Capítulo 1

Agentes *inteligentes* y ambientes

Objetivo

El alumno explicará qué es un agente inteligente, su medio, y cómo se construyen mediante su estructura y tipos de ambientes

1.1. Estructura general de *agentes*

Un *agente*, del latín *agere*, "hacer", es todo aquello que ***percibe*** su ambiente mediante *sensores* y que ***responde*** en tal ambiente por medio de *actuadores.*

Figura 1.1: Agente secreto

Los agentes humanos tienen ojos, oídos y otros órganos que sirven de sensores, así como manos, piernas, boca y otras partes de su cuerpo que sirven de actuadores. En el caso de agentes robóticos, los sensores son sustituidos por cámaras, infrarrojos, etc., y los actuadores generalmente son motores.

Figura 1.2: ¿Agentes patológicos?

Un *agente de software* (http://bit.ly/124UvOL) es un (sub)programa que interactúa con otros (sub)programas. En este caso, sus percepciones y acciones son cadenas codificadas de bits. Un *agente de software "inteligente"* (http://bit.ly/y2gQk) es una entidad **autónoma** que percibe y actúa en un entorno de forma tal que dirige sus actividades para alcanzar sus objetivos. Un *agente racional* (http://bit.ly/9WnPTq) es aquel que **siempre** hace lo **correcto**.

En ocasiones es conveniente asumir que lo correcto es aquello que permite al agente *obtener el mejor desempeño*. Dicho lo anterior, ahora será necesario decidir cómo y cuándo evaluar ese buen desempeño del agente.

Medir el desempeño implica definir el criterio que sirve para determinar qué tan exitoso ha sido un agente. Resulta evidente que no existe una medida fija que se pueda aplicar por igual a todos los agentes. Es necesario contar con una medición objetiva del desempeño, medida que deberá ser propuesta por una *autoridad*. Esta autoridad define la norma para considerar un *desempeño satisfactorio* en un ambiente y emplearlo en la medición del desempeño de los agentes.

1.1.1. Agente racional ideal

Un agente racional ideal se caracteriza porque en todos los casos de posibles secuencias de percepciones, deberá realizar todas aquellas acciones que favorezcan *obtener el máximo* de su medida de rendimiento, basándose en las evidencias aportadas por la secuencia de percepciones y en todo conocimiento incorporado en tal agente.

1.1.1.1. *Mapeo* de secuencias de percepciones

El comportamiento de un agente depende exclusivamente de la secuencia de percepciones en un momento dado, por tanto, es *posible* caracterizar cualquier agente en particular elaborando una **tabla de acciones** que lleva a cabo como respuesta a cualquier secuencia de percepciones. Sin embargo, para la mayoría de los agentes no es *viable* almacenar dicha tabla ya que es muy grande pudiendo llegar a ser infinita.

Esta lista se conoce como *mapeo* de percepciones a acciones. En principio, es posible determinar qué mapeo describe acertadamente a un agente, ensayando todas las posibles secuencias de

percepciones y llevando un registro de las acciones que en respuesta realiza el agente. Si mediante los mapeos se caracteriza a los agentes, los **mapeos ideales** caracterizan a los agentes ideales. Especificar qué tipo de acción deberá llevar a cabo un agente como respuesta a una determinada secuencia de percepciones constituye el diseño de un agente ideal. Lo anterior no implica, desde luego, que hay que crear una tabla explícita con una entrada por cada posible secuencia de percepciones, en vez de ello, se puede definir una especificación del mapeo sin tener que enumerarlo exhaustivamente: mediante una función.

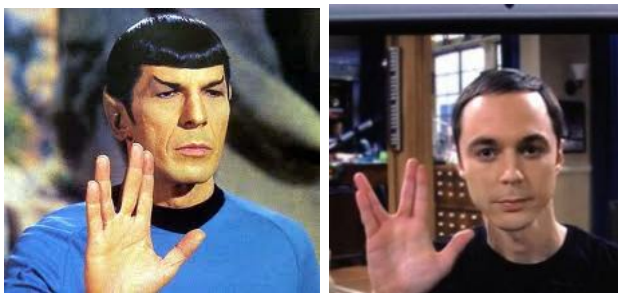

Figura 1.3: Agentes racionales

Sin embargo, en la práctica, obtener un agente racional ideal no es viable, debido a que el proceso para optimizar el resultado generalmente requiere una cantidad de tiempo excesivamente grande, aún para instancias pequeñas del problema a resolver. Por esto, se recurre a otro tipo de agentes más simples.

