SAT Solving

Informações sobre o aluno

Nome: Rui Filipe Pimenta Armada

Número: PG50737

Curso: Mestrado em Engenharia Informática

```
pip install python-sat[pblib,aiger]
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/cola
Collecting python-sat[aiger,pblib]
  Downloading python_sat-0.1.7.dev19-cp37-cp37m-manylinux2010_x86_64.whl (1
     |████████████████████████████████| 1.8 MB 7.1 MB/s
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-package
Collecting pypblib>=0.0.3
  Downloading pypblib-0.0.4-cp37-cp37m-manylinux2014_x86_64.whl (3.4 MB)
     |████████████████████████████████| 3.4 MB 47.7 MB/s
Collecting py-aiger-cnf>=2.0.0
  Downloading py_aiger_cnf-5.0.4-py3-none-any.whl (5.2 kB)
Collecting bidict<0.22.0,>=0.21.0
  Downloading bidict-0.21.4-py3-none-any.whl (36 kB)
Collecting funcy<2.0,>=1.12
  Downloading funcy-1.17-py2.py3-none-any.whl (33 kB)
Collecting py-aiger<7.0.0,>=6.0.0
  Downloading py_aiger-6.1.25-py3-none-any.whl (18 kB)
Collecting attrs<22,>=21
  Downloading attrs-21.4.0-py2.py3-none-any.whl (60 kB)
     |████████████████████████████████| 60 kB 7.4 MB/s
Collecting toposort<2.0,>=1.5
  Downloading toposort-1.7-py2.py3-none-any.whl (9.0 kB)
Requirement already satisfied: sortedcontainers<3.0.0,>=2.3.0 in /usr/local
Collecting parsimonious<0.9.0,>=0.8.1
  Downloading parsimonious-0.8.1.tar.gz (45 kB)
     |████████████████████████████████| 45 kB 2.8 MB/s
Requirement already satisfied: pyrsistent<0.19.0,>=0.18.0 in /usr/local/lib
Building wheels for collected packages: parsimonious
  Building wheel for parsimonious (setup.py) ... done
  Created wheel for parsimonious: filename=parsimonious-0.8.1-py3-none-any.
  Stored in directory: /root/.cache/pip/wheels/88/5d/ba/f27d8af07306b65ee44
Successfully built parsimonious
Installing collected packages: toposort, parsimonious, funcy, bidict, attrs
  Attempting uninstall: attrs
    Found existing installation: attrs 22.1.0
    Uninstalling attrs-22.1.0:
      Successfully uninstalled attrs-22.1.0
Successfully installed attrs-21.4.0 bidict-0.21.4 funcy-1.17 parsimonious-0
```

```
from pysat.solvers import Minisat22
```

Pergunta 1

✓ 0s completed at 11:20 PM

```
# DICIONÁRIO DAS VARIÁVEIS PROPOSICIONAIS
x = {
    'CPU1': 1,
    'CPU2': 2,
    'RAM1': 3,
    'RAM2': 4,
    'MB1': 5,
    'MB2': 6,
    'PG1': 7,
    'PG2': 8,
    'PG3': 9,
    'MON1': 10,
    'MON2': 11,
    'MON3': 12
}
```

Formulas proposicionais

1. Cada computador tem que obrigatóriamente uma única motherboard, um único CPU, uma única placa gráfica e uma única memória RAM.
    - Motherboard

      $$(MB1 \vee MB2) \wedge (\neg MB1 \vee \neg MB2)$$

    - CPU

      $$(CPU1 \vee CPU2) \wedge (\neg CPU1 \vee \neg CPU2)$$

    - Placa Gráfica

      $$(PG1 \vee PG2 \vee PG3) \wedge (\neg PG1 \vee \neg PG2) \wedge (\neg PG1 \vee \neg PG3) \wedge (\neg PG2 \vee -$$

    - RAM

      $$(RAM1 \vee RAM2) \wedge (\neg RAM1 \vee \neg RAM2)$$

Relativamente à restrição dos monitores, visto que se pode ou nao pode ter monitores, a fórmula será uma tautologia, pelo que irá sempre adquirir o valor Verdadeiro.

2. A motherboard MB1 quando combinada com a placa gráfica PG1, obriga à utilização da RAM1.

```
MB1 ∧ PG1 -> RAM1
    transformando em CNF:
        ¬(MB1 ∧ PG1) ∨ (RAM1)
    concluindo:
        ¬MB1 ∨ ¬PG1 ∨ RAM1
```

3. A placa gráfica PG1 precisa do CPU1, excepto quando combinada com uma memória RAM2

```
PG1 -> (CPU1 ∨ RAM2)
    transformando diretamente em CNF:
        ¬PG1 ∨ CPU1 ∨ RAM2
```

4. O CPU2 só pode ser instalado na motherboard MB2

```
CPU2 -> MB2
    transformando diretamente em CNF:
        ¬ CPU2 ∨ MB2
```

5. O monitor MON1 para poder funcionar precisa da placa gráfica PG1 e da memória RAM2.

```
MON1 -> PG1 ∧ RAM2
    transformando em CNF:
        ¬MON1 ∨ (PG1 ∧ RAM2)
    concluindo:
        (¬MON1 ∨ PG1) ∧ (¬MON1 ∨ RAM2)
```

6. O monitor MON2 precisa da memória RAM2 para poder trabalhar com a placa gráfica PG3

```
(MON2 ∧ PG3) -> RAM2
    transformando em CNF:
        ¬(MON2 ∧ PG3) ∨ RAM2
    concluindo:
        ¬MON2 ∨ ¬PG3 ∨ RAM2
```

Pergunta 2

```
s = Minisat22() #cria o solver

# RESTRIÇÃO 1
s.add_clause([x['MB1'], x['MB2']])
s.add_clause([-x['MB1'], -x['MB2']])
s.add_clause([x['CPU1'], x['CPU2']])
s.add_clause([-x['CPU1'], -x['CPU2']])
s.add_clause([x['PG1'], x['PG2'], x['PG3']])
s.add_clause([-x['PG1'], -x['PG2']])
s.add_clause([-x['PG1'], -x['PG3']])
s.add_clause([-x['PG2'], -x['PG3']])
s.add_clause([x['RAM1'], x['RAM2']])
s.add_clause([-x['RAM1'], -x['RAM2']])
# RESTRIÇÃO 2
s.add_clause([-x['MB1'], -x['PG1'], x['RAM1']])
# RESTRIÇÃO 3
s.add_clause([-x['PG1'], x['CPU1'], x['RAM2']])
# RESTRIÇÃO 4
s.add_clause([-x['CPU2'], x['MB2']])
# RESTRIÇÃO 5
s.add_clause([-x['MON1'], x['PG1']])
s.add_clause([-x['MON1'],x['RAM2']])
# RESTRIÇÃO 6
s.add_clause([-x['MON2'], -x['PG3'], x['RAM2']])

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

s.delete()
```

```
SAT
[1, -2, 3, -4, 5, -6, 7, -8, -9, -10, -11]
```

Como se pode verificar pelo *output* do *SAT solver*, o conjunto de fórmulas utilizado é consistente. Uma resposta válida para o problema é um computador com a seguinte configuração:

- 1 CPU1
- 1 RAM1
- 1 MB1
- 1 PG1

Pergunta 3

a) O monitor MON1 só poderá ser usado com uma motherboard MB1 ?

