1. SAT

Definición 1. *Una fórmula booleana es satisfacible si se puede asignar valores de verdad a los correspondientes variables booleanas en la fórmula de forma que al ser evaluada con estos valores de verdad produce un valor verdadero (True).*

Nota:
Por ejemplo, $p \wedge \overline{q}$ es satisfacible cuando p se asigna verdadera y q se asigna falsa. Y $p \wedge \overline{p}$ no es satisfacible porque es falsa sin importar el valor que se asigne a p.

Nota:
El problema general es la satisfacibilidad booleana (SAT), el cual consiste en dada una fórmula booleana determinar si existe una asignación que haga la fórmula booleana satisfacible. Como cualquier fórmula booleana puede ser presentada en forma normal conjuntiva, es equivalente resolver el problema SAT para dichas expresiones booleanas (ver 3SAT donde las expresiones son 3CNF).

$$SAT = \{< \phi > | \phi \text{ es una fórmula boolena satisfacible } \}.$$

Definición 2. *Una literal es una variable booleana o la negación de una variable booleana. Una variable booleana es una literal positiva y su negación es una literal negativa. Para cualquier variable booleana p, { p,¬p} es una pareja complementaria de literales.*

Definición 3. *Una fórmula está en forma normal conjuntiva (CNF) ssi es una conjunción de disyunciones de literales.*

Teorema 1. *Cada fórmula booleana puede ser transformada en una fórmula equivalente en CNF.*

Definición 4. • *Una clásula es un conjunto de literales.*
• *Una cláusula se considera ser una disyunción implicita de sus literales.*
• *Una claúsula unitaria es una cláusula que consiste de exactamente una literal.*
• *El conjunto vacío de literales es la cláusula vacía,* $\square$.
• *Una fórmula en forma clausal es un conjunto de cláusulas.*
• *Una fórmula se considera ser una conjunción implícita de sus cláusulas.*
• *La fórmula que es el conjunto vacío de clásulas es* $\emptyset$.

Teorema 2. *Cada fórmula booleana* ϕ *puede ser transformada en una fórmula equivalente en forma clausal.*

2. Instrucciones del proyecto

1. Realice un programa en Python que implemente un algoritmo utilizando fuerza bruta, donde la entrada sea una fórmula booleana en forma de cláusula y devuelva falsa con asignación vacía o nula (en caso de ser insatisfacible) o devuelva verdadero con la asignación correspondiente (en caso de ser satisfacible).

2. Realice un programa en Python que implemente el algoritmo DPLL sencillo descrito abajo, donde la entrada sea una fórmula booleana en forma de cláusula e asignación vacía y devuelva falsa con asignación vacía o nula (en caso de ser insatisfacible) o devuelva verdadero con la asignación correspondiente (en caso de ser satisfacible).

Algoritmo DPLL:
Entrada: Una fórmula booleana A en forma de cláusulas e asignación vacía.
Salida: Reporta si A es unsatisfacible con asignación parcial vacía o si A es satisfacible y regresa la intepretación parcial que cumple A.

El algortimo se expresa como una función recursiva DPLL(B,I), donde B es la fórmula booleana en forma de cláusulas y I es una asignación parcial.
Inicialmente A es fórmula y la asignación parcial es vacía.

DPLL(B,I)

- Si B es vacía, entonces regresar True e I
- Si hay algúna disyunción vacía en B, entonces regresar False e asignación vacía o nula
- L $\leftarrow$ seleccionar_literal(B) # Pone en forma positiva
- Elimine todas las cláusulas que contiene la literal L en B y elimine las ocurrencias en las cláusulas de la literal complementaria de L en B, construyendo B'
- I' = I $\cup$ {valor de L es verdadero}
- resultado e $I_1 \leftarrow$ DPLL(B',I')
- if resultado es verdadera, entonces regresar True e I_1
- Elimine todas las cláusulas que contiene la literal complementaria L en B y elimine las ocurrencias en las cláusulas de la literal L en B, construyendo B'
- I' = I $\cup$ {valor de L es falso}
- resultado e $I_2 \leftarrow$ DPLL(B',I')
- if resultado es verdadero, entonces regresar True e I_2
- Regresar False y la asignación vacía o nula

Nota: El algoritmo DPLL es no deterministico: se debe seleccionar una literal no asignada en la asignación parcial y luego seleccionar el valor de verdad.

Ejemplos:

fórmula booleana	forma clausal
$p \wedge \overline{p}$	$\{\{p\}, \{\overline{p}\}\}$
$q \vee p \vee \overline{p}$	$\{\{q, p, \overline{p}\}\}$
$(\overline{p} \vee \overline{r} \vee \overline{s}) \wedge (\overline{q} \vee \overline{p} \vee \overline{s})$	$\{\{\overline{p}, \overline{r}, \overline{s}\}, \{\overline{q}, \overline{p}, \overline{s}\}\}$
$(\overline{p} \vee \overline{q}) \wedge (q \vee \overline{s}) \wedge (\overline{p} \vee s) \wedge (\overline{q} \vee s)$	$\{\{\overline{p}, \overline{q}\}, \{q, \overline{s}\}, \{\overline{p}, s\}, \{\overline{q}, s\}\}$
$(\overline{p} \vee \overline{q} \vee \overline{r}) \wedge (q \vee \overline{r} \vee p) \wedge (\overline{p} \vee q \vee r)$	$\{\{\overline{p}, \overline{q}, \overline{r}\}, \{q, \overline{r}, p\}, \{\overline{p}, q, r\}\}$
$r \wedge (\overline{q} \vee \overline{r}) \wedge (\overline{p} \vee q \vee \overline{r}) \wedge q$	$\{\{r\}, \{\overline{q}, \overline{r}\}, \{\overline{p}, q, \overline{r}\}, \{q\}\}$

3. Bibliografía

- Mordechai Ben-Ari. Mathematical Logic for Computer Science. Springer Verlag, Third Edition, London, 2012

- Sipser, M. Introduction to the theory of computation. PWS Publishing company. Boston. 1997.

- https://davefernig.com/2018/05/07/solving-sat-in-python/

- https://en.wikipedia.org/wiki/DPLL_algorithm