1.1.2. Agentes de reflejo simple

Obtener un mapeo ideal es un trabajo muy difícil de realizar: tener una tabla para todas las posibles percepciones podría requerir espacio superior a todas las partículas del Universo y, por otro lado, hay problemas para los cuales no es posible obtener una función que regrese la solución esperada.

Sin embargo, hay problemas muy sencillos en los que se puede aplicar una estrategia simple, llamada *condición-acción* que conecta percepciones con acciones predefinidas, es de la forma siguiente:

$$SI\ percepcion = condicion_i$$
$$ENTONCES\ realizarAccion_i$$

Los humanos funcionan también en ocasiones recurriendo a estas conexiones, algunas de las cuales son respuestas aprendidas (como frenar un vehículo cuando se observa algún obstáculo) y otras son reflejos innatos (como el parpadeo del ojo cuando se le aproxima algún objeto).

En la siguiente figura se presenta la estructura de un agente de reflejo simple en forma esquematizada, en ella se puede observar cómo las reglas condición-acción permiten al agente establecer la conexión entre percepciones y acciones.

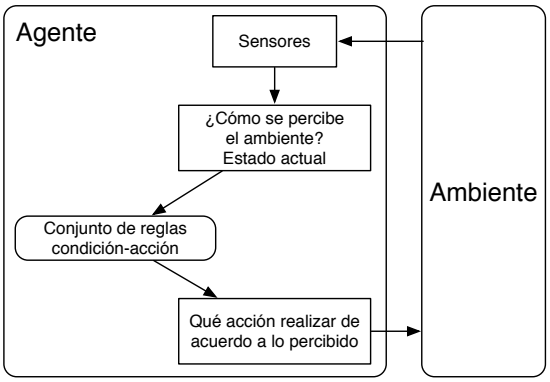

Figura 1.4: Agente de reflejo simple

El algoritmo de este tipo de agente, también muy sencillo.

Algoritmo 1.1 Agente de reflejo simple

$func\ agenteReflejoSimple\,(percepcion)$
$estado \leftarrow interpretarEntrada\,(percepcion)$
$accion \leftarrow condicionAccion\,(estado)$
$regresar\,(accion)$

La función auxiliar *interpretarEntrada* genera una descripción abstracta del estado actual dada la percepción y la función *condicionAccion* devuelve la acción asociada al estado actual. Si bien agentes como el anterior se pueden implementar con bastante eficiencia (p.e. utilizando tablas *hash*), el ámbito en donde se pueden aplicar es bastante limitado.

1.1.3. Agentes basados en logro de metas

Para decidir qué hay que hacer no siempre basta contener información acerca del estado que prevalece en el ambiente. Por ejemplo, si un robot llega a un cruce de caminos, podría tomar la decisión de seguir derecho, dar vuelta a la izquierda o a la derecha: la decisión adecuada depende del lugar al que desea llegar el robot.

Es decir, además del estado actual del ambiente, el robot necesita información acerca de su ***meta***: información acerca de situaciones deseables. El programa del agente puede combinar lo anterior con la información relativa al resultado que producirían las posibles acciones que se emprendan y, de esta manera elegir aquellas acciones que permitan alcanzar la meta. En ocasiones esto es muy sencillo: cuando alcanzar la meta depende de respondercon una sola acción; otras veces es más complicado: cuando el agente tenga que considerar largas secuencias de acciones hasta que logre encontrar las acciones que lo lleven a la meta.

La **búsqueda** y la **planificación** son subcampos de la IA que se ocupan de encontrar las secuencias de acciones que permiten a un agente alcanzar sus metas.

Es importante notar que la toma de decisiones de este tipo difiere radicalmente de las reglas *condición-acción* explicadas anteriormente. Un agente basado en metas toma en cuenta el futuro: tanto "¿Qué sucedería si realizo tal acción?" como "¿Eso me resulta *deseable*?". En el diseño

del agente reflejo esta información no se utiliza explícitamente puesto que el diseñador calcula previamente la acción *correcta* correspondiente a diversos casos.

