Guião para aula laboratorial de Verificação Formal (2020/2021)

SAT solving

Comece por instalar um SAT solver. Recomendamos que instale o MiniSat.

1 Exemplos (DIMACS CNF format)

A fórmula proposicional $A_1 \wedge (A_1 \vee P) \wedge (\neg A_1 \vee \neg P \vee A_2) \wedge (A_1 \vee \neg A_2)$ encontra-se já em CNF e pode ser escrita no formato DIMACS como se segue (`example.cnf`):

```
p cnf 3 4
1 0
1 3 0
-1 -3 2 0
1 -2 0
```

Exemplifica-se a invocação de um solver com o MiniSat:

```
$ minisat example.cnf OUT
```

A solução calculada é:

```
SAT
 1 -2 -3 0
```

ou seja, $A_1 = 1$, $A_2 = 0$ e $P = 0$.

Experimente agora invocar o SAT solver com os ficheiros `sat-100v429c.cnf` e `unsat-175v753c.cnf` e analise a resposta do solver.

2 Modelação com lógica proposicional

Tendo em conta as restrições seguintes, em que dia poderá ter lugar uma reunião envolvendo todas as pessoas referidas?

- *Maria cannot meet on Wednesday.*
- *Peter can only meet either on Monday, Wednesday or Thursday.*
- *Anne cannot meet on Friday.*
- *Mike cannot meet neither on Tuesday nor on Thursday.*

Considere variáveis proposicionais Mon, Tue, Wed, Thu, Fri, e escreva uma fórmula em CNF que exprima as restrições acima. Note que não é relevante referir as diferentes pessoas!

Um modelo para essa fórmula corresponderá a uma solução para o problema. Encontre-a codificando a fórmula em DIMACS e invocando o SAT solver.

3 Conversão para CNF e classificação de fórmulas

Converta cada uma das fórmulas seguintes para CNF e determine, com a ajuda de um SAT solver, se ela é **satisfazível**, **válida**, **refutável**, ou uma **contradição**.

1. $A \lor (A \to B) \to A \lor \neg B$
2. $(A \to B \lor C) \land \neg(A \land \neg B \to C)$
3. $(\neg A \to \neg B) \to (\neg A \to B) \to A$

Uma alternativa é utilizar a transformação de Tseitin para obter uma fórmula **equisatisfazível** à original introduzindo novas variáveis proposicionais. Determine a satisfazibilidade da fórmula $P \land Q \lor (R \land P)$, começando por lhe aplicar esta transformação.

4 Puzzle do Unicórnio

Considere o seguinte enigma:

- *If the unicorn is mythical, then it is immortal.*
- *If the unicorn is not mythical, then it is a mortal mammal.*
- *If the unicorn is either immortal or a mammal, then it is horned.*
- *The unicorn is magical if it is horned.*

- *Is the unicorn mythical? Is it magical? Is it horned?*

Para o resolver considere 5 variáveis proposicionais, correspondentes a 5 propriedades dos unicórnios, e começe por completar o seguinte ficheiro no formato DIMACS, `unicornpuzzle.cnf`, com a descrição das restrições acima:

```
c The Unicorn puzzle
c
c  1 mythical?
c  2 immortal?
c  3 mammal?
c  4 horned?
c  5 magical?
c
p cnf 5 ???
(...)
```

Invoque depois o seu SAT solver preferido, por exemplo:

```
$ minisat unicornpuzzle.cnf OUT
```

Caso o problema seja satisfazível, a solução pode ser lida no ficheiro `OUT`.

Note que é possível obter soluções alternativas para o problema, incluindo soluções já conhecidas nas restrições. Suponha por exemplo que na invocação acima o solver encontrou a solução seguinte:

```
$ more OUT
SAT
-1 -2 3 4 5 0
```

Considerada como solução (modelo), o significado da última linha é:

$$(\neg mythical) \wedge (\neg immortal) \wedge mammal \wedge horned \wedge magical$$

Para obtermos uma nova solução basta incluir no ficheiro unicornpuzzle.cnf a negação desta fórmula como restrição:

```
1 2 -3 -4 -5 0
```

Interpretada como restrição, o seu significado é o seguinte, exprimindo que pelo menos um dos valores lógicos atribuídos pelo modelo anterior terá agora de ser diferente.

$$mythical \vee immortal \vee (\neg mammal) \vee (\neg horned) \vee (\neg magical)$$

1. Tendo em conta isto, quantos modelos existem para este problema?
2. Comente as restrições que acrescentou e use agora o SAT solver para responder às perguntas do enigma:

   - Is the unicorn mythical? Is it magical? Is it horned?

