Ejercicio Semanal 2

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

12 de marzo de 2018

1. Objetivo

Teniendo en cuenta la importancia de la alta complejidad computacional del problema de satisfacibilidad de la lógica proposicional, que el alumno se convenza de la importancia de métodos como el del tableau semántico.

2. Ejercicios

El esqueleto de código del ejercicio semanal se encuentra en `https://github.com/alorozco53/LabLogComp-2018-2/tree/ejsem2`. Considere los siguientes tipos de datos:

```
data FormType = Alpha | Beta  deriving (Show, Eq)
data Tableau = Leaf [Prop]
             | AlphaBranch Prop Prop Tableau
             | BetaBranch Tableau Tableau
            deriving Show
```

`FormType` clasifica una fórmula como α o β, según la tabla presentada en la Nota 4 del curso. Por otro lado, `Tableau` representa el árbol de decisiones construído para realizar el método de verificación de correctud de argumentos visto en clase.

Utilizando el tipo de datos `Prop`, el cual se trabajó en la Práctica 1, realizar las siguientes funciones.

1. Escriba una función `isLiteral` que indique si una fórmula dada es una *literal*.

2. Implemente una función `propType` que diga si el tipo `FormType` (α o *beta*) al que corresponde dicha fórmula.

3. Dé una función `transformProp` que dada una fórmula Φ, devuelva una tupla cuya primera entrada indique el tipo de fórmula de Φ y cuya segunda entrada sea una tupla que contenga las fórmulas (α_1, α_2) o (β_1, β_2), según corresponda y de acuerdo a la tabla presentada en la Nota 4. Por ejemplo, si $\Phi := p \leftrightarrow q$, entonces se espera como salida una tupla como:

```
(Alpha, (Imp (VarP "p") (VarP "q"), Imp (VarP "q") (VarP "p")))
```

   Observación IMPORTANTE: La doble negación es el único caso en el que no hay dos proposiciones resultantes, tras clasificar una fórmula como α. Por ello, se debe de considerar eliminar este caso ya sea implementando una forma normal negativa o detectando la doble negación en la implementación.

4. Implemente una función `makeTableau` que dada una lista de fórmulas (de tipo `[Prop]`), devuelva el tableau semántico de ésta. Se aconseja empezar produciendo una fórmula tras realizar la conjunción de todas las fórmulas de la lista dada. En símbolos, dada la lista $[\Phi_1, \Phi_2, \ldots, \Phi_n]$, sería conveniente empezar el tableau con la fórmula $\Phi_1 \wedge \Phi_2 \wedge \ldots \wedge \Phi_n$.

5. Escriba una función `literalsBranches` que dado un tableau semántico, construya la lista de literales para cada una de las ramas del tableau. Éstas listas de literales serían devueltas dentro de una lista.

6. Dé una función `compLiterals` que decida si dadas dos *literales*, éstas son **complementarias**.

7. Finalmente, escriba una función `correctTableau` que dada una lista de fórmulas Γ y una fórmula Φ, decida si $\Gamma \models \Phi$. Recuerde usar el principio de refutación que reducirá esta pregunta a revisar la satisfacibilidad del conjunto $\Gamma \bigcup \{\neg\Phi\}$. (Una vez tenido dicho conjunto, se procederá a construir el respectivo tableau semántico).

3. Entrega

La fecha de entrega es el próximo **lunes 19 de marzo de 2018** por la plataforma de *Google Classroom* del curso y siguiendo los lineamientos del laboratorio.