstáculos iniciales – Numero choques con obstáculos)/(Numero de obstáculos iniciales)

6. Conclusiones

- La aplicación arroja mejores resultados al determinar una cantidad menor de obstáculos dentro del ambiente, como se espera que sea.

128

Tabla 8. Rendimiento realizado por el agente, hasta obtener la salida

Tamaño matriz	Rendimiento	Numero obstáculos	Numero de choques con obstáculos	% de éxito
3	103	4	1	0.75

- El uso de la memoria del agente es indispensable para evitar que el agente llegue a ciclos repetitivos infinitos.
- El uso de Reglas de Producción como forma de Representación de Conocimiento es la más adecuada para este tipo de agente ya que el objetivo es llegar a la meta siendo indiferente el costo computacional.

7. Trabajos futuros

Este trabajo de desarrollo implementar un solo agente inteligente, como trabajo futuro se puede ampliar, a la construcción de tres agentes inteligentes, que se comunican entre si. Otra trabajo futuro tendría que ver con el ambiente, el cual se puede hacer más complejo e implementado otros agentes que permitan comparar el desempeño.

8. Referencias

[1] INTELIGENCIA ARTIFICIAL. UN ENFOQUE MODERNO. Segunda edición. **Stuart** J. Russell y Peter Norvig. Traducción: ‚Juan Manuef Corchzrdo Rodriguez

[2] Jennings, N. Wooldridge, M.:Applications of Intelligent Agents. Queen Mary & Westfield College. University of London

[3] Informe metodología GAIA: "Buscador de talento amigo" Arquitectura de sistemas multi-agentes Universidad Autónoma de Manizales 2008.

[4] Ingeniería de Software Orientada a Agentes; Universidad Pontificia de Salamanca Campus de Madrid.

[5] Uso de la metodología GAIA para modelar el comportamiento de personajes en un juego de estrategia en tiempo real; Fac. Ing. Univ. Antioquia N° 53 pp. 214-224. Junio, 2010.

[6] Estudio de métodos de desarrollo de sistemas multiagente; Vicente J. Julián, Vicente J. Botti Departamento de Sistemas Informáticos y Computación Universidad Politécnica de Valencia

[7] Breve análisis de algunas metodologías e diseño de SMA; Departamento de Ciencia de la Computación e Inteligencia Artificial; Universidad de Alicante 2004