Si bien el agente basado en metas es menos eficiente, también es cierto que es más flexible: tan sólo con especificar un nuevo destino, se produce en el agente basado en metas una nueva conducta. Las reglas de un agente de reflejo acerca de cuándo dar vuelta y cuándo seguir derecho funcionarán tan sólo en elcaso de un destino; siempre que surja uno nuevo habrá que sustituirlas.

La figura muestra la estructura del agente basado en metas.

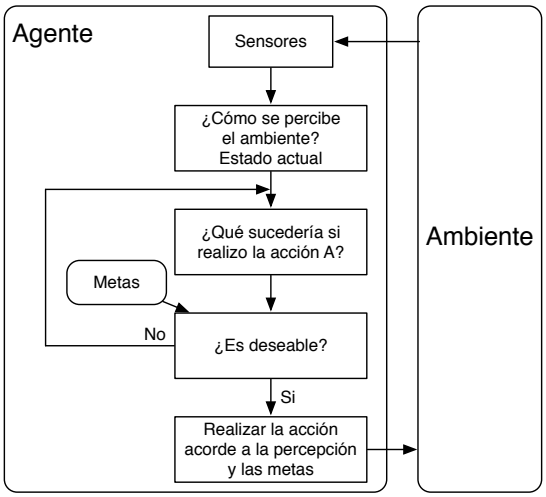

Figura 1.5: Agente basado en el logro de metas

1.1.4. Agentes basados en el logro del mejor desempeño

Las metas no bastan por sí mismas para generar una conducta *razonablemente* buena. Por ejemplo, son muchas las secuencias de acciones que permitirían a un robot llegar a su destino, con lo que se habría logrado la meta, pero de todas ellas, algunas son más rápidas o baratas que las demás. Las metas permiten establecer una tajante distinción entre estados "deseables" e "indeseables", pero con una medida de desempeño más general sería posible establecer una comparación entre los diversos estados (o secuencias de estados) del mundo.

La terminología que se acostumbra utilizar es afirmar que si se prefiere un estado del mundo por otro, entonces ese estado ofrece mayor utilidad al agente (y ofrece un mejor desempeño). Por lo tanto, la utilidad es una función que relaciona un estado con un número real que caracteriza el correspondiente grado de satisfacción. La completa especificación de la función de utilidad permite la toma de decisiones racionales en dos tipos de casos en los que las metas se encuentran con problemas.

1. Cuando el logro de algunas metas implica un conflicto, y sólo algunas de ellas se pueden obtener (p.e. la velocidad y la seguridad), la función de utilidad definirá cuál es el compromiso adecuado por el que hay que optar.

2. Cuando son varias las metas que el agente desea alcanzar, pero no existe la certeza de poder lograr ninguna de ellas, la utilidad es una vía para ponderar la posibilidad de tener éxito considerando la importancia de las diferentes metas.

Un agente que tiene una función de utilidad explícita está en posibilidad de tomar decisiones racionales, aunque quizás tenga que comparar las utilidades obtenidas mediante diversas acciones. Las metas, no obstante su inflexibilidad, permiten al agente optar de inmediato por una acción cuando ésta permite alcanzarlas. Además, en algunos casos, se puede traducir la función de utilidad en un conjunto de metas, de manera que las decisiones adoptadas por el agente basado en metas tomando como base tal conjunto de metas resulten casi idénticas a las que haría el agente basado en el desempeño.

La estructura general de un agente basado en desempeño se muestra en la figura.

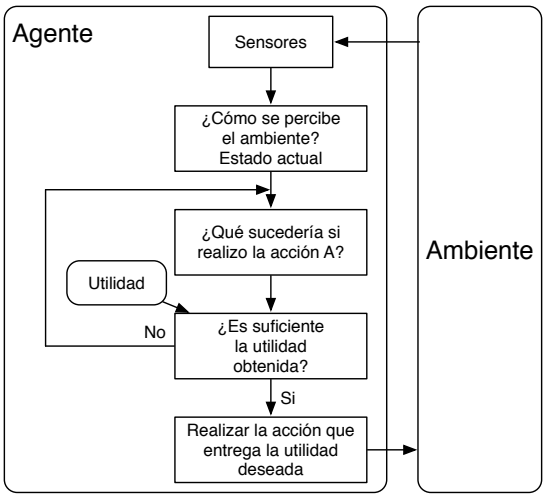

Figura 1.6: Agente basado en el mejor desempeño

Tarea:

