Método de *Tableaux*

Na construção de derivações no sistema de cálculo de sequentes:

- Na aplicação de cada regra, só a manipulação referente à *fórmula principal* é informativa.
- A cópia dos contextos revela-se assim uma actividade burocrática: destina-se a propagar informação ao longo da árvore.
- Na mecanização do processo de construção de derivações, é conveniente “destilar a sua essência”.
- O **método Tableaux** pode ser explicado como constituindo essa essência na construção de derivações em cálculo de sequentes.

Método de *Tableaux*

Método de *Tableaux*:

- O objectivo do método é o de construir uma árvore que corresponda à estrutura da derivação de um dado sequente no *cálculo de sequentes*.
- Cada nó da árvore é etiquetado com fórmulas prefixadas com um sinal ($+$ ou $-$). Fórmulas positivas correspondem a fórmulas que ocorrem no lado direito dos sequentes, fórmulas negativas correspondem às do lado esquerdo.
- A árvore inicial contém um único nó com as fórmulas do sequente que se pretende validar.
- A árvore é expandida até que a validade seja estabelecida ou se atingir um ponto onde não seja possível prosseguir com o processo de expansão.

Expansão da árvore

- O processo de construção da árvore segue **regras** que capturam o tratamento dado à fórmula principal em cada uma das regras correspondentes no cálculo de sequentes.
- Os **caminhos** de um *Tableaux* são sequências de nós da raiz até às folhas.
- Um caminho diz-se **fechado** quando contém nós $+P$ e $-P$ (para uma dada fórmula P). Os caminhos fechados correspondem a percursos da derivação fechados com a regra *Axioma*. Vamos assinalar os caminhos fechados com o símbolo $\otimes$.
- O processo de construção da árvore conclui-se quando todas as fórmulas não atómicas tiverem sido decompostas pelas regras.
- Se, uma vez concluído o processo de construção da árvore, todos os caminhos forem fechados, o sequente original é **válido**.
- Caso contrário, o sequente é **refutável**. Cada caminho não fechado dá origem a um modelo que refuta o sequente: o conjunto de todas as fórmulas atómicas positivas do caminho.

Regras de Construção

- Regras α (disjuntivas): correspondem às regras do cálculo de sequentes com duas premissas. Estas regras bifurcam a árvore de derivação.

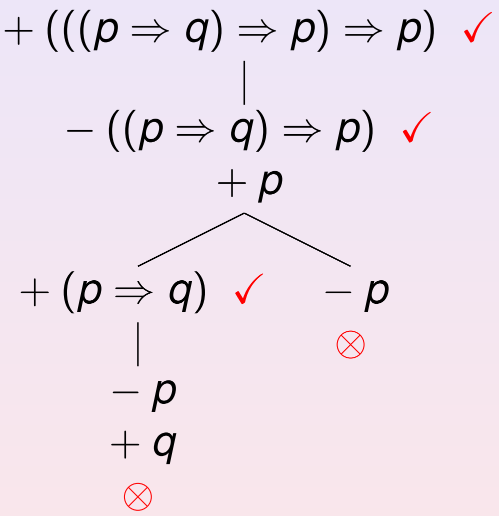
α	α_1	α_2
$+(X \wedge Y)$	$+X$	$+Y$
$-(X \vee Y)$	$-X$	$-Y$
$-(X \Rightarrow Y)$	$+X$	$-Y$

- Regras β (conjuntivas): correspondem às regras com uma única premissa. Estas regras não bifurcam a árvore (mas podem dar origem a mais do que uma fórmula)

β	β_1, β_2
$-(X \wedge Y)$	$-X, -Y$
$+(X \vee Y)$	$+X, +Y$
$+(X \Rightarrow Y)$	$-X, +Y$
$+(\neg X)$	$-X$
$-(\neg X)$	$+X$

Exemplo de construção de um *Tableaux*

- Pretende-se verificar a fórmula $(((p \Rightarrow q) \Rightarrow p) \Rightarrow p)$:



$+\,(((p \Rightarrow q) \Rightarrow p) \Rightarrow p)$ ✓

$-\,((p \Rightarrow q) \Rightarrow p)$ ✓

$+\,p$

$+\,(p \Rightarrow q)$ ✓ $\qquad -\,p$ ⊗

$-\,p$

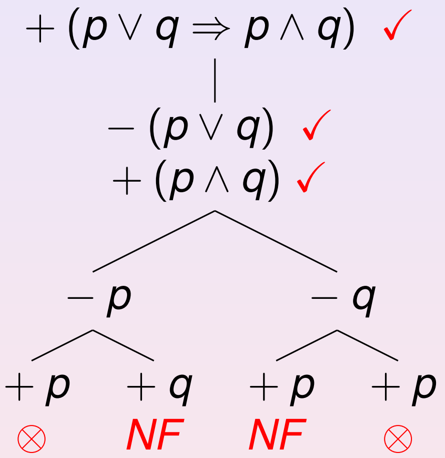
$+\,q$

⊗

- Como todos os caminhos são *fechados*, a fórmula é uma tautologia.

