Las guerras clon

> Fuente: Ejercios de Lógica Computacional
> (https://www.cs.us.es/~fsancho/?p=logica-informatica-2020-21)

Las guerras clon han comenzado. Durante el transcurso de una refriega, tres caballeros Jedi, Anakin, Obi Wan y Yoda, se encuentran con el conde Dooku. Utilizaremos el lenguaje proposicional A, O, Y para denotar que el correspondiente caballero participa en el combate, y G para denotar los Jedi han ganado.

Se sabe que:

- Para derrotar al conde Dooku deben participar al menos dos caballeros Jedi.
- El Conde Dooku gana cuando sólo participa un caballero.
- Si el Conde Dooku pierde entonces Anakin ha participado en el combate.

¿Es cierto que se puede deucir que si los Jedi ganan entonces Anakin y Obi Wan han participado en el combate?

Solución

Los enunciados pueden formalizarse como:

- $G \rightarrow (A \wedge O) \vee (A \wedge Y) \vee (O \wedge Y)$
- $(A \wedge \neg O \wedge \neg Y) \vee (\neg A \wedge O \wedge \neg Y) \vee (\neg A \wedge \neg O \wedge Y) \rightarrow \neg G$
- $G \rightarrow A$

Y la consecuencia como

- $G \rightarrow A \wedge O$

Ahora, determinar la consistencia del conjunto es equivalente a determinar la satisfactibilidad de la conjunción de las fórmulas. Para lo que utilizaremos la FND. Haciendo uso de LogicUS:

```
import LogicUS.PL.SintaxSemantics exposing (..)
import LogicUS.PL.Clauses exposing (..)
import LogicUS.PL.DPLL exposing (..)

fplRead : String -> FormulaPL
fplRead = fplReadExtraction << fplReadFromString
```

Definimos las fórmulas:

```
f1 : FormulaPL
f1 = fplRead "g -> (a & o ) | ( a & y ) | (o & y)"

f2 : FormulaPL
f2 = fplRead "a & ¬o & ¬y | ¬a & o & ¬y | ¬a & ¬o & y -> ¬g"

f3 : FormulaPL
f3 = fplRead "g -> a"

f4 : FormulaPL
f4 = fplRead "g -> a & o"
```

Entonces, determinar la consecuencia es equivalente a determinar la inconsistencia del conjunto $\{F_1, F_2, F_3, \neg F_4\}$

```
cs : ClausePLSet
cs = splToClauses [f1,f2,f3, Neg f4]
```

[[("a",True),("g",False),("o",True)],[("a",True),("g",False),("y",True)],[("a",True),("g",False),("o",True),("y",True)],[("g",False),("o",True),("y",True)],[("a",False),("g",False),("o",True),("y",True)],[("a",True),("g",False),("o",False),("y",True)],[("a",True),("g",False),("o",True),("y",False)],[("a",True),("g",False)],[("g",True)],[("a",False),("o",False)]]

Que corresponde a una lista de cláusulas correspondientes a listas de literales expresados como pares *(símbolo,signo)*.
Representándolas:

$$\{\{a, \neg g, o\}, \{a, \neg g, y\}, \{a, \neg g, o, y\}, \{\neg g, o, y\}, \{\neg a, \neg g, o, y\}, \{a, \neg g, \neg o, y\}, \{a, \neg g, o, \neg y\}, \{a, \neg g\}, \{g\}, \{\neg a, \neg o\}\}$$

De forma que omitiendo las que son subsumidas por otras:

$$\{\{\neg g, o, y\}, \{a, \neg g\}, \{g\}, \{\neg a, \neg o\}\}$$

De forma que aplicando el algoritmo dpll y representándolo en formato DOT

```
csTableau : String
csTableau =  dpll cs |> dpllTableauToDOT
```

"digraph G {\n rankdir=TB\n graph []\n node [shape=box, color=black]\n edge [dir=none, color=blue, fontcolor=blue]\n\n 0 -> 1 [label=\"g\"]\n 1 -> 2 [label=\"a\"]\n 2 -> 3 [label=\"¬ o\"]\n 3 -> 4 [label=\"y\"]\n\n 0 [label=\"{¬ g,o,y}, {a,¬ g}, {g}, {¬ a,¬ o}\"]\n 1 [label=\"{o,y}, {a}, {¬ a,¬ o}\"]\n 2 [label=\"{o,y}, {¬ o}\"]\n 3 [label=\"{y}\"]\n 4 [label=\"◯\"]\n}"
Representándolo: