Lógica Computacional, 2017-2

Práctica 3: Resolución binaria

Víctor Zamora Gutiérrez Manuel Soto Romero

Fecha de inicio: 1 de marzo de 2017

Fecha de término: 15 de marzo de 2017

Instrucciones generales

- Completar de manera clara y ordenada las funciones de los archivos `prop.hs` y `graficas.hs`.
- Para tener derecho a calificación, la práctica debe ejecutarse sin errores ni advertencias. No está permitido utilizar primitivas de Haskell que resuelvan directamente los ejercicios, ni modificar la firma de ninguna función.
- La entrega es por **equipos de 3 a 4 integrantes**. Seguir los lineamientos especificados en: `http://sites.ciencias.unam.mx/logica-computacional-2017-2/laboratorio/lineamientos`.

Parte I: Resolución binaria

Ejercicios

1. **Equivalencia de fórmulas**

   Completar el cuerpo de la función `equivalentes` que recibe dos fórmulas proposicionales e indica si son equivalentes. Ejemplo:

```
> equivalentes (Op (FA (Var P)) Impl (FA (Var Q))) (Op (Neg (FA (Var P))) Disy
(FA (Var Q)))
True
```

2. **Mejora a `simplifica`**

   En la práctica 2, se implementó la función simplifica que toma una fórmula proposicional y realizaba algunas simplificaciones. El ejercicio consiste en modificar esta función para que realice las siguientes simplificaciones sobre el tipo `Prop`.

$$\neg\varphi \vee \varphi \equiv V \quad V \vee \varphi \equiv V \quad V \wedge \varphi \equiv \varphi \quad F \vee \varphi \equiv \varphi \quad F \wedge \varphi \equiv F \quad \varphi \vee \varphi \equiv \varphi \quad \varphi \wedge \varphi \equiv \varphi$$

   Nota: Recordar que los conectivos lógicos de conjunción y disyunción conmutan.

3. **Resolución binaria**

   Completar el cuerpo de la función `resBin` que toma dos cláusulas y una literal como parámetros y regresa su resolución binaria. Ejemplos:

```
-- resBin (¬p ∨ q) (¬q ∨ r) = ¬p ∨ r
> resBin (Op (Neg (FA (Var P))) Disy (FA (Var Q))) (Op (Neg (FA (Var Q))) Disy
(FA (Var R))) (FA (Var Q))
(Op (Neg (FA (Var P))) Disy (FA (Var R)))
```

Parte II: Gráficas y SAT

Para esta parte se anexa el archivo `graficas.hs` donde se encuentran definidas las gramáticas necesarias para resolver cada ejercicio y que fueron explicadas en clase.

Ejercicios

1. **Vértices de una gráfica**

   Completar el cuerpo de la función `vertices` que dada una gráfica regresa una lista con los vértices que conforman la misma. Ejemplo:

```
> vertices [(1,[]),(2,[3]),(3,[2])]
[1,2,3]
```

2. **Conexidad de gráficas**

   Completar el cuerpo de la función `esConexa` que dada una gráfica determine si es conexa o no. Ejemplos:

```
> esConexa [(1,[2,3]),(2,[1]),(3,[])]
True
> esConexa [(1,[2]),(2,[1]),(3,[])]
False
```

3. **Gráficas completas**

   Completar el cuerpo de la función `esCompleta` que dada una gráfica determina si es completa o no. Ejemplos:

```
> esCompleta [(1,[2]),(2,[1,3]),(3,[2])]
False
> esCompleta [(1,[2,3]),(2,[1,3]),(3,[1,2])]
True
```

4. **Camino hamiltoniano**

   Completar el cuerpo de la función `caminoHamiltoniano` que dada una gráfica determina si contiene un camino hamiltoniano. Ejemplos:

```
> caminoHamiltoniano [(1,[2,3]),(2,[1]),(3,[])]
True
> caminoHamiltoniano [(1,[2]),(2,[1]),(3,[])]
False
```

5. **Clan de una gráfica**

   Completar el cuerpo de la función `clan` que dada una gráfica y un entero k determina si contiene un clan de tamaño k. Ejemplos:

```
> clan [(1,[2,3]),(2,[1,3]),(3,[1,2]),(4,[])] 3
True
> clan [(1,[2,3]),(2,[1,3]),(3,[1,2]),(4,[])] 4
False
```

Puntos extra

Nota: Sólo se puede escoger un punto extra. Los ejercicios deben definirse dentro de `prop.hs`.

- (1 pt.) Definir una función `resolventes :: Clausula -> Clausula -> [Clausula]` que toma dos cláusulas y obtiene el conjunto de posibles resolventes.

- (2 pts) Definir una función `satisfacible :: [Clausula] -> Booleano` que toma una lista de cláusulas y decide si el conjunto es satisfacible.