IIC1253 - Sección 1 - Segundo Semestre 2014
Profesor: Marcelo Arenas
Ayudantes: Matías San Martín (*masanmartin@uc.cl*) - Martín Muñoz (*mmunos@uc.cl*)

Ayudantía 4.

Modelación con lógica proposicional.

Problema 1. Considere un laberinto cuadrado como el siguiente:

×	×	×	×	×	×
×	⊢				×
×		×	×		×
×		×			×
×				○	×
×	×	×	×	×	×

donde las casillas $\boxed{\times}$ representan muros y las casillas $\boxed{}$ representan pasillos libres.

Considere a un jugador dentro del laberinto, que comienza en la casilla $\boxed{\vdash}$ y que quiere llegar a la casilla $\boxed{\circ}$ en no más de T_n pasos con n la dimensión del tablero y T_n es de tamaño polinomial respecto a n. Para definir reglas razonables, se tiene que el jugador no puede ni pisar ni atravesar muros y sus movimientos solo pueden ser horizontales o verticales (además de cosas obvias que van implícitas, como que siempre está en el laberinto o que solo puede estar en un solo lugar al mismo tiempo, entre otros).

Claramente, podemos definir una matriz M de $n \times n$ que contenga toda la información sobre muros y pasillos del laberinto. Ya que dado un tablero podemos definir:

$$M_{i,j} = \begin{cases} 1 & \text{si hay un muro en la celda } (i,j) \\ 0 & \text{si no} \end{cases}, \qquad \text{para } i,j \in \{1,\ldots,n\}.$$

Asumimos que todos los bordes del tablero están rodeados por muros y que el jugador siempre comienza en el casillero de la esquina superior izquierda, intentando de llegar a la casilla final que se encuentra en la esquina inferior derecha. Tanto la celda inicial como la celda final nunca tienen un muro.

Dado un tablero y usando la familia de variables proposicionales $\{m_{i,j} : i,j \in \{1,\ldots,n\}\}$ para representar el tablero y $\{p_{i,j,t} : i,j \in \{1,\ldots,n\}, t \in \{0,\ldots,T_n\}\}$ para poder describir el comportamiento del jugador en el tablero a través del tiempo, construya una fórmula $\varphi \in L(P)$ tal que φ sea sastisfacible si y solo si este juego es factible, consistente y tiene solución; es decir, que el jugador puede llegar a la casilla final en no más de T_n pasos y se respeten todas las reglas.

¿En qué nos ayuda construir esta fórmula φ para poder resolver el problema inicial del laberinto?

Solución. Dado el tablero, tenemos la matriz M que la describe y tal como recomiendan en el enunciado, ocuparemos las siguientes variables proposicionales:

$m_{i,j}$: es verdad si hay un muro en la casilla (i,j), es decir, $M_{i,j} = 1$ con $i,j \in \{1,\ldots,n\}$.

$p_{i,j,t}$: es verdad si en el instante t el jugador está en la casilla (i,j) para $i,j \in \{1,\ldots,n\}$ y $t \in \{0,\ldots,T_n\}$.

Buscamos representar en lógica proposicional las siguientes restricciones:

- El tablero está bien definido, es decir, se sabe si hay muros o pasillos para cada celda. La fórmula sería algo como:

$$\varphi_0 = \bigwedge_{(i,j):M_{i,j}=1} m_{i,j} \ \wedge \bigwedge_{(i,j):M_{i,j}=0} \neg m_{i,j}.$$

- Cuando comienza el juego ($t = 0$), el jugador debe estar en su celda inicial, es decir, la celda $(1, 1)$. Luego, la fórmula sería:

$$\varphi_1 = P_{1,1,0}.$$

- El jugador no puede estar nunca en una celda donde hay un muro. La fórmula sería:

$$\varphi_2 = \bigwedge_{i=1}^{n} \bigwedge_{j=1}^{n} \left(m_{i,j} \rightarrow \bigwedge_{t=0}^{T_n} \neg p_{i,j,t} \right).$$

- En todo momento, el jugador debe estar en una única celda del laberinto.

$$\varphi_3 = \bigwedge_{t=0}^{T_n} \bigwedge_{i=1}^{n} \bigwedge_{j=1}^{n} \left(p_{i,j,t} \rightarrow \left(\bigwedge_{k=1}^{n} \bigwedge_{l=1}^{n} \neg p_{k,l,t} \right) \right).$$

- Dado un instante t, en el próximo instante el jugador puede estar en la celda inmediatamente de arriba, de abajo, de la izquierda, de la derecha o se mantiene en la misma celda que estaba. Y claramente, esta celda del siguiente instante, no debe ser un muro.

$$\varphi_4 = \bigwedge_{t=0}^{T_n-1} \bigwedge_{i=1}^{n} \bigwedge_{j=1}^{n} \big(p_{i,j,t} \rightarrow (p_{i,j,t+1} \vee p_{i+1,j,t+1} \vee p_{i,j+1,t} \vee p_{i-1,j,t+1} \vee p_{i,j-1,t}) \big).$$

- Eventualmente debe llegar a la casilla final, que correspondería a la casilla $(n - 1, n - 1)$.

$$\varphi_5 = \bigvee_{t=1}^{T_n} p_{n-1,n-1,t}.$$

Y definimos $\varphi = \bigwedge_{i=0}^{5} \varphi_i$ como la fórmula que cumple lo pedido.

