El Cluedo como problema de lógica de segundo orden

alvaro

2017-08-29 mar

El Cluedo es un juego infantil cuyas normas pueden resumirse en las siguientes:

- Las cartas tienen tres categorías: **personajes**, **herramientas** y **lugar**.
- Se oculta una carta de cada tipo en un sobre.
- El resto de cartas se reparte entre los jugadores
- Los jugadores deben deducir qué cartas están ocultas mediante sus preguntas.
- En cada turno, un jugador hace una *predicción* de tres cartas
  - El jugador a su derecha debe mostrar una de esas cartas al jugador que ha preguntado (si es poseedor de ellas).
  - Si no posee ninguna, debe admitirlo y se hace la misma pregunta al siguiente jugador, hasta que algún jugador muestra una carta, o se llega de nuevo al jugador que hizo la pregunta.
- Cuando un jugador crea tener la combinación oculta en el sobre, debe hacer una *acusación*, diciendo en alto esas tres cartas.
  - Comprobará él mismo si ha acertado con las tres cartas, en cuyo caso ha ganado el juego.
  - Si no ha acertado, seguirá jugando, pero sin la posibilidad de ganar.
- En las normas completas del juego se utilizan también dados y fichas sobre un tablero, pero esas normas no son importantes para el problema de lógica.

Con el material del juego se incluyen unas plantillas de ayuda a los jugadores, donde hay una casilla por cada posible carta. En las instrucciones se indica que se debe apuntar cada carta que se posea, o que se descubra mediante preguntas que otros jugadores tienen. Al final, las tres cartas que queden sin apuntar serán las cartas ocultas.

Aunque este enfoque es correcto, es mejorable utilizando el resto de información disponible para cada jugador. Como se verá posteriormente, cada respuesta a una pregunta puede representarse como una fórmula lógica de primer orden, permite realizar muchas más deducciones.

Variables lógicas

Supongamos que hay J jugadores, P cartas de personaje, H cartas de herramientas y L cartas de lugar. En total habrá $M = J + P + H$ cartas. El sobre puede considerarse un jugador más (por ejemplo, el número 0). Para cada carta y jugador tendremos en cuenta una variable que indica si ese jugador posee esa carta, c_{ij}, con los subíndices variando entre los posibles valores de jugadores y cartas ($i \in M, j \in J$).

En el resto de la discusión, supondremos que el valor lógico *verdadero* equivale a 1 al utilizarse con operadores numéricos, y que $falso$ equivale a 0.

La estructura del problema impone varias condiciones sobre las variables c_{ij}:

- Si un jugador posee una carta, el resto de jugadores no pueden tenerla: $\sum_j c_{ij} = 1, \forall i \in M$
- Cada jugador tiene un número de cartas n_j, por lo que $\sum_i c_{ij} = n_j$. Por ejemplo, en el sobre (jugador 0) siembre hay tres cartas, por lo que $\sum_i c_{i0} = 3$.
- En el sobre hay una carta de cada tipo. Por tanto $\sum_{i \in J} c_{i0} = 1$, $\sum_{i \in P} c_{i0} = 1$, $\sum_{i \in L} c_{i0} = 1$
- Si el jugador j nos muestra la carta i, podemos asignar el valor $c_{ij} = true$.
- Si el jugador j reconoce no tener ninguna de las cartas x, y, z podemos asignar los valores $c_{xj} = false$, $c_{yj} = false$, $c_{zj} = false$
- Si el jugador j reconoce tener alguna de las cartas x, y, z a otro jugador, podemos deducir que $(c_{xj} \vee c_{yj} \vee c_{zj}) = true$

- Si un jugador hace una *acusación* con cartas x, y, z, pero no acierta, podemos deducir que $\neg(c_{x0} \wedge c_{y0} \wedge c_{z0}) = true$.

Puede verse que cada una de estas ecuaciones es una **restricción** que deben cumplir las variables c_{ij}. Las restricciones impiden que todos los valores sean válidos.

