# PG47089 - Carlos João Teixeira Preto - Mestrado em Engenharia Informática

**1. Por forma a codificar este puzzle como problema SAT, defina um conjunto adequado de variáveis proposicionais, exprima as regras acima como fórmulas proposicionais, e converta essas fórmulas para CNF.**

**Conjunto de variáveis proposicionais:**

B -> Ter Bigode

C -> Ser Casado

R -> Ser de Ribeirão

A -> Usar Camisola Amarela

D -> Assistir jogos ao Domingo

**Regras Proposicionais:**

Todos os sócios que usam bigode são casados. B -> C **Simplificado:** -B \ / C

Cada sócio do clube que não é de Ribeirão tem que usar camisola amarela. -R -> A **Simplificado:** R \ / A

Os sócios casados não podem assistir aos jogos ao Domingo. C -> -D **Simplificado:** -C \ / -D

Um sócio vai aos jogos ao Domingo se e só se é de Ribeirão. D <-> R **Simplificado:** (-D \ / R) /\ (-R \ / D)

Cada sócio usa bigode ou não usa camisola amarela. B \ / -A

Todos os sócios de Ribeirão usam bigode. R -> B **Simplificado:** -R \ / B

**CNF**:

(-B \ / C) /\ (R \ / A) /\ (-C \ / -D) /\ (-D \ / R) /\ (-R \ / D) /\ (B \ / -A) /\ (-R \ / B)

**2. Codifique o problema num SAT solver e comprove que o conjunto de regras é consistente.**

```
#instalar biblioteca pysat
!pip install python-sat[pblib,aiger]
```

```
from pysat.solvers import Minisat22

s = Minisat22()
conditions = ['B', 'C', 'R', 'A', 'D']
x = {}
```

```
c = 1
for d in conditions:
  x[d] = c
  c += 1

s.add_clause([-x['B'], x['C']])
s.add_clause([x['R'], x['A']])
s.add_clause([-x['C'], -x['D']])
s.add_clause([-x['D'], x['R']])
s.add_clause([-x['R'], x['D']])
s.add_clause([x['B'], -x['A']])
s.add_clause([-x['R'], x['B']])

if s.solve():
  m = s.get_model()
  print("SAT")
  print(m)
else:
  print("UNSAT.")
s.delete()
```

```
SAT
[1, 2, -3, 4, -5]
```

Como se pode comprovar, é possível arranjar um conjunto consistente de valores para as diferentes variáveis proposicionais de modo a que o problema seja satisfazível (SAT), logo o conjunto de regras é consistente.

### 3. Use agora o SAT solver para o ajudar a responder às seguintes questões:

### (a) A afirmação “Quem usa bigode não pode ir ao jogo ao Domingo.” é correcta?

A regra proposicional pode ser descrita como: B -> -D, que simplificado fica: -B \ / -D. Para comprovar se a afirmação é correta, temos verificar se a negação da expressão é UNSAT. Assim, adicionamos a sua negação (B /\ D) às egras proposicionais previamente definidas na alínea 2.

```
from pysat.solvers import Minisat22

s = Minisat22()
conditions = ['B', 'C', 'R', 'A', 'D']
x = {}
c = 1
for d in conditions:
  x[d] = c
  c += 1

s.add_clause([-x['B'], x['C']])
```

```
s.add_clause([x['R'], x['A']])
s.add_clause([-x['C'], -x['D']])
s.add_clause([-x['D'], x['R']])
s.add_clause([-x['R'], x['D']])
s.add_clause([x['B'], -x['A']])
s.add_clause([-x['R'], x['B']])

s.add_clause([x['B']])
s.add_clause([x['D']])

if s.solve():
  m = s.get_model()
  print("SAT")
  print(m)
else:
  print("UNSAT")

s.delete()
```

```
UNSAT
```

Como a negação da afirmação dá UNSAT, comprova-se então que a afirmação é correta.