♢ Desarrollar el algoritmo que describa el comportamiento de un agente

- basado en metas
- basado en el logro del mejor desempeño

Programa: Agente secreto

Consiste en hacer un código que se encargue de cifrar y descifrar texto de acuerdo al método mostrado a continuación:

1. Este tipo de cifrado por columna con clave consiste en formar una tabla con el número de columnas que indica la clave para codificar; a continuación, se escribe el texto en la tabla de izquierda a derecha y de arriba hacia abajo (sin espacios y, si hace falta, se rellenan los espacios de la última fila con algún caracter):

   Texto: LA CRIPTOGRAFIA ES ROMANTICA
   Clave: 4

L	A	C	R
I	P	T	O
G	R	A	F
I	A	E	S
R	O	M	A
N	T	I	C
A	S	S	S

2. Posteriormente se invierte el orden de las columnas:

L	A	C	R		R	C	A	L
I	P	T	O		O	T	P	I
G	R	A	F		F	A	R	G
I	A	E	S	⇒	S	E	A	I
R	O	M	A		A	M	O	R
N	T	I	C		C	I	T	N
A	S	S	S		S	S	S	A

3. Finalmente se toman los caracteres por columna de arriba hacia abajo y de izquierda a derecha obteniendo finalmente el texto codificado:

   ROFSACSCTAEMISAPRAOTSLIGIRNA

Para descifrar un texto codificado con este método, es necesario saber la clave, y a continuación se aplican las operaciones siguientes, mostradas para descrifrar el ejemplo anterior:

Texto: ROFSACSCTAEMISAPRAOTSLIGIRNA
Clave: 4

1. Se divide el texto en tantas partes como indica la clave (debe ser exacta):

   Grupos: ROFSACS CTAEMIS APRAOTS LIGIRNA

2. Se coloca cada grupo como una columna:

R	C	A	L
O	T	P	I
F	A	R	G
S	E	A	I
A	M	O	R
C	I	T	N
S	S	S	A

3. Se invierten las columnas:

R	C	A	L		L	A	C	R
O	T	P	I		I	P	T	O
F	A	R	G		G	R	A	F
S	E	A	I	⇒	I	A	E	S
A	M	O	R		R	O	M	A
C	I	T	N		N	T	I	C
S	S	S	A		A	S	S	S

4. Se concatena cada línea de la tabla para obtener el texto el claro:

   LACRIPTOGRAFIAESROMANTICASSS

1.2. Ambientes

La palabra *ambiente* procede del latín *ambiens*, *ambientis*, del verbo *ambere*: "rodear", "estar a ambos lados". En ciencia e ingeniería, un sistema es la parte del universo que se estudia y el ambiente es el resto del universo que está fuera de los límites de ese sistema.

En secciones anteriores se presentaron diversos tipos de agentes. En todos los casos, la relación que existe con el ambiente es siempre la misma: el agente quien ejerce acciones sobre el ambiente, el que, a su vez, aporta percepciones al agente.

1.2.1. Tipos de ambientes

Existen diversos tipos de ambientes. Las diferencias básicas son las siguientes:

1.2.1.1. Accesibles y no accesibles

Si el aparato sensorial de un agente le permite tener acceso al estado total de un ambiente, se dice que éste es accesible a tal agente. Un ambiente es realmente accesible si los sensores detectan todos los aspectos relevantes a la elección de una acción.

Los ambientes accesibles son cómodos, puesto que no es necesario que el agente mantenga un estado interno para estar al tanto de lo que sucede en el ambiente.

1.2.1.2. Deterministas y no deterministas

Si el estado siguiente de un ambiente se determina completamente mediante el estado actual y las acciones elegidas por los agentes, se dice que el ambiente es determinista. En principio, un agente no tiene por qué preocuparse sobre la incertidumbre en un ambiente accesible y determinista. Pero si el ambiente es inaccesible, entonces podría parecer que es no determinista.

