Práctica 3

Lógica Computacional, 2018-2
Facultad de Ciencias, UNAM

Noé Salomón Hernández Sánchez
no.hernan@gmail.com

María del Carmen Sánchez Almanza
carmensanchez@ciencias.unam.mx

Albert Manuel Orozco Camacho
alorozco53@ciencias.unam.mx

1 de mayo de 2018

1. Objetivo

Los asistentes automáticos de demostraciones nos brindan un paradigma especial de lo que implica escribir una prueba formal: en cada momento nos planteamos un objetivo al que hemos de llegar mediante argumentos bien fundamentados. Para este curso, el algoritmo de verificación de fórmulas derivado de las reglas de deducción natural motivan al uso de una herramienta como `Coq`.

Mediante el asistente de pruebas francés exploraremos otro de los métodos de resolución del problema de satisfacibilidad, cuya eficiencia depende de la destreza del programador entorno a su uso. A pesar de que restringiremos el uso de `Coq` a fórmulas de los cálculos vistos en el curso, es importante notar y destacar su uso para tareas como *verificación de software* y *razonamiento automatizado*, las cuales están adquiriendo mayor preponderancia en el mundo computacional moderno.

2. Ejercicios

El esqueleto de código de la práctica se encuentra en `https://github.com/alorozco53/LabLogComp-2018-2/tree/pract3/src`. Asegúrese de haber instalado `CoqIDE` o `ProofGeneral` en su computadora, mediante el tutorial presentado en clase.

2.1. Correctud de fórmulas

Para las siguientes fórmulas, dé una demostración basada en tácticas de Coq.

1. $p \to q, q \to r \vee s, \neg s, p \vdash r$.

2. $\vdash (p \to q) \to (p \vee r \to q \vee r)$. *Consejo*: dé una demostración *por reducción al absurdo* mediante el uso de una doble negación del consecuente de la fórmula. Es decir, recuerde que para toda fórmula A y B,

$$A \to B \equiv A \to \neg\neg B.$$

3. $$\begin{array}{l} \exists x Q(x) \\ \forall x(Q(x) \wedge \exists y P(y) \to Q(f(x))) \\ \forall z(Q(z) \to Q(g(z))) \\ \hline \therefore P(b) \to \exists w Q(f(g(w))) \end{array}$$

   Recuerde que se debe de definir un universo de discurso (`D`) y que los predicados se modelan mediante un tipo que toma cuantos valores indique su aridad para "devolver" algo de tipo `Prop`. Por ejemplo, el predicado P tendría tipo `D -> Prop`. Asímismo, no olvide definir la constante b como `Variable` de tipo `D` y las variables f y g ambas como `Variable` de tipo `D -> D`.

4. $$\begin{array}{l} \exists x(\neg P(x) \vee Q(x)) \\ \hline \therefore \exists x \neg(P(x) \wedge \neg Q(x)) \end{array}$$

5. $$\begin{array}{l} \forall x(P(x) \to Q(x)) \\ \forall y(R(b) \vee Q(y) \to S(a)) \\ \hline \therefore \forall z(P(z) \to \exists y S(y)) \end{array}$$

2.2. Traducción y verificación

Demostrar la correctud de los siguientes argumentos.

6. Usando exclusivamente la signatura $\{F^{(1)}, V^{(1)}, B^{(1)}, E^{(1)}, A^{(1)}, L^{(1)}, S^{(1)}, C^{(2)}\}$ con significados *ser feliz, ser vampiro, ser bruja, emborracharse, pernoctar en cementerio, violar la ley, ser sepulturero, convivir*, respectivamente, demostrar usando tácticas que:

Un vampiro sólo es feliz si se emborracha o pernocta en un cementerio. Quien pernocta en un cementerio viola la ley o es sepulturero. Una bruja sólo es feliz si convive con un vampiro feliz, que no se emborracha y que no viola la ley. Hay brujas felices. Por lo tanto algunos vampiros son sepultureros.

7. Harry Potter estaba teniendo dificultades para resolver una pista en la penúltima prueba del Torneo de los Tres Magos. La prueba consiste en revisar la validez del argumento siguiente:

   El Patronum lo hizo Snape o, hay que jugar Quidditch y molestar a Ron. Si el Patronum lo hizo Snape, hay que jugar Quidditch. Hay que jugar Quidditch si y sólo si Slytherin es la mejor casa de Hogwarts. Por lo tanto, hay que jugar Quidditch y Slytherin es la mejor casa de Hogwarts.

   Ayude a Harry Potter a pasar a la siguiente fase, dándole una serie de tácticas de `Coq` que verifiquen su argumento.

2.3. Un agente racional

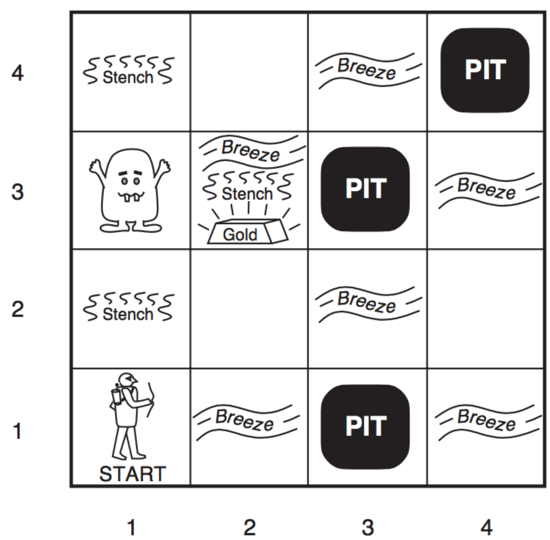

Figura 1: El tablero del *mundo de wumpus*. El agente, inicialmente, se encuentra en la casilla `[1,1]` y debe de averiguar la mejor estrategia que le conduzca al bloque de oro (casilla `[2,3]`) sin entrar en contacto con el monstruo y sin caer en un pozo (*PIT*). Tomado de [1].

