Cálculo Situacional: Tron

Rojas Fredini Emmanuel
Bedrij Walter

1 Julio, 2011

Resumen

Con el objeto de estudiar la inteligencia artificial se propone el diseño de un agente basado en calculo situacional. Se analiza la representación del mundo y el modelado del problema usando cálculo situacional.

1. Enunciado

Suponga que se desea emular el juego Tron, en el cual Kevin Flynn debe guiar su "light cycle" azul en un recuadro de 10x10 celdas, mientras debe evitar chocarse los muros y los rastros (muros de luz) creados por él mismo y por su oponente. Kevin debe maniobrar rápidamente para forzar a su oponente a chocar contra los muros.

Cada vez que Kevin realiza un movimiento, su enemigo también lo hace. Kevin y su enemigo solo pueden moverse vertical u horizontalmente. ¿Puedes ayudar a Kevin a vencer a su oponente utilizando búsqueda?

2. Representación del mundo

El mundo es una grilla cuadrada de 10×10 casillas. Cada casilla tiene información acerca del tipo de objeto que la ocupa: *tron*, *wall* o *libre*. Los competidores se ubican en esquinas opuestas y el movimiento es alternado.

2.1. Caracterización del ambiente

Cada agente sólo puede observar una porción limitada del mundo, las celdas adyacentes a su posición.

Las acciones del agente son determinísticas. Pudiendo elegir una de las casillas adyacentes. Si la casilla de destino se encuentre *vacía*, el agente se mueve a esa posición, caso contrario muere.

El objetivo es sobrevivir más tiempo que el contrincante.

2.2. Comunicación entre el ambiente y el agente

El ambiente impacta los sentidos del agente enviándole una copia fiel de su entorno próximo. Como respuesta a esto el agente devuelve una acción posible. El ambiente entonces ejecuta la acción, modificando el mundo.
La tupla de percepciones es:

$$P = [A, B, C, D]$$

Donde:

- $A = 1$ *sii* la derecha de Tron es inaccesible
- $B = 1$ *sii* la izquierda de Tron es inaccesible
- $C = 1$ *sii* arriba de Tron es inaccesible
- $D = 1$ *sii* abajo de Tron es inaccesible

3. Agente

El conocimiento del mundo se limita a las dimensiones del mismo, y a las posiciones iniciales tanto del agente como de su enemigo.

El agente utiliza las percepciones para identificar las celdas bloqueadas, información que será usada en la toma de decisiones. Luego de elegir una acción a efectuar, deberá actualizar su situación en correspondencia.

Se ejecuta entonces la acción sobre el ambiente. Si no se pudo proveer de una acción válida, el agente es descalificado.

3.1. Reglas de Diagnóstico

Estas se usan para marcar las celdas ocupadas, es decir, aquellas por las que el Tron ya no podrá pasar.

Si el contenido de una celda es distinto de *libre*, entonces se marcará como *ocupado* en la situación actual.
Las reglas de diagnostico son:

$$\forall S \quad P = [1, \_, \_, \_] \rightarrow ocupado\,(X+1, Y\,, S)$$
$$\forall S \quad P = [\_, 1, \_, \_] \rightarrow ocupado\,(X-1, Y\,, S)$$
$$\forall S \quad P = [\_, \_, 1, \_] \rightarrow ocupado\,(X, Y+1\,, S)$$
$$\forall S \quad P = [\_, \_, \_, 1] \rightarrow ocupado\,(X, Y-1\,, S)$$

3.2. Aciones

Las acciones realizables por el agene son las de desplzamiento. Estas seran las que permitan, si el desplazamiento es entre celdas adyacentes, moverse hacia la dercha, izquierda, abajo o arriba.
Las definiciones son:

$$\neg ocupado(X+1, Y) \wedge pos(X, Y) \rightarrow IrDerecha \rightarrow pos(X+1, Y) \wedge ocupado(X, Y)$$

$$\neg ocupado(X-1, Y) \wedge pos(X, Y) \rightarrow IrIzquierda \rightarrow pos(X-1, Y) \wedge ocupado(X, Y)$$

$$\neg ocupado(X, Y+1) \wedge pos(X, Y) \rightarrow IrArriba \rightarrow pos(X, Y+1) \wedge ocupado(X, Y)$$

$$\neg ocupado(X, Y-1) \wedge pos(X, Y) \rightarrow IrAbajo \rightarrow pos(X, Y-1) \wedge ocupado(X, Y)$$

3.3. Decisión

Con el objeto de establecer la acción a tomar se estableció un *ranking* de acciones en función de la situación. Se evalúan en orden hasta obtener una acción válida. Se tienen entonces tres categorías:

1. Buscar la mayor cantidad de casillas libre:
   Mediante el algoritmo de *fill flood* se intenta predecir la cantidad máxima de movimientos siguientes según qué dirección se elija. En caso de que no haya una dirección preferencial esta prueba falla. Podemos observar en la Figura 1 un caso cuya situación corresponde a esta categoria y es resuelto el ranking en función al algoritmo fill flood. Se ejecuta fill flood en las direciones izquierda, inferior y superior, como se ve en el bloque central, luego evaluando que la cantidad de casillas rellenadas por fill flood depende de la direcion en la que se lo ejecuto, en el grafico podemos apreciar que la region pintada azul es la mayor, entonces elgimos la direcion superior por tener la mayor cantidad de casillas "rellenadas".