- De forma a conseguir responder à questão é necessário inserir clausulas em que o valor lógico de MON1 seja verdadeiro e o de MB1 seja falso.
- Se for possível obter um modelo pode-se usar o MON1 com qualquer outra motherboard, caso contrário, apenas pode ser usada a MB1.

```
s = Minisat22() #cria o solver

# RESTRIÇÃO 1
s.add_clause([x['MB1'], x['MB2']])
s.add_clause([-x['MB1'], -x['MB2']])
s.add_clause([x['CPU1'], x['CPU2']])
s.add_clause([-x['CPU1'], -x['CPU2']])
s.add_clause([x['PG1'], x['PG2'], x['PG3']])
s.add_clause([-x['PG1'], -x['PG2']])
s.add_clause([-x['PG1'], -x['PG3']])
s.add_clause([-x['PG2'], -x['PG3']])
s.add_clause([x['RAM1'], x['RAM2']])
s.add_clause([-x['RAM1'], -x['RAM2']])
# RESTRIÇÃO 2
s.add_clause([-x['MB1'], -x['PG1'], x['RAM1']])
# RESTRIÇÃO 3
s.add_clause([-x['PG1'], x['CPU1'], x['RAM2']])
# RESTRIÇÃO 4
s.add_clause([-x['CPU2'], x['MB2']])
# RESTRIÇÃO 5
s.add_clause([-x['MON1'], x['PG1']])
s.add_clause([-x['MON1'],x['RAM2']])
# RESTRIÇÃO 6
s.add_clause([-x['MON2'], -x['PG3'], x['RAM2']])
# CLAUSULAS INSERIDAS
s.add_clause([x['MON1']])
s.add_clause([-x['MB1']])


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

s.delete()
```

```
SAT
[1, -2, -3, 4, -5, 6, 7, -8, -9, 10, -11]
```

Como é possivel obter um modelo, pode-se verificar que o MON1 pode ser usado com

qualquer outra motherboard. Uma resposta válida para o problema é um computador com a seguinte configuração:

- 1 CPU1
- 1 RAM2
- 1 MB2
- 1 PG1
- 1 MON1

b) Um cliente pode personalizar o seu computador da seguinte forma: uma motherboard MB1, o CPU1, a placa gráfica PG2 e a memória RAM1 ?

- De forma a conseguir responder à questão basta inserir clausulas de forma a descrever a configuração do computador pretendido.
- Se for possível obter um modelo idêntico como resultado signidica que a configuração é possível, caso contr+ario, é uma configuração impossível.

Double-click (or enter) to edit

```
s = Minisat22() #cria o solver

# RESTRIÇÃO 1
s.add_clause([x['MB1'], x['MB2']])
s.add_clause([-x['MB1'], -x['MB2']])
s.add_clause([x['CPU1'], x['CPU2']])
s.add_clause([-x['CPU1'], -x['CPU2']])
s.add_clause([x['PG1'], x['PG2'], x['PG3']])
s.add_clause([-x['PG1'], -x['PG2']])
s.add_clause([-x['PG1'], -x['PG3']])
s.add_clause([-x['PG2'], -x['PG3']])
s.add_clause([x['RAM1'], x['RAM2']])
s.add_clause([-x['RAM1'], -x['RAM2']])
# RESTRIÇÃO 2
s.add_clause([-x['MB1'], -x['PG1'], x['RAM1']])
# RESTRIÇÃO 3
s.add_clause([-x['PG1'], x['CPU1'], x['RAM2']])
# RESTRIÇÃO 4
s.add_clause([-x['CPU2'], x['MB2']])
# RESTRIÇÃO 5
s.add_clause([-x['MON1'], x['PG1']])
s.add_clause([-x['MON1'],x['RAM2']])
# RESTRIÇÃO 6
s.add_clause([-x['MON2'], -x['PG3'], x['RAM2']])
# CLAUSULAS INSERIDAS
s.add_clause([x['MB1']])
s.add_clause([x['CPU1']])
s.add_clause([x['PG2']])
s.add_clause([x['RAM1']])
```

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

s.delete()
```

```
SAT
[1, -2, 3, -4, 5, -6, -7, 8, -9, -10, -11]
```

Portanto, a solução apresentada pelo *SAT solver* é a seguinte:

- 1 CPU1
- 1 RAM1
- 1 MB1
- 1 PG2

Logo, pode observar-se que a solução obtida é idêntica à configuração colocada pela questão, logo pode-se confirmar que o cliente pode personalizar um computador com as componentes propostas

c) É possivel combinar a motherboard MB2, a placa gráfica PG3 e a RAM1 num mesmo computador ?

- De forma a conseguir responder à questão é necessário inserir clausulas que indiquem as componentes que se pretendem usar.
- Se for possível obter um modelo pode-se confirmar que é possível combinar as componentes desejadas, caso contrário, não é possivel configuar o computador de tal forma.