En el mundo de *wumpus* (término inglés para referirse a una especie de monstruo) un jugador debe de explorar un camino lleno de peligros con el fin de encontrar el bloque

[1] `http://www.cs.uku.fi/~mnykanen/TEK/teklectures6.pdf`

de oro que le dé la victoria. El juego presentado en este apartado es (casi) equivalente al utilizado dentro del libro *Artificial Intelligence: A Modern Approach*[2].

El mundo de *wumpus* consta de una cuadrícula; el jugador inicialmente comienza su aventura en una de las esquinas y su objetivo es regresar a su posición inicial con el bloque de oro en su haber. El personaje únicamente se puede mover horizontal y verticalmente en un entorno que contiene algunas casillas que le harían perdedor automáticamente de entrar en ellas:

- hay un **monstruo** hediondo, el cual es capaz de comerse vivo al personaje y
- existen varias casillas con **pozos** (*PIT*), los cuales pueden succionar al protagonista al más allá.

Afortunadamente existen algunas pistas que le permitirán al jugador esquivar las casillas peligrosas para sí mismo. Si el jugador se encuentra en la casilla `[i,j]` (donde `i` indica la posición horizontal de izquierda a derecha y `j` indica la posición vertical de abajo hacia arriba), entonces éste puede identificar la existencia del monstruo o de un pozo en una casilla adyacente:

- El fuerte olor del monstruo, despide un hedor (*STENCH*) que se percibe en las casillas vertical y horizontalmente adyacentes.
- El poder succionador de un pozo, ocasiona que se sienta una brisa (*BREEZE*) en las casillas vertical y horizontalmente adyacentes.

Habiendo establecido lo anterior, modelamos el mundo de *wumpus* mediante las siguientes variables proposicionales:

- $P_{i,j}$ *si y sólo si* hay un pozo en la casilla `[i,j]`;
- $W_{i,j}$ *si y sólo si* está el monstruo en la casilla `[i,j]`;
- $B_{i,j}$ *si y sólo si* se siente una brisa en la casilla `[i,j]`;
- $S_{i,j}$ *si y sólo si* se percibe mal olor en la casilla `[i,j]`.

De las anteriores definiciones, de pueden (¡y deben!) deducir algunas equivalencias dadas las pistas, de percepción que posee el jugador. Por ejemplo la sensación de brisa en la casilla `[i,j]` equivale a la existencia de un pozo en alguna de las casillas `[i+1,j]`, `[i,j+1]`, `[i-1,j]` ó `[i,j-1]`, a saber,

$$B_{i,j} \leftrightarrow (P_{i+1,j} \vee P_{i,j+1} \vee P_{i-1,j} \vee P_{i,j-1}). \tag{1}$$

Para la implementación en `Coq`, definimos las variables anteriormente establecidas, mediante tipos que toman dos números naturales y devuelven el tipo `Prop`. Por ejemplo, $S_{i,j}$ se definiría con el enunciado:

[2] `http://aima.cs.berkeley.edu`

```
Variable S : nat -> nat -> Prop.
```

Obsérvese que estamos trabajando únicamente con lógica proposicional. En `Coq`, decir que hay un pozo en la casilla `[3,5]` se expresa mediante `P 3 5`. De esta manera, cualquier expresión que posea los dos números que indican la posición de las casillas es, en sí, una variable proposicional; por ejemplo, se puede demostrar el siguiente teorema trivial:

```
((P 1 1 -> W 4 6) /\ P 1 1) -> W 4 6.
```

Supongamos que el jugador desarrolla su partida como en la cuadrícula de la Figura 1. Inicialmente, no hay ni pozo ni monstruo en la casilla inicial:

H1 $\neg P_{1,1} \wedge \neg W_{1,1}$.

Más aún, supongamos que el jugador ya ha visitado las casillas `[1,2]` y `[2,1]`, por lo que es conocimiento del jugador las siguientes equivalencias, del estilo de la Ecuación 1:

H2 $B_{1,1} \leftrightarrow (P_{2,1} \vee P_{1,2})$,

H3 $S_{1,1} \leftrightarrow (W_{2,1} \vee W_{1,2})$,

H4 $B_{2,1} \leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$,

H5 $S_{1,2} \leftrightarrow (W_{1,1} \vee W_{2,2} \vee W_{1,3})$.

Además, se tiene como hecho lo siguiente:

H6 $\neg B_{1,1}$,

H7 $\neg S_{1,1}$,

H8 $\neg B_{1,2}$,

H9 $\neg S_{2,1}$,

H10 $S_{1,2}$,

H11 $B_{2,1}$.

Queremos, entonces, ayudarle al jugador a decidir correctamente que la casilla `[2,2]` es una opción segura para continuar con su trayecto. Para ello, necesitamos mostrar formalmente que $\neg P_{2,2} \wedge \neg W_{2,2}$. Para finalizar con la implementación en `Coq`, se recomienda asumir las hipótesis (H1 - H11) previamente presentadas mediante enunciados `Axiom`. Por ejemplo,

```
Axiom H2 : B 1 1 <-> (P 2 1 \/ P 1 2).
```

Se recomienda realizar la solución de este problema en un archivo aparte de los demás de la práctica.

3. Entrega

La fecha de entrega es el próximo **viernes 11 de mayo de 2018** por la plataforma de *Google Classroom* del curso y siguiendo los lineamientos del laboratorio.