2. Buscar el centro:
   Trata de conducir al tron hacia el centro del tablero (en realidad la celda $[5, 6]$), se da preferencia a los movimientos horizontales. En la Figura 2 hay un caso de una situacion como esta, vemos que como en todo momento fill flood dara en todas las direciones igual cantidad de casillas rellenas, luego la situacion corresponde a esta categoria. Tron recorre hacia el centro($[5, 6]$), primero realizando movimientos horizontales y luego verticales.

3. Aleatorio:
   Elige de forma aleatoria entre las casillas adyacentes que se encuentran libres.

3.4. Estado Sucesor

Se identificaron como dinámicas a la posición que ocupa el Tron, y a la lista de casillas ocupadas. Es decir que tendremos dos estados sucesor que mantienen el estado interno de *pos* y de *ocupado*.

Según la acción que se decidió se actualiza a la siguiente situación

- La posición del agente será la casilla adyacente correspondiente a la acción elegida

- La celda donde se encontraba el agente, pasará a marcarse como ocupada
- Una celda ocupada continuará ocupada

La reglas del estado sucesor de pos es:

$$\begin{gathered}\text{pos}(X, Y, resultado\,(A, S)) \leftrightarrow \\ \{(A = IrDerecha \wedge pos\,(X-1, Y, S)) \vee \\ (A = IrIzquierda \wedge pos\,(X+1, Y, S)) \vee \\ (A = IrArriba \wedge pos\,(X, Y-1, S)) \vee \\ (A = IrAbajo \wedge pos\,(X, Y+1, S))\end{gathered}$$

4. Figuras

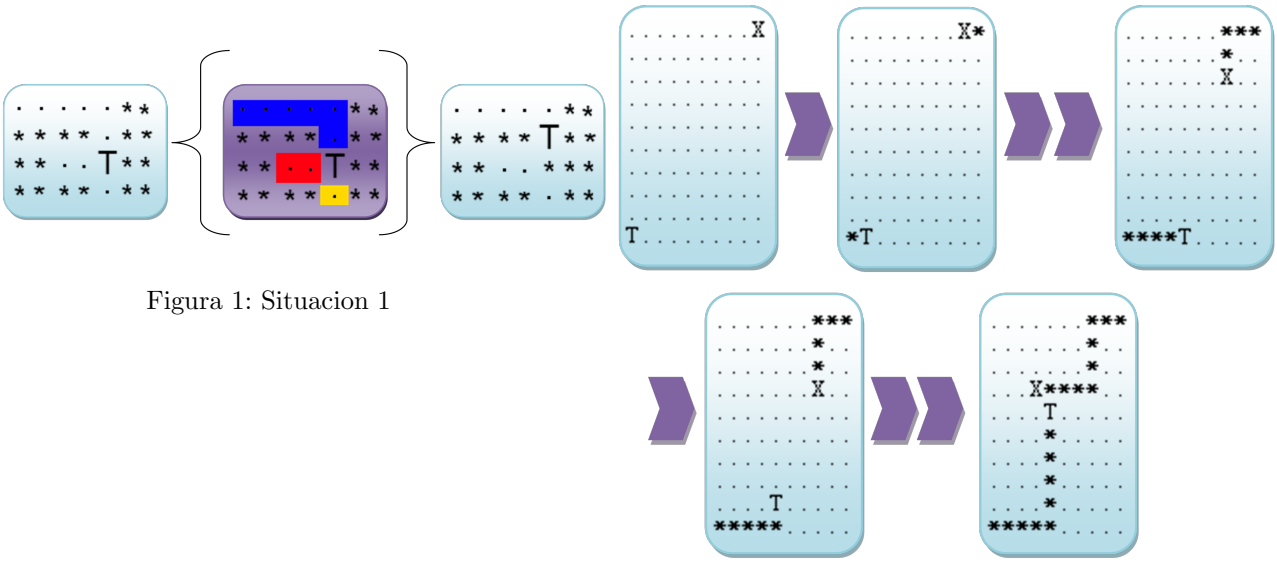

Figura 1: Situacion 1

Figura 2: Situacion 2