¿Qué representa una valuación para φ? ¿Qué información nos entrega respecto al problema original? ¿Cómo se demuestra que efectivamente tenemos que φ es satisfacible si y solo si representa el problema anterior?

Problema 2. Una grilla $\mathcal{G}$ de $(n+1)\times(n+1)$ es un conjunto de celdas que se puede definir como:

$$\mathcal{G} = \big\{(i,j) : i,j \in \{0,\ldots,n\}\big\}.$$

Una curva Γ_1 que va de $(0,0)$ a (n,n) es un subconjunto de $\mathcal{G}$ tal que $(0,0)\in\Gamma_1$, $(n,n)\in\Gamma_1$ y si $(i,j)\in\Gamma_1\backslash\{(n,n)\}$, entonces o bien $(i+1,j)\in\Gamma_1$ o bien $(i,j+1)\in\Gamma_1$ (pero no ambos). De la misma forma, definimos la curva Γ_2 que va de $(0,n)$ a $(n,0)$ como el subconjunto de $\mathcal{G}$ tal que $(0,n)\in\Gamma_2$, $(n,0)\in\Gamma_2$ y si $(i,j)\in\Gamma_2\backslash\{(n,0)\}$ entonces o bien $(i+1,j)\in\Gamma_2$ o bien $(i,j-1)\in\Gamma_2$ (pero nuevamente, no ambos). Claramente, ambas curvas se intersectan en algún punto.

Dada una grilla $\mathcal{G}$ de $(n+1)\times(n+1)$, construya una fórmula $\varphi\in L(P)$ que sea sastisfacible si y solo si modela al problema descrito arriba. Esto significa si tenemos una curva Γ_1 que va de $(0,0)$ a (n,n) y una curva Γ_2 que va de $(0,n)$ a $(n,0)$, entonces $\Gamma_1\cap\Gamma_2\neq\emptyset$, es decir, ambas curvas se intersectan en algún punto.

Solución. Para modelar el problema, usamos las siguientes variables proposicionales:

$$\begin{array}{rcl} p_{i,j} & : & \text{es verdad cuando } (i,j)\in\Gamma_1 \text{ para } i,j\in\{0,\ldots,n\}.\\ q_{i,j} & : & \text{es verdad cuando } (i,j)\in\Gamma_2 \text{ para } i,j\in\{0,\ldots,n\}.\end{array}$$

Para definir los puntos de la curva Γ_1 definimos la fórmula φ_{Γ_1} como sigue:

$$\begin{aligned}\varphi_{\Gamma_1} \;=\; & \bigwedge_{i=0}^{n-1}\bigwedge_{j=0}^{n-1}\Big(p_{i,j}\to\big((p_{i+1,j}\wedge\neg p_{i,j+1})\vee(\neg p_{i+1,j}\wedge p_{i,j+1})\big)\Big)\wedge\\ & \bigwedge_{i=0}^{n-1}\big(p_{i,n}\to p_{i+1,n}\big)\;\wedge\;\bigwedge_{j=0}^{n-1}\big(p_{n,j}\to p_{n,j+1}\big)\wedge p_{0,0}\;\wedge\;p_{n,n}.\end{aligned}$$

De la misma forma, definimos φ_{Γ_2}:

$$\begin{aligned}\varphi_{\Gamma_1} \;=\; & \bigwedge_{i=0}^{n-1}\bigwedge_{j=0}^{n-1}\Big(q_{i,j}\to\big((q_{i+1,j}\wedge\neg q_{i,j-1})\vee(\neg q_{i+1,j}\wedge q_{i,j-1})\big)\Big)\wedge\\ & \bigwedge_{i=0}^{n-1}\big(q_{i,0}\to q_{i+1,0}\big)\;\wedge\;\bigwedge_{j=1}^{n}\big(q_{n,j}\to q_{n,j-1}\big)\wedge q_{0,n}\;\wedge\;q_{n,0}.\end{aligned}$$

Para decir que se intersectan, escribimos la fórmula φ_I como:

$$\varphi_I = \bigvee_{i=0}^{n}\bigvee_{i=0}^{n}(p_{i,j}\wedge q_{i,j}).$$

Por último, la fórmula que representa la situación y cumple con lo pedido corresponde a:

$$\varphi = (\varphi_{\Gamma_1}\wedge\varphi_{\Gamma_2})\to\varphi_I.$$

Es claro que con una valuación de φ podemos construir las curvas y verificar que se intersectan. Del mismo modo, si nos dan las curvas, podemos encontrar de manera directa una valuación que satisface a φ.

