

Lógica Computacional

- O Método de Resolução
- Problema SAT: Decidibilidade e Complexidade
- Cláusulas de Horn
- Algoritmos Horn-SAT e sua correcção

Resolução

- Embora o método de dedução natural simule de alguma forma tipo de raciocínio usado pelas pessoas, o sistema DN não é em geral adequado a uma implementação de raciocínio computacional.
- De facto, existem muitas regras que podem ser usadas em cada passo de uma demonstração. Se uma pessoa **com algum treino** consegue determinar qual o curso mais adequado para prosseguir a demonstração essa decisão obriga em geral a ter em conta um grande número de informações, não sendo fácil especificar um algoritmo que simule a **estratégia** usada.
- Desta forma foram estudados métodos de demonstração alternativos e mais adequados a uma implementação computacional, quer para a lógica proposicional quer para a lógica de predicados (com quantificadores).
- De entre este, realçamos o método de **Resolução**.
- Neste método, e no sistema associado \mathcal{R} , a variedade de fórmulas utilizadas e o número de regras de inferência é reduzido a um mínimo de forma a facilitar a sua implementação.

Resolução Proposicional

- Começando pelo caso proposicional (sem igualdade nem quantificação). Neste contexto pretende-se estudar a validade de argumentos do tipo

$$\{ P_1, P_2, \dots, P_n \} \models G$$

em que as várias premissas P_i e a conclusão C são FPOs, isto é, fórmulas de primeira ordem não quantificadas nem contendo o predicado de igualdade.

- A primeira característica do sistema \mathcal{R} (de Resolução) é utilizar a **refutação** como método de demonstração. A refutação não é mais do que a redução ao absurdo da **negação** da conclusão.
- Assim, em vez da demonstração acima, o sistema \mathcal{R} faz a demonstração equivalente:

$$\{ P_1, P_2, \dots, P_n, \neg G \} \vdash_{\mathcal{R}} \perp$$

Desta forma fazer uma demonstração no sistema \mathcal{R} é demonstrar que a fórmula

$$F =_{\text{def}} P_1 \wedge P_2 \wedge \dots \wedge P_n \wedge \neg G$$

é **insatisfazível!**

Resolução Proposicional

$$F =_{\text{def}} P_1 \wedge P_2 \wedge \dots \wedge P_n \wedge \neg G$$

- Uma vez definida a tarefa de demonstrar que a fórmula F é insatisfazível, uma segunda característica do sistema de resolução \mathcal{R} é o de se aplicar apenas a fórmulas na sua Forma Normal Conjuntiva (CNF).
- Em abstracto, não distinguindo as premissas da negação da conclusão e convertendo a fórmula F para CNF, o sistema de Resolução permite determinar a satisfazibilidade de uma fórmula

$$F =_{\text{def}} C_1 \wedge C_2 \wedge \dots \wedge C_m$$

em que os vários C_i s são **cláusulas** do tipo

$$A_1 \vee A_2 \vee \dots \vee A_k$$

e em que os vários A_j s são **literais**, isto é fórmulas atómicas (átomos) ou a sua negação, respectivamente denominados de literais positivos ou negativos.

Antes de estudar o sistema R , analisemos a dificuldade do problema de determinar a satisfazibilidade de uma fórmula CNF.

Satisfação Booleana - SAT

$$F =_{\text{def}} C_1 \wedge C_2 \wedge \dots \wedge C_m$$

Satisfação Booleana 1:

- O problema de **Satisfação Booleana** (abreviado vulgarmente para **SAT**) consiste em verificar se é possível descobrir uma interpretação dos átomos que ocorrem numa fórmula CNF de forma a torná-la verdadeira.
- Como a fórmula F é uma **conjunção** de cláusulas, o problema SAT pode ser reformulado como

Satisfação Booleana 2:

- O problema de **Satisfação Booleana** consiste em verificar se é possível descobrir uma interpretação que satisfaça um conjunto de cláusulas

$$\{C_1, C_2, \dots, C_m\}$$

isto é, determinar uma valoração dos átomos que ocorrem nas cláusulas C_i de forma a torná-las todas verdadeiras.

Decidibilidade de SAT

$$S = \{C_1, C_2, \dots, C_m\}$$

- Naturalmente o problema SAT é **decidível**, isto é, é sempre possível em tempo finito determinar se o conjunto de cláusulas C é satisfazível.
- Tudo o que há a fazer é usar o método das tabelas de verdade, e analisar todas as possíveis interpretações ou valorações, isto é todas as combinações de atribuições dos valores $\{V, F\}$ aos átomos das cláusulas.
- Por exemplo o conjunto $S_1 = \{A \vee C, \neg A \vee \neg B, B \vee \neg C\}$ é satisfazível. Na tabela de verdade abaixo, várias linhas correspondem a interpretações que satisfazem as cláusulas, nomeadamente as linhas 4 e 5.

#	A	B	C	$A \vee C$	$\neg A \vee \neg B$	$B \vee \neg C$
1	V	V	V	V	F	V
2	V	V	F	V	F	V
3	V	F	V	V	V	F
4	V	F	F	V	V	V
5	F	V	V	V	V	V
6	F	V	F	F	V	V
7	F	F	V	V	V	F
8	F	F	F	F	V	V

Decidibilidade de SAT

$$S = \{C_1, C_2, \dots, C_m\}$$

- Já o conjunto $S_2 = \{ \neg A \vee C, A \vee B, \neg A \vee \neg C, A \vee \neg B \}$ é insatisfazível como se pode observar na tabela abaixo.

#	A	B	C	$\neg A \vee C$	$A \vee B$	$\neg A \vee \neg C$	$A \vee \neg B$
1	V	V	V	V	V	F	V
2	V	V	F	F	V	V	V
3	V	F	V	V	V	F	V
4	V	F	F	F	V	V	V
5	F	V	V	V	V	V	F
6	F	V	F	V	V	V	F
7	F	F	V	V	F	V	V
8	F	F	F	V	F	V	V

Em qualquer linha da tabela, uma das cláusulas é falsa, pelo que não há interpretações que satisfaçam o conjunto de cláusulas.

Complexidade de SAT

$$S = \{C_1, C_2, \dots, C_m\}$$

- Sendo o problema decidível, é conveniente saber qual a sua complexidade, isto é o “tempo” necessário para resolver uma instância do problema.
- Claramente, e no pior caso, o problema pode ser resolvido por análise de todas as linhas da respectiva tabela.
- Havendo n átomos, existirão 2^n linhas, e portanto a complexidade não é pior que $O(2^n)$, isto é o problema pode ser resolvido num número de passos **exponencial** no número de literais.
- De facto, consegue-se demonstrar que este problema é representativo de uma classe de problemas, denominados **NP-Completo**s, para os quais não se descobriu, e não se espera descobrir, qualquer algoritmo que permita resolver **qualquer** instância do problema em tempo polinomial (isto é de complexidade $O(n^k)$, em que k é um inteiro).
- No entanto, é fácil verificar uma solução em tempo polinomial. No caso de SAT, havendo m cláusulas, essa verificação pode ser feita uma a uma, em tempo $O(m)$.

Complexidade de SAT

- A tabela abaixo mostra uma comparação dos tempos necessários para resolver problemas de complexidade polinomial, $O(n^k)$, e exponencial, $O(k^n)$, assumindo que cada um dos passos de resolução é feito em 1 ns.

n^1 : Pesquisa de elemento num vector; n^2 : Ordenação de um vector; n^3 : Multiplicação de matrizes

n	10	20	30	40	50	60	70
n^1	10 nseg	20 nseg	30 nseg	40 nseg	50 nseg	60 nseg	70 nseg
n^2	100 nseg	400 nseg	900 nseg	1.6 μ seg	2.5 μ seg	3.6 μ seg	4.9 μ seg
n^3	1 μ seg	8 μ seg	27 μ seg	64 μ seg	125 μ seg	216 μ seg	343 μ seg
2^n	1 μ seg	1 mseg	1 seg	18 min	13 dias	37 anos	37 K anos

- O facto de um problema ser NP-completo ou polinomial tem profundas implicações na abordagem à sua resolução, e em geral impõe restrições bastante fortes ao tamanho dos problemas que podem ser resolvidos em “tempo útil”.
- Apesar desta complexidade existem casos especiais em que o problema SAT se pode resolver em tempo polinomial. Um desses casos ocorre quando num problema todas as cláusulas são cláusulas de Horn.

Horn - SAT

Cláusulas de Horn

Um cláusula é de Horn se contém no máximo um literal positivo.

- Por exemplo

- A é uma cláusula de Horn Positiva (facto)
- $\neg A \vee \neg B$ é uma cláusula de Horn Negativa (golo)
- $A \vee \neg B \vee \neg C \vee \neg D$ é uma cláusula de Horn Mista (regra)
- $A \vee B$ **não** é uma cláusula de Horn

- Cláusulas de Horn podem ser escritas na forma de **regras**, em que uma conjunção de átomos (o **corpo** da regra) implica outro átomo (a **cabeça** da regra). Em particular, regras cujo corpo é o átomo T (verdade) correspondem a factos e regras cuja cabeça é o átomo \perp (falso/absurdo) correspondem a golos (questões).

- $A \leftarrow T$ **Facto:** A é verdade
- $\perp \leftarrow A \wedge B$ **Golo:** É Falso que A e B sejam verdade ?
- $A \leftarrow B \wedge C \wedge D$ **Regra:** A é verdade se B , C e D o forem

Algoritmo Horn - SAT

Algoritmo Horn-SAT

Para verificar se existe um **modelo** para uma **conjunto S** de cláusulas **Horn** (isto é, uma interpretação ou valoração dos seus átomos que tornem as cláusulas todas verdadeiras) pode ser usado o seguinte algoritmo:

1. Incluir num conjunto **I**, todos os factos (cabeças de regras cujo corpo é T).
2. Enquanto existir uma regra (ainda não “marcada”) em que todos os átomos do seu corpo estejam incluídos em **I**
 - a) Se a cabeça da regra for um átomo **H** ($\neq \perp$), então incluir esse átomo em **I**, marcar essa regra, e voltar a 2.
 - b) Se a cabeça da regra for o átomo \perp , então **terminar** o algoritmo: **S** não é satisfazível .
3. **S** é satisfazível e um seu modelo **M** é obtido atribuindo o valor **Verdade** a todos os átomos de **I** e o valor **Falso** a todos os restantes átomos de **S**.

Algoritmo Horn-SAT

Exemplo Horn-SAT 1

1. $E \leftarrow C \wedge D$
2. $C \leftarrow A \wedge D$
3. $F \leftarrow A \wedge C$
4. $A \leftarrow T$
5. $D \leftarrow T$
6. $\perp \leftarrow E$

1. $E \vee \neg C \vee \neg D$
2. $C \vee \neg A \vee \neg D$
3. $F \vee \neg A \vee \neg C$
4. A
5. D
6. $\neg E$

1. Inicializa-se uma interpretação I com os factos 4 e 5: $I = \{ A, D \}$.
- 2.i A regra 2 é “marcada” e o literal C é adicionado à interpretação: $I = \{ A, C, D \}$
- 2.ii As regras 1 e 3 são “marcadas” e os literais E e F adicionados: $I = \{ A, C, D, E, F \}$
- 2.iii É selecionada a regra 6 cuja cabeça é o átomo \perp . Logo
 - O conjunto S não é satisfazível!
 - Fica **provado** o golo E : E é consequência tautológica do conjunto S , isto é da **Base de Conhecimentos** constituída pelos factos e regras de S .

Algoritmo Horn-SAT

Exemplo Horn-SAT 2

1. $E \leftarrow C \wedge D$
2. $C \leftarrow A \wedge D$
3. $F \leftarrow A \wedge C$
4. $A \leftarrow B$
5. $A \leftarrow T$
6. $D \leftarrow T$
7. $\perp \leftarrow B$

1. $E \vee \neg C \vee \neg D$
2. $C \vee \neg A \vee \neg D$
3. $F \vee \neg A \vee \neg C$
4. $A \vee \neg B$
5. A
6. D
7. $\neg B$

1. Inicializa-se uma interpretação I com os factos 5 e 6: $I = \{A, D\}$.
- 2.i A regra 2 é “marcada” e o literal C é adicionado à interpretação: $I = \{A, C, D\}$
- 2.ii As regras 1 e 3 são “marcadas” e os literais E e F adicionados: $I = \{A, C, D, E, F\}$
3. Nenhuma outra regra (3 ou 6) é seleccionada. Logo
 - O conjunto S é **satisfazível**, pelo modelo $M = \{A, \neg B, C, D, E, F\}$.
 - **Não se prova** o golo B : B **não** é consequência tautológica dos factos e regras da **Base de Conhecimento** incluída em S . De facto, existe uma interpretação, M , que satisfaz os factos e regras de S mas em que o átomo B é falso.

Correcção do Algoritmo Horn-SAT

- Para provar a correcção do algoritmo Horn-SAT é conveniente considerar o conjunto de átomos a que foi atribuído o valor verdade em cada “fase” do algoritmo. Mais especificamente podemos definir indutivamente os conjuntos T_i da seguinte forma

- **Cláusula Base:** T_0 é o conjunto $\{T\}$
- **Cláusula de Indução:** T_{i+1} é o conjunto de átomos de T_i unido com os literais que estão na cabeça de regras em que todos os átomo do seu corpo estejam em T_i .

$$T_0 = \{T\}$$

$$T_1 = \{T, A, D\}$$

$$T_2 = \{T, A, D, C\}$$

$$T_3 = \{T, A, D, C, E, F\}$$

$$T_4 = \{T, A, D, C, E, F, \perp\}$$

$$T_k = T_4 \quad (k > 4)$$

$$1. \quad E \leftarrow C \wedge D$$

$$2. \quad C \leftarrow A \wedge D$$

$$3. \quad F \leftarrow A \wedge C$$

$$4. \quad A \leftarrow T$$

$$5. \quad D \leftarrow T$$

$$6. \quad \perp \leftarrow E$$

- Como se verifica no exemplo acima, temos

- $\forall i \quad T_i \subseteq T_{i+1}$
- A sequência de conjuntos T_i é finita: $\exists N \forall k (k > N \rightarrow T_k = T_N)$. $N = 4$ no exemplo.
- A não satisfação de S corresponde a inclusão do átomo \perp no conjunto T_N ,