Expresiones lógicas

En el punto anterior, se vio que puede modelarse el problema utilizando las funciones lógicas $\neg$ (negación), $\wedge$ (conjunción), $\vee$ (disyunción), y una función que cuente si el número de variables con valor $true$ es un número determinado n, que llamaremos $V_n()$.

De estas funciones, sólo $\neg$ y $V_n()$ son primitivas:

- $a \wedge b$ puede expresarse como $V_2(a, b)$
- $a \vee b$ puede expresarse como $\neg(\neg a \wedge \neg b)$ (leyes de Morgan)

Programación por restricciones

La programación por restricciones es una técnica para dar valores a variables de una forma coherente con las restricciones impuestas entre ellas.

Los elementos de un sistema de programación por restricciones son:

- Las variables (nuestras c_{ij})
- Las restricciones entre ellas
- Una forma de **propagar** las restricciones: aplicar las consecuencias de los valores de variables y restricciones en los posibles valores de otras variables
- Un sistema de **prueba y error**: cuando la propagación no es suficiente para dar valor a todas las variables, se realiza una búsqueda entre los posibles valores hasta encontrar un conjunto coherente.

Variables

Las variables tienen un dominio inicial, que es el conjunto de sus valores posibles. Como son variables lógicas, este dominio es $\{true, false\}$. Es

importante señalar que durante la propagación y la búsqueda este dominio nunca se amplía, sino que se reduce.

Si una variable tiene solo un valor en su dominio, se considera que ese es su valor, y la variable está **definida**.

Si alguna variable llega a tener un dominio sin posibles valores (dominio vacío), es porque dicha variable no puede tener ningún valor posible, por lo que las restricciones y los dominios de las demás variables no son coherentes.

Expresiones

Las expresiones pueden verse también como variables. Por ejemplo, si el dominio de a y b es $\{true, false\}$, $a \wedge b$ tiene el mismo dominio. Pero si el dominio de b se reduce a $\{false\}$, el dominio de $a \wedge b$ también se reduce (ya no puede ser $true$). Esto hace que una *expresión* pueda utilizarse como una variable más.

Restricciones

Una restricción es una expresión a la que se fija un valor. Por ejemplo, $a \wedge b$ es una expresión, pero $a \wedge b = false$ se convierte en una restricción.

Propagación

En la propagación se extraen consecuencias de las expresiones y los dominios de variables. Basta con estudiar $\neg$ y $V_n()$, puesto que las demás pueden basarse en estas.

Pueden distinguirse dos direcciones en la propagación: desde los elementos de una expresión hacia la expresión (hacia *arriba*), y desde la expresión hacia sus elementos (hacia *abajo*)

Propagación hacia *arriba*

- Si se elimina $true$ de a, puede eliminarse $false$ de $\neg a$.
- Si se elimina $false$ de a, puede eliminarse $true$ de $\neg a$.
- Para $V_n(a_1, a_2, \ldots, a_m)$
  - Si hay más de n variables definidas a $true$, la expresión es $false$ (se elimina $true$ del dominio de la expresión)
  - Si hay más de $m - n$ variables definidas a $false$, la expresión es $false$ (se elimina $true$ del dominio de la expresión)

- Si están definidas todas las variables y hay n a $true$, se elimina $false$ del dominio de la expresión.

Propagación hacia *abajo*

- Si se elimina $true$ de $\neg a$, puede eliminarse $false$ de a.
- Si se elimina $false$ de $\neg a$, puede eliminarse $true$ de a.
- Si $V_n(a_1, a_2, \ldots, a_m)$ es $false$ y todas las variables están definidas menos a_i
  - Si hay $n-1$ variables $true$, entonces a_i es $false$ (se le quita $true$)
  - Si hay n variables a $true$, entonces a_i es $true$ (se le quita $false$)
- Si $V_n(a_1, a_2, \ldots, a_m)$ es $true$ y todas las variables están definidas menos l de ellas:
  - Si hay n variables $true$, entonces todas las l variables sin definir son $false$ (se les quita $true$)
  - Si hay $n - l$ variables a $true$, entonces todas las l variables son $true$ (se les quita $false$)

*