Problema 3. Dado dos grafos, $G_1 = (V_1, E_1)$ y $G_2 = (V_2, E_2)$, se dice que un *homomorfismo* de G_1 en G_2 es una función $h : V_1 \to V_2$ tal que para todo $a, b \in V_1$ se tiene que si $(a, b) \in E_1$, entonces $(h(a), h(b)) \in E_2$. Se dice que G_1 es *homomórfico* a G_2. Encuentre un ejemplo de esto.

Defina un algoritmo eficiente tal que dado dos grafos $G_1 = (V_1, E_1)$ y $G_2 = (V_2, E_2)$ construya una fórmula $\varphi \in L(P)$ tal que φ es satisfacible si y solo si existe un homomorfismo h de G_1 en G_2.

Si h es un homomorfismo de G_1 en G_2, pero h no es una función sobre, se dice que es un *homomorfismo propio*. Si existe un homomorfismo propio de G en sí mismo, se dice que G es *non-lean*.

Pensando en la fórmula φ ya encontrada, encuentre un algoritmo eficiente tal que dado un grafo $G = (V, E)$ que construya una fórmula $\psi \in L(P)$ tal que G es non-lean si y solo si ψ es satisfacible.

Solución. Usamos las siguientes variables proposicionales:

$$\begin{array}{lcl} a_{i,j} & : & \text{es verdad si } (i,j) \in E_1, \text{ para } i, j \in V_1. \\ e_{k,\ell} & : & \text{es verdad si } (k,\ell) \in E_2, \text{ para } k, l \in V_2. \\ h_{i,k} & : & \text{es verdad si se define } h(i) = k, \text{ para } i \in V_1 \text{ y } k \in V_2. \end{array}$$

Usamos φ_1 y φ_2 para definir los grafos G_1 y G_2 respectivamente:

$$\varphi_1 = \bigwedge_{(i,j)\in E_1} a_{i,j} \ \wedge \bigwedge_{(i,j)\notin E_1} \neg a_{i,j}, \qquad \varphi_2 = \bigwedge_{(k,\ell)\in E_2} e_{k,\ell} \ \wedge \bigwedge_{(k,\ell)\notin E_2} \neg e_{k,\ell}.$$

Controlamos que el h que buscamos representar sea una función bien definida:

$$\varphi_f = \bigwedge_{i\in V_1} \left(\bigvee_{k\in V_2} \left(h_{i,k} \ \wedge \bigwedge_{\substack{\ell\in V_2 \\ \ell\neq k}} \neg h_{i,\ell} \right) \right).$$

Y por último, nos queda controlar que preserva los arcos. De esta forma, nos queda

$$\varphi_H = \bigwedge_{i\in V_1} \bigwedge_{j\in V_1} \left(a_{i,j} \to \left(\bigwedge_{k\in V_2} \bigwedge_{\ell\in V_2} (h_{i,k} \wedge h_{j,\ell}) \to e_{k,\ell} \right) \right).$$

Claramente, la fórmula buscada φ está dada por: $\varphi = \varphi_1 \wedge \varphi_2 \wedge \varphi_f \wedge \varphi_H$. Si tenemos una valuación para φ, podemos construir el homomomorfismo h entre G_1 y G_2. Y de la misma forma, dado un homomorfismo h, encontramos una valuación que satisface a φ. Esto implica que está bien construido.

Para la parte de homomorfismo propio, decimos que existe un vértice $v \in V_2$ tal que $h(i) \neq v$ para todo $i \in V_1$. Luego, la fórmula que lo define estaría dada por:

$$\varphi_s = \bigvee_{v\in V_2} \left(\bigwedge_{i\in V_1} \neg h_{i,v} \right).$$

Para lo de non-lean, debiese bastar usar las mismas fórmulas ya definidas, pero considerando que $G_1 = G_2 = G$, por lo que hay que revisar los conjuntos y los dominios de las variables.