1.2.1.3. Episódicos y no episódicos

En un ambiente episódico, la experiencia del agentese divide en "*episodios*". Cada episodio consta de un agente que percibe y actúa. La calidad de su actuación dependerá del episodio mismo, dado que los episodios subsecuentes no dependerán de las acciones producidas en episodios anteriores. Los ambientes episódicos son más sencillos puesto que el agente puede *pensar* por adelantado.

1.2.1.4. Estáticos y dinámicos

Si existe la posibilidad de que el ambiente sufra modificaciones mientras el agente se encuentra deliberando, se dice que tal ambiente se comporta en forma dinámica en relación con el agente; de lo contrario, se dice que es estático. Es más fácil trabajar con ambientes estáticos puesto que el

agente no tiene que observar lo que sucede en elmundo al mismo tiempo que decide sobre el curso de una acción, y tampoco tiene que preocuparse por el paso del tiempo. Si el ambiente no cambia con el paso del tiempo, pero sí se modifica la calificación asignada al desempeño de un agente, se dice que el ambiente es semidinámico.

1.2.1.5. Discretos y continuos

Si existe una cantidad limitada de percepciones y acciones distintas y claramente discernibles, se dice que el ambiente es discreto. El ajedrez es discreto: existe una cantidad fija de posibles jugadas en cada ronda. La conducción de un taxi escontinua: la velocidad y la ubicación del taxi y de los demás vehículos se extiende a través de un rango de valores continuos.

Figura 1.7: Ambiente complicado; aplica similar a lo hecho para las ondas gravitacionales

El caso más difícil se caracteriza por ser **inaccesible**, **no determinista**, **no episódico**, **dinámico** y **continuo**.

1.2.2. Programas de ambientes

El algoritmo genérico de ambiente ilustra la relación básica que existe entre agentes y ambientes. El simulador toma como entrada uno o más agentes y dispone de lo necesario para proporcionar las percepciones correctas una y otra vez a cada agente y así recibir como respuesta una acción. El simulador actualiza el ambiente tomando como base las acciones, y posiblemente otros procesos dinámicos del ambiente que no se consideran como agentes. De esta forma, el ambiente se define por el estado inicial y la función de actualización. Desde luego, se espera que un agente que opera en un simulador deberá ser capaz también de operar en un ambiente real que le proporcione el mismo tipo de percepciones y que acepte el mismo tipo de acciones.

Algoritmo 1.2 Simulacion de agentes en un ambiente

```
func simulaAmbiente (estado, agentes, terminacion)
  repetir
    para cada agente en los agentes
      percepciones [agente] ← obtenerPercepcion (agente, estado)
      acciones [agente] ← programa [agente] (percepciones [agente])
    estado ← actualizaAmbiente (acciones, agentes, estado)
    evaluaciones [] ← funcionDesempeño (evaluaciones, agentes, estado)
  hasta terminacion (estado)
  regresar (evaluaciones)
```

El procedimiento *simulaAmbiente* permite la adecuada ejecución de los agentes en un ambiente. En el caso de agentes simples, es suficiente con observar su conducta. Para obtener información más detallada acerca del desempeño de un agente, es necesario insertar algún tipo de código para medición de desempeño. La $funcionDesempeño$ realiza esta tarea; aplica una medición de desempeño a cada agente y produce una lista de $calificaciones$ obtenidas, con esta lista se está al tanto de la calificación de cada uno de los agentes. La función *terminacion* evalúa si el estado actual cumple con las características necesarias para terminar la simulación.

En general, lo que se use como medición de desempeño dependerá de la secuencia total de estados ambientales generados durante la operación del programa. Sin embargo, por lo general, la medida del desempeño se realiza a través de una simple acumulación, recurriendo ya sea a una adición, a un promedio o tomando un máximo.

Tarea:

♢ Desarrollar con más detalle el algoritmo de las funciones

- *obtenerPercepcion*
- *programa*
- *actualizaAmbiente*
- $funcionDesempeño$

  para un sistema en el que conviven dos robots seguidores de línea tipo *Mo* (limpiador); las líneas pueden cruzarse en diferentes puntos. Puedes agregar sensores a cada agente.

Figura 1.8: ¿Limpiar o seguir la línea?

*