5 Placement of Guests

Temos 3 cadeiras numa fila (*left*, *middle*, *right*), e precisamos de distribuir por elas 3 pessoas (*Anne*, *Susan* e *Peter*), com as seguintes restrições:

- Anne does not want to sit near Peter.
- Anne does not want to sit in the left chair.
- Susan does not want to sit to the right of Peter.

Para formular o problema em lógica proposicional, consideramos a seguinte indexação de pessoas e posições:

$Anne = 1$, $Susan = 2$, $Peter = 3$
$left\ chair = 1$, $middle\ chair = 2$, $right\ chair = 3$

Introduzimos depois variáveis proposicionais x_{ij} para $i \in \{1, 2, 3\}$ e $j \in \{1, 2, 3\}$, sendo que

$x_{ij} = 1$ *sse* a pessoa i se senta na cadeira j

As restrições a escrever pertencem a várias categorias:

1. **Todas as pessoas devem estar sentadas numa cadeira**

$$(x_{11} \vee x_{12} \vee x_{13}) \wedge (x_{21} \vee x_{22} \vee x_{23}) \wedge (x_{31} \vee x_{32} \vee x_{33})$$

2. **Não se poderá sentar mais do que uma pessoa em cada cadeira**, o que corresponde a um conjunto de restrições de incompatibilidade entre as variáveis relativas à mesma cadeira.

   A fórmula $\neg x_{ic} \vee \neg x_{jc}$ exprime que as pessoas i e j não podem estar ambas sentadas na cadeira c (note que pode ser escrita como $x_{ic} \to \neg x_{jc}$ ou como $x_{jc} \to \neg x_{ic}$).

   Escreva todas as restrições de incompatibilidade necessárias.

3. Restrições correspondentes às **preferências** das 3 pessoas.

Escreva todas as restrições necessárias e resolva o problema invocando o SAT solver.

6 N-raínhas

Considere, no contexto do jogo de xadrez (tabuleiro de 8×8) o problema de dispor 8 rainhas por forma a que nenhuma delas seja alvo de qualquer outra.

Tendo em conta que a rainha tem 4 direcções de movimento (horizontal, vertical e duas diagonais), isto significa na prática que:

- haverá **no máximo** uma raínha em cada linha, coluna, ou linha diagonal do tabuleiro

Além disso, uma vez que se pretende colocar 8 raínhas, terá necessariamente de ser verdade o seguinte:

- haverá **pelo menos** uma raínha em cada linha e em cada coluna do tabuleiro

O problema pode ser generalizado para tabuleiros de qualquer dimensão $N \times N$, pretendendo-se nesse caso nele dispor N rainhas. Neste exercício pretende-se resolver o problema para $N = 4$, com a ajuda de um SAT solver.

Restrições do tipo "no máximo uma..."

Imaginemos que as variáveis q_{11} a q_{14} denotam a presença de uma raínha nas 4 posições da linha 1 do tabuleiro. Uma forma de exprimir a restrição de que não poderá haver mais do que uma raínha nesta linha será:

$$\begin{array}{l}
(q_{11} \to \neg(q_{12} \vee q_{13} \vee q_{14})) \wedge \\
(q_{12} \to \neg(q_{11} \vee q_{13} \vee q_{14})) \wedge \\
(q_{13} \to \neg(q_{11} \vee q_{12} \vee q_{14})) \wedge \\
(q_{14} \to \neg(q_{11} \vee q_{12} \vee q_{13}))
\end{array}$$

A implicação da primeira linha poderá ser escrita como

$$(q_{11} \to \neg q_{12}) \wedge (q_{11} \to \neg q_{13}) \wedge (q_{11} \to \neg q_{14})$$

ou

$$(\neg q_{11} \vee \neg q_{12}) \wedge (\neg q_{11} \vee \neg q_{13}) \wedge (\neg q_{11} \vee \neg q_{14})$$