**(b) Pode um membro de camisola amarela ser casado?**

A regra proposicional pode ser descrita como: A /\ C. Para comprovar se tal pode ocorrer, basta adicionar a regra ao conjunto de regras proposicionais previamente definidas na alínea 2 e verificar se dá SAT.

```
from pysat.solvers import Minisat22

s = Minisat22()
conditions = ['B', 'C', 'R', 'A', 'D']
x = {}
c = 1
for d in conditions:
  x[d] = c
  c += 1

s.add_clause([-x['B'], x['C']])
s.add_clause([x['R'], x['A']])
s.add_clause([-x['C'], -x['D']])
s.add_clause([-x['D'], x['R']])
s.add_clause([-x['R'], x['D']])
s.add_clause([x['B'], -x['A']])
s.add_clause([-x['R'], x['B']])

s.add_clause([x['A']])
s.add_clause([x['C']])
```

```
if s.solve():
  m = s.get_model()
  print("SAT")
  print(m)
else:
  print("UNSAT")

s.delete()
```

```
SAT
[1, 2, -3, 4, -5]
```

Obteve-se SAT, logo confirma-se que um membro de camisola amarela pode ser casado.

**(c) A afirmação “Afinal o clube não pode ter sócios Ribeironenses.” é correcta?**

A regra proposicional pode ser descrita como: -R. Para comprovar se a afirmação é correta, temos verificar se a negação da expressão é UNSAT. Assim, adicionamos a sua negação (R) às regras proposicionais previamente definidas na alínea 2.

```
from pysat.solvers import Minisat22

s = Minisat22()
conditions = ['B', 'C', 'R', 'A', 'D']
x = {}
c = 1
for d in conditions:
  x[d] = c
  c += 1

s.add_clause([-x['B'], x['C']])
s.add_clause([x['R'], x['A']])
s.add_clause([-x['C'], -x['D']])
s.add_clause([-x['D'], x['R']])
s.add_clause([-x['R'], x['D']])
s.add_clause([x['B'], -x['A']])
s.add_clause([-x['R'], x['B']])

s.add_clause([x['R']])

if s.solve():
  m = s.get_model()
  print("SAT")
  print(m)
else:
  print("UNSAT")

s.delete()
```

```
UNSAT
```

Como a negação da afirmação dá UNSAT, comprova-se então que a afirmação é correta e que o clube não pode ter sócios Ribeironenses.

### (d) Os sócios casados têm todos bigode?

A regra proposicional pode ser descrita como: C -> B, que simplificada fica -C \ / B. Para comprovar se tal é verdadeiro, basta verificar se não existe nenhum sócio casado e que não tenha bigode. Assim, adiciona-se (C /\ -B) às regras proposicionais previamente definidas na alínea 2.

```
from pysat.solvers import Minisat22

s = Minisat22()
conditions = ['B', 'C', 'R', 'A', 'D']
x = {}
c = 1
for d in conditions:
  x[d] = c
  c += 1

s.add_clause([-x['B'], x['C']])
s.add_clause([x['R'], x['A']])
s.add_clause([-x['C'], -x['D']])
s.add_clause([-x['D'], x['R']])
s.add_clause([-x['R'], x['D']])
s.add_clause([x['B'], -x['A']])
s.add_clause([-x['R'], x['B']])

s.add_clause([x['C']])
s.add_clause([-x['B']])

if s.solve():
  m = s.get_model()
  print("SAT")
  print(m)
else:
  print("UNSAT")

s.delete()
```

```
UNSAT
```

Obteve-se UNSAT, logo confirma-se que não há nenhum sócio casado que não tenha bigode, e que por isso todos os sócios casados têm bigode.

### (e) A afirmação “Ao domingo nunca há sócios a assistir aos jogos.” é correcta?

A regra proposicional pode ser descrita como: -D. Para comprovar se a afirmação é correta, temos verificar se a negação da expressão é UNSAT. Assim, adicionamos a sua negação (D) às regras proposicionais previamente definidas na alínea 2.

```
from pysat.solvers import Minisat22

s = Minisat22()
conditions = ['B', 'C', 'R', 'A', 'D']
x = {}
c = 1
for d in conditions:
  x[d] = c
  c += 1

s.add_clause([-x['B'], x['C']])
s.add_clause([x['R'], x['A']])
s.add_clause([-x['C'], -x['D']])
s.add_clause([-x['D'], x['R']])
s.add_clause([-x['R'], x['D']])
s.add_clause([x['B'], -x['A']])
s.add_clause([-x['R'], x['B']])

s.add_clause([x['D']])

if s.solve():
  m = s.get_model()
  print("SAT")
  print(m)
else:
  print("UNSAT")

s.delete()
```

```
UNSAT
```

Como não é possível obter uma solução, comprova-se então que a afirmação é correta e que aos domingos nunca há sócios a assistir aos jogos.