SAT trabalho

Renan Vieira

Julho de 2019

1 Fluxograma

1. Modele o problema. Aqui não há "mistério": é entender as regras de cada tema e converter para lógica proposicional. Para isso, escreva as regras em um papel ou documento de texto. Por exemplo:

   - Uma regra para o tema do curso é "todo curso deve possuir somente uma única cor", onde a cor seria um horário específico.
   - Uma regra para o tema do campo minado é "O campo possui uma única bomba"
   - Uma regra para o quadrado latino é "a diagonal deve possuir cada um dos números".
   - Uma regra para o problema das rainhas é "não pode haver mais de uma rainha na diagonal"

   Para convertermos essas regras em lógica proposicional, definimos atômicas de cada um das regras:

   - Seja c_{ij}, representando que *o curso i possui cor j*;
   - Seja b_{ij}, representando que *há uma bomba no quadrado linha i coluna j*
   - Seja n_{kij}, representando que *há o número k na linha i coluna j*
   - Seja q_{ij}, representando que *há uma rainha na linha i coluna j*

   E com essas atômicas podemos descrever cada uma das regras estabelecidas anteriormente. Observem que essas **NÃO** são as únicas regras de cada um dos temas e essa **NÃO** é a única forma de modelar esses problemas. São só exemplos. Ou seja, você pode modelar as regras de formas diferentes e vocês podem precisar de outras atômicas para outras regras. Observe que algumas formulas podem ficar MUITO grandes, mas por isso que vocês devem utilizar alguma linguagem de programação para automatizar essa etapa. Naturalmente, vocês devem fazer alguns esboços no papel e automatizar a criação de todas as regras.

2. Converta as regras/fórmulas para CNF: Assumindo que você criou as fórmulas anteriores em forma não CNF, vocês tem 4 opções:

   - Converter manualmente: isso pode até funcionar para alguns problemas e regras, mas para alguns muito grandes não vai dar certo, visto que algumas podem ficar muito complexas.
   - Converter online (recomendado): há alguns sites que convertem formulas para CNF (http://formal.cs.utah.edu:8080/pbl/PBL.php). O problema é que esse site gera formulas muitas vezes redundantes e repletas de cláusulas triviais. Logo, seria necessário algum tipo de filtragem para remover essas cláusulas. Eu fiz meus exemplos com esse site e fiz um código em python que remove cláusulas triviais, cláusulas repetidas e literais repetidos em uma mesma cláusula.
   - Implementar o código de converter para CNF. Outra opção viável, mas exige que vocês representem as formulas em alguma estrutura de dados. Fazer via string pode funcionar (ainda mais em python que ajuda muito a manipular string) mas pode dar trabalho.
   - Buscar bibliotecas/funções prontas online (recomendado): provavelmente alguém já implementou isso na sua linguagem preferida (eu achei para Python, o próprio site acima disponibiliza uma biblioteca). Google it.

3. Converta para formato Dimacs;

   Observem o exemplo abaixo, acredito que irá elucidar bem.

4. Carregue o arquivo para o algoritmo DPLL

   Implemente o algoritmo DPLL que recebe como entrada o arquivo Dimacs.

2 Exemplo:

Vamos fazer o passo a passo para uma questão da prova:

Uma montadora de carros constrói carros com as seguintes características de motor, transmissão e tração:

a) Φ_1: O motor deve seguir exatamente uma entre as seguintes opções: 1.2, 1.4 e 1.8.

b) Φ_2: A transmissão automática não é disponível no modelo com motor 1.2.

c) Φ_3: Um carro tem transmissão automática ou manual, mas não ambos.

d) Φ_4: A tração nas quatro rodas é disponível apenas no modelo com motor 1.8.

e) Φ_5: A transmissão manual não pode ser combinada com a tração nas quatro rodas.

2.1 Primeiro Passo: Modelar para lógica proposicional:

Definimos as atômicas:

- $m_{1.2}$: O carro possui motor 1.2;
- $m_{1.4}$: O carro possui motor 1.4;
- $m_{1.8}$: O carro possui motor 1.8;
- t_a: O carro possui transmissão automática;
- t_m: O carro possui transmissão manual;
- q: O carro possui tração quatro rodas;