```
s = Minisat22() #cria o solver

# RESTRIÇÃO 1
s.add_clause([x['MB1'], x['MB2']])
s.add_clause([-x['MB1'], -x['MB2']])
s.add_clause([x['CPU1'], x['CPU2']])
s.add_clause([-x['CPU1'], -x['CPU2']])
s.add_clause([x['PG1'], x['PG2'], x['PG3']])
s.add_clause([-x['PG1'], -x['PG2']])
s.add_clause([-x['PG1'], -x['PG3']])
s.add_clause([-x['PG2'], -x['PG3']])
s.add_clause([x['RAM1'], x['RAM2']])
s.add_clause([-x['RAM1'], -x['RAM2']])
# RESTRIÇÃO 2
s.add_clause([-x['MB1'], -x['PG1'], x['RAM1']])
# RESTRIÇÃO 3
s.add_clause([-x['PG1'], x['CPU1'], x['RAM2']])
# RESTRICÃO 4
```

```
s.add_clause([-x['CPU2'], x['MB2']])
# RESTRIÇÃO 5
s.add_clause([-x['MON1'], x['PG1']])
s.add_clause([-x['MON1'],x['RAM2']])
# RESTRIÇÃO 6
s.add_clause([-x['MON2'], -x['PG3'], x['RAM2']])
# CLAUSULAS INSERIDAS
s.add_clause([x['MB2']])
s.add_clause([x['PG3']])
s.add_clause([x['RAM1']])


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

s.delete()
```

```
SAT
[1, -2, 3, -4, -5, 6, -7, -8, 9, -10, -11]
```

Portanto, a solução apresentada pelo *SAT solver* é a seguinte:

- 1 CPU1
- 1 RAM1
- 1 MB1
- 1 PG2

Logo, como é possível obter um modelo, pode-se afirmar que as componentes indicadas podem ser utilizadas em conjunto.

d) Para combinarmos a placa gráfica PG2 e a RAM1 temos que usar o CPU2 ?

- De forma a conseguir responder à questão é necessário inserir clausulas que indiquem as que se pretende utilizar a PG2 e a RAM1 mas que nao se pretende usar o CPU2.
- Se for possível obter um modelo pode-se confirmar que não é necessário utilizar o CPU2 para combinar a PG2 e a RAM1.

```
s = Minisat22() #cria o solver

# RESTRIÇÃO 1
s.add_clause([x['MB1'], x['MB2']])
s.add_clause([-x['MB1'], -x['MB2']])
s.add_clause([x['CPU1'], x['CPU2']])
s.add_clause([-x['CPU1'], -x['CPU2']])
s.add_clause([x['PG1'], x['PG2'], x['PG3']])
s.add_clause([-x['PG1'], -x['PG2']])
s.add_clause([-x['PG1'], -x['PG3']])
```

```
s.add_clause([-x['PG2'], -x['PG3']])
s.add_clause([x['RAM1'], x['RAM2']])
s.add_clause([-x['RAM1'], -x['RAM2']])
# RESTRIÇÃO 2
s.add_clause([-x['MB1'], -x['PG1'], x['RAM1']])
# RESTRIÇÃO 3
s.add_clause([-x['PG1'], x['CPU1'], x['RAM2']])
# RESTRIÇÃO 4
s.add_clause([-x['CPU2'], x['MB2']])
# RESTRIÇÃO 5
s.add_clause([-x['MON1'], x['PG1']])
s.add_clause([-x['MON1'],x['RAM2']])
# RESTRIÇÃO 6
s.add_clause([-x['MON2'], -x['PG3'], x['RAM2']])
# CLAUSULAS INSERIDAS
s.add_clause([x['PG2']])
s.add_clause([x['RAM1']])
s.add_clause([-x['CPU2']])


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

s.delete()
```

```
SAT
[1, -2, 3, -4, 5, -6, -7, 8, -9, -10, -11]
```

Portanto, a solução apresentada pelo *SAT solver* é a seguinte:

- 1 CPU1
- 1 RAM1
- 1 MB1
- 1 PG2

Logo, como é possível obter um modelo, pode-se concluir que não é necessário utilizar o CPU2 para combinar a PG2 e a RAM1.

Colab paid products - Cancel contracts here