Problema 4. Dado un grafo $G = (V, E)$ decimos que G contiene un *ciclo Hamiltoniano* si:

- Existen vértices $v_1, v_2, \ldots, v_n \in V$ tal que $(v_i, v_{i+1}) \in E$ para todo $i \in \{1, \ldots, n-1\}$. Es decir, que hay un *camino* entre los vértices v_1 y v_n.
- Se tiene que todos los vértices definidos en el punto anterior son distintos. Es decir, que $v_i \neq v_j$ para todo i distinto de j que están en $\{1, \ldots, n\}$. Notar que esto implica que se pasa una sola vez por cada uno de los vértices del camino.
- V es un conjunto de n elementos, por lo que lo podemos escribir como $V = \{a_1, a_2, \ldots, a_n\}$. Notar que como el camino simple definido en los puntos anteriores es de largo n, se tiene que los vértices $\{v_i\}_{i=1}^n$ corresponden a un reordenamiento de los términos de los vértices $\{a_j\}_{j=1}^n$.
- Se tiene que $(v_n, v_1) \in E$. De esta forma, el camino se convierte en un ciclo, ya que vuelve al mismo punto inicial.

Defina un algoritmo eficiente que dado un grafo $G = (V, E)$, sea capaz de construir una fórmula $\varphi \in L(P)$ tal que el grafo G tiene un ciclo Hamiltoniano si y solo si φ es satisfacible.

Extendemos la idea recién descrita y dado $k \in \mathbb{N}$, decimos que un grafo G contiene un *k-ciclo Hamiltoniano* si existe un conjunto $\{e_1, e_2, \ldots, e_n\} \subset V \times V$ tal que el grafo $G' = (V, E \cup \{e_1, \ldots, e_n\})$ contiene un ciclo hamiltoniano.

Describa un algoritmo eficiente que para un grafo $G = (V, E)$ y un natural k construya una fórmula $\psi \in L(P)$ tal que ψ es satisfacible si y solo si G contiene un k-ciclo Hamiltoniano.

Solución. Nuestra primera intuición es hacer una gran disyunción que pase por todas las combinaciones de los caminos simples entre los vértices, que cumplan las restricciones pedidas para que en alguna de ellas, corresponda a un ciclo Hamiltoniano. El problema con resolver el problema así es que nos quedará una fórmula de tamaño exponencial al grafo dado, por lo que no podrá ser un algoritmo eficiente, que es lo pedido por este problema.

Notamos que como en el ciclo siempre se vuelve al vértice original y pasa por todos los vértices del grafo, podemos partir con cualquiera y mantener el ciclo. Esto nos motiva a definir las siguientes variables proposicionales:

$$\begin{aligned} a_{i,j} &: \text{ es verdad si } (a_i, a_j) \in E, \text{ para } i, j \in \{1, \ldots, n\}. \\ c_{i,p} &: \text{ es verdad si el vértice } a_i \in V \text{ está en la posición } p \text{ del ciclo Hamiltoniano,} \\ &\quad \text{con } p \in \{1, \ldots, n\} \text{ e } i \in \{1, \ldots, n\}. \end{aligned}$$

Definimos el grafo con la fórmula:

$$\varphi_G = \bigwedge_{(i,j) \in E} a_{i,j} \ \wedge \bigwedge_{(i,j) \notin E} \neg a_{i,j}.$$

Como explicamos antes, podemos suponer que parte en el vértice a_1 de manera arbitraria sin que nos traiga problemas. Luego, debemos tener $\varphi_0 = c_{1,1}$. De aquí construimos las siguientes restricciones:

- Cada vértice debe estar en una única posición del ciclo. Luego, la fórmula sería:

$$\varphi_1 = \bigwedge_{i=1}^{n} \left(\bigvee_{p=1}^{n} \left(c_{i,p} \wedge \bigwedge_{\substack{1 \leq q \leq n \\ q \neq p}} \neg c_{i,q} \right) \right).$$

- Si tengo dos vértices que están seguidos en el ciclo, entonces deben estar conectados en el grafo.

$$\varphi_2 = \bigwedge_{i=1}^{n} \bigwedge_{j=1}^{n} \bigwedge_{p=1}^{n-1} \big((c_{i,p} \wedge c_{j,p+1}) \to a_{i,j} \big).$$

- El vértice que está en la última posición debe estar conectado al primero del camino, que por suposición era el vértice a_1.

$$\varphi_3 = \bigwedge_{i=2}^{n} \big(c_{i,n} \to a_{i,1} \big).$$

Y con eso debiese estar listo. Basta tomar $\varphi = \varphi_G \wedge \bigwedge_{i=0}^{3} \varphi_i$.

Esta fórmula es satisfacible si y solo si G tiene un ciclo Hamiltoniano, ya que dada una valuación para φ tendremos que las variables $c_{i,p}$ serán verdaderas para los vértices que forman un ciclo Hamiltoniano en el grafo G, y de la misma forma, a partir del ciclo Hamiltoniano en G se puede construir una valuación que satisfaga a φ.

El resto del problema se deja como propuesto.