Escrevemos as restrições como lógica proposicional utilizando as atômicas:

a) Φ_1: $(m_{1.2} \wedge \neg m_{1.4} \wedge \neg m_{1.8}) \vee (\neg m_{1.2} \wedge m_{1.4} \wedge \neg m_{1.8}) \vee (\neg m_{1.2} \wedge \neg m_{1.4} \wedge m_{1.8})$

b) Φ_2: $m_{1.2} \rightarrow \neg t_a$

c) Φ_3: $(t_m \wedge \neg t_a) \vee (\neg t_m \wedge t_a)$

d) Φ_4: $q \rightarrow m_{1.8}$

e) Φ_5: $(t_m \wedge \neg q) \vee (\neg t_m \wedge q)$

2.2 Segundo Passo: Transformar para CNF

a) Φ_1: $(m_{1.2} \vee m_{1.4} \vee m_{1.8}) \wedge (\neg m_{1.2} \vee \neg m_{1.4} \vee \neg m_{1.8}) \wedge (m_{1.2} \vee \neg m_{1.4} \vee \neg m_{1.8}) \wedge (\neg m_{1.2} \vee m_{1.4} \vee \neg m_{1.8}) \wedge (\neg m_{1.2} \vee \neg m_{1.4} \vee m_{1.8}) \wedge (\neg m_{1.2} \vee \neg m_{1.4}) \wedge (\neg m_{1.4} \vee \neg m_{1.8}) \wedge (\neg m_{1.2} \vee \neg m_{1.8})$

b) Φ_2: $\neg m_{1.2} \vee \neg t_a$

c) Φ_3: $(t_m \vee t_a) \wedge (\neg t_m \vee \neg t_a)$

d) Φ_4: $\neg q \vee m_{1.8}$

e) Φ_5: $(t_m \vee q) \wedge (\neg t_m \vee \neg q)$

2.3 Terceiro Passo: Formato Dimacs

Nosso formato Dimacs será formado pela formula: $\Psi : \Phi_1 \wedge \Phi_2 \wedge \Phi_3 \wedge \Phi_4 \wedge \Phi_5 \wedge \Phi_6$. Observe que ainda não definimos Φ_6. Ela será nossa fórmula que representará nossa configuração de entrada, ou seja, de algum carro que queremos verificar se é possível produzir. Dessa forma, ela vai variar de acordo com a entrada do usuário. Entraremos em mais detalhes sobre ela a posteriormente.

Uma vez que você tem tem suas formulas em CNF, converter para formato dimacs é bem simples: atribua para cada literal um número inteiro. Os literais negativos irão receber o mesmo inteiro, mas negativo. A partir das atômicas definidas em 2.1, atribuímos um valor inteiro para cada uma delas:

- $m_{1.2}$: 1, $\neg m_{1.2}$: -1;
- $m_{1.4}$: 2, $\neg m_{1.4}$: -2;
- $m_{1.8}$: 3, $\neg m_{1.8}$: -3;
- t_a: 4, $\neg t_a$: -4;
- t_m: 5, $\neg t_m$: -5;
- q: 6, $\neg q$: -6,

...E criamos a fórmula Ψ. Por notação de conjunto:

- $\Phi_1 = \{1,2,3\},\{-1,-2,-3\},\{1,-2,-3\},\{-1,2,-3\},\{-1,-2,3\},\{-1,-2\},\{-2,-3\},\{-1,-3\}$
- $\Phi_2 = \{-1,-4\}$
- $\Phi_3 = \{4,5\},\{-4,-5\}$
- $\Phi_4 = \{-6,3\}$
- $\Phi_5 = \{5,6\},\{-5,-6\}$

Agora vamos considerar diferentes cenários para Φ_6 :

- Φ_6' : Transmissão manual, tração nas quatro rodas, motor 1.4
- Φ_6'' : Transmissão automática, sem tração nas quatro rodas, motor 1.8
- Φ_6''' : Transmissão manual, tração nas quatro rodas, motor 1.8
- Φ_6'''' : Transmissão automática, sem tração nas quatro rodas, motor 1.2 e 1.8

Converter para lógica proposicional e CNF:

- Φ_6' : $t_m \wedge q \wedge m_{1.4}$; CNF = $\{5\},\{6\},\{2\}$
- Φ_6'' : $t_a \wedge \neg q \wedge m_{1.8}$; CNF = $\{4\},\{-6\},\{3\}$
- Φ_6''' : $t_m \wedge q \wedge m_{1.8}$; CNF = $\{5\},\{6\},\{3\}$
- Φ_6'''' : $t_a \wedge \neg q \wedge m_{1.2} \wedge m_{1.8}$; CNF = $\{5\},\{6\},\{1\},\{3\}$

Note que para **QUALQUER ARQUIVO DIMACS** desse problema, sempre conterá as restrições $(\Phi_1, \Phi_2, \Phi_3, \Phi_4, \Phi_5)$. Contudo, o arquivo será diferente dependendo da entrada a ser verificada (Φ_6). Há exemplos dos arquivos Dimacs junto a esse documento para os exemplos feitos acima.

2.4 Quarto Passo: Algoritmo DPLL

Nesse último passo, algoritmo deve ler um arquivo Dimacs qualquer e executar. Inclusive, teste sua implementação nos arquivos que está junto a esse documento.