É fácil ver que ao aplicar esta transformação a todas as cláusulas da fórmula inicial surgirá redundância, que depois de eliminada resultará na seguinte fórmula (CNF):

$$\begin{array}{l}
(\neg q_{11} \vee \neg q_{12}) \wedge (\neg q_{11} \vee \neg q_{13}) \wedge (\neg q_{11} \vee \neg q_{14}) \wedge \\
(\neg q_{12} \vee \neg q_{13}) \wedge (\neg q_{12} \vee \neg q_{14}) \wedge \\
(\neg q_{13} \vee \neg q_{14})
\end{array}$$

Esqueçamos o primeiro índice das variáveis, que não desempenha aqui qualquer papel. A incompatibilidade par-a-par de 4 variáveis proposicionais q_1 a q_4 poderá ser escrita em geral como:

$$(\neg q_1 \vee \neg q_2) \wedge (\neg q_1 \vee \neg q_3) \wedge (\neg q_1 \vee \neg q_4) \wedge (\neg q_2 \vee \neg q_3) \wedge (\neg q_2 \vee \neg q_4) \wedge (\neg q_3 \vee \neg q_4)$$

Ou generalizando para N variáveis:

$$\bigwedge_{i=1}^{N-1} \bigwedge_{j=i+1}^{N} (\neg q_i \vee \neg q_j)$$

Tendo em conta o exposto:

1. Modele o problema das 4 rainhas como problema SAT, e codifique-o no formato DIMACS (o template em baixo contém indicação do número de cláusulas necessárias para codificar cada conjunto de restrições).
2. Resolva-o com um SAT solver.
3. Verifique manualmente que de facto a solução encontrada pelo solver satisfaz as restrições do problema enunciadas acima.

```
c The 4-Queens Problem
c
c  Board represented by 16 prop vars:
c       1  2  3  4
c       5  6  7  8
c       9 10 11 12
c      13 14 15 16
c  Each prop var denotes the presence of a queen in the corresponding board position
c
c  Constraints:
c  * at least one queen per row (4)
c  * at least one queen per column (4)
c  * at most one queen per row (24)
c  * at most one queen per column (24)
c  * at most one queen per diagonal (14 + 14)
c
p cnf 16 ???
(...)
```

7 Equivalência de *branching programs*

Considere os programas

(1)
```
if (!a && !b) h();
else if (!a) g();
     else f();
```

(2)

```
if (a) f();
      else if (b) g();
           else h();
```

É possível determinar se eles são equivalentes com a ajuda de um SAT solver. Para isso codificamos logicamente cada um deles, usando a seguinte regra de compilação:

$$\mathsf{compile}(\textbf{if } x \textbf{ then } y \textbf{ else } z) \;=\; (x \wedge y) \vee (\neg x \wedge z)$$

Basta depois decidir se as fórmulas $\mathsf{compile}(\mathbf{1})$ e $\mathsf{compile}(\mathbf{2})$ são **equivalentes**. Para isso teremos que resolver um problema de **validade** da fórmula

$$\mathsf{compile}(\mathbf{1}) \leftrightarrow \mathsf{compile}(\mathbf{2})$$

.

Para resolver um problema de validade com um solver de satisfazibilidade, há que negar a fórmula:

$$\neg(\mathsf{compile}(1) \leftrightarrow \mathsf{compile}(2))$$
$$\equiv$$
$$\neg((\mathsf{compile}(1) \wedge \mathsf{compile}(2)) \vee (\neg\mathsf{compile}(1) \wedge \neg\mathsf{compile}(2)))$$
$$\equiv$$
$$(\neg\mathsf{compile}(1) \vee \neg\mathsf{compile}(2)) \wedge (\mathsf{compile}(1) \vee \mathsf{compile}(2))$$

Se esta fórmula negada for UNSAT, então os programas serão equivalentes.

Considerando variáveis a, b, f, g, h, codifique os programas **(1)** e **(2)** acima e determine, convertendo a fórmula para CNF e usando o SAT solver, se eles são ou não equivalentes.