Correcção do Algoritmo Horn-SAT

- A correcção do algoritmo Horn-SAT corresponde a garantir que um conjunto **S** de cláusulas Horn é satisfazível **sse** o algoritmo não parar no passo 2.b ou seja :

Teorema: **S** é satisfazível **sse** o conjunto T_N **não** contiver o átomo \perp .

- A condição necessária pode ser provada por indução.

\Rightarrow Se **S** é satisfazível, então o átomo \perp não pertence a T_N .

Passo Base: Se **S** for satisfazível, então o conjunto T_0 não contem o átomo \perp .

- Por definição de T_0 .

Passo de Indução: Se **S** for satisfazível e o conjunto T_i não contiver o átomo \perp , então o conjunto T_{i+1} também não o contém.

- Qualquer átomo **H** de T_{i+1} que não seja elemento de T_i é obtido pela aplicação de uma regra de S da forma

$$H \leftarrow B_1 \wedge B_2 \wedge \dots \wedge B_k$$

em que todos os átomos $B_1 \dots B_k$ pertencem a T_i e portanto têm de ser verdadeiros em qualquer interpretação que satisfaça **S**. Mas então terá de ser $H \neq \perp$, caso contrário existiria uma cláusula $\neg B_1 \vee \neg B_2 \vee \dots \vee \neg B_k$ que não seria satisfeita, contrariamente à hipótese de ser **S** satisfazível. Logo o átomo \perp não pertence a T_{i+1} .

Correcção do Algoritmo Horn-SAT

Teorema: S é satisfazível **sse** o conjunto T_N **não** contiver o átomo \perp .

- Passemos agora à condição suficiente.

<= Se o átomo \perp não pertence a T_N então S é satisfazível.

- Se o átomo \perp não pertence a T_N então qualquer interpretação M que apenas torne verdadeiros os átomos que pertencem a T_N satisfaz S . Com efeito, qualquer cláusula de S na forma

$$H \leftarrow T \Leftrightarrow H$$

é satisfeita, pois o átomo H pertence a T_1 e $T_1 \subseteq T_2 \dots \subseteq T_N$. Por outro lado, todas as cláusulas de S da forma

$$\perp \leftarrow B_1 \wedge B_2 \wedge \dots \wedge B_k \Leftrightarrow \neg B_1 \vee \neg B_2 \vee \dots \vee \neg B_k$$

também são satisfeitas pois, por construção do modelo M , é garantido que pelo menos um dos B_j é falso. Finalmente, as cláusulas da forma

$$H \leftarrow B_1 \wedge B_2 \wedge \dots \wedge B_k \Leftrightarrow H \vee \neg B_1 \vee \neg B_2 \vee \dots \vee \neg B_k$$

também são satisfeitas:

- ou porque todos os B_j são membros de um T_i e então H é membro de $T_{i+1} \subseteq T_N$;
- ou porque algum dos B_j é falso.

Complexidade do Algoritmo Horn-SAT

- O algoritmo Horn-SAT tem complexidade polinomial, isto é, para um conjunto de cláusulas de Horn, com m cláusulas e n átomos distintos, determina se S é satisfazível em tempo polinomial em m e n . Sem perda de generalidade pode assumir-se que:
 - Existem k factos ($0 < k < n$), caso contrário S é satisfazível (porquê?)
- Neste caso, no primeiro ciclo do passo 2 o algoritmo analisará no máximo $m-k$ cláusulas. Se incluir o átomo \perp em I , S não é satisfazível e o algoritmo termina. Caso contrário, no próximo ciclo haverá $m-k-1$ cláusulas para analisar e $k+1$ átomos em I .
- Nos vários passos, o número de cláusulas a analisar será pois $m-k$, $m-k-1$, ..., e portanto o número máximo de passos do algoritmo é $m-k$.
- Em cada passo serão analisadas até $m-k$ cláusulas (serão menos se uma delas fizer incluir o átomo \perp em S), portanto haverá um máximo de $(m-k)^2$ avaliações.
- Em cada avaliação é verificado se os átomos do corpo da cláusula (em número de n , no máximo) estão no conjunto I , que terá no máximo n literais; logo a complexidade de cada avaliação é inferior a $n \times n$.
- Assim, a complexidade do algoritmo Horn-SAT é inferior a $(m-k)^2 \times n^2$ ou seja $O(m^2 n^2)$.