Outro Exemplo

- Verificar a fórmula $(p \vee q \Rightarrow p \wedge q)$:



$+\,(p \vee q \Rightarrow p \wedge q)$ ✓

$-\,(p \vee q)$ ✓

$+\,(p \wedge q)$ ✓

$-\,p \qquad -\,q$

$+\,p \quad +\,q \quad +\,p \quad +\,p$

⊗ *NF* *NF* ⊗

- Como nem todos os caminhos são fechados, a fórmula é *refutável*.
- Dos caminhos não fechados podemos extrair modelos que refutam a fórmula — neste caso temos os modelos $\{q\}$ e $\{p\}$.

Formas Normais e Clausais

É normalmente vantajoso estabelecer a validade de fórmulas lógicas em dois passos:

1. Transformando a fórmula dada numa semânticamente equivalente e pertencente a uma classe sintáctica restrita (ditas *formas normais*);
2. Aplicar um algoritmo que permita determinar a validade das fórmulas na classe sintáctica considerada.

A vantagem é que se irá poder tirar partido da classe restrita de fórmulas para desenvolver algoritmos consideravelmente mais eficientes do que os estudados até agora.

Forma Normal Negativa

Definição

Uma fórmula lógica diz-se na ***forma normal negativa*** *(FNN) quando não faz uso da conectiva $\Rightarrow$ e a negação só é aplicada a símbolos proposicionais.*

Para, a partir de uma fórmula lógica arbitrária, se obter uma equivalente na forma normal negativa, poder-se-á proceder da seguinte forma:

1. Reescrevem-se todas as ocorrências da implicação por aplicação da equivalência semântica

$$A \Rightarrow B \equiv \neg A \vee B$$

2. Aplicam-se repetidamente as *leis de Morgan* para reescrever as negações de fórmulas estruturadas.

$$\begin{aligned} \neg(A \wedge B) &\equiv \neg A \vee \neg B \\ \neg(A \vee B) &\equiv \neg A \wedge \neg B \\ \neg(\neg A) &\equiv A \end{aligned}$$

Forma Normal Conjuntiva

Definição

Diz-se que uma fórmula é uma ***forma normal conjuntiva*** *(FNC) quando está na forma normal negativa e a conectiva $\wedge$ nunca é utilizada em argumentos da conectiva $\vee$.*

- As FNCs são da forma

$$(a_1 \vee a_2 \vee \ldots \vee a_k) \wedge \ldots \wedge (b_1 \vee \ldots \vee b_l)$$

- É normal designarmos por **literais** os símbolos proposicionais e as suas negações. Às disjunções de literais $(a_1 \vee \ldots \vee a_k)$ damos o nome de **cláusulas** (daí dizer-se que as FNC é uma *forma clausal*). A cláusula vazia é denotada por $\perp$.
- Para, a partir de uma fórmula na FNN, se obter uma equivalente na FNC podemos aplicar repetidamente a distributividade do $\vee$ sobre o $\wedge$ para remover as conjunções que ocorrem como argumentos das disjunções.

$$\begin{aligned} A \vee (B \wedge C) &\equiv (A \vee B) \wedge (A \vee C) \\ (A \wedge B) \vee C &\equiv (A \vee C) \wedge (B \vee C) \end{aligned}$$

Exemplos

- Para determinar a forma normal conjuntiva de $((p \Rightarrow q) \Rightarrow p) \Rightarrow p$ procedemos da seguinte forma:
  1. Cálculo da FNN:
     1. Remoção das implicações:

$$\begin{aligned} ((p \Rightarrow q) \Rightarrow p) \Rightarrow p &\equiv \neg((p \Rightarrow q) \Rightarrow p) \vee p \\ &\equiv \neg(\neg(p \Rightarrow q) \vee p) \vee p \\ &\equiv \neg(\neg(\neg p \vee q) \vee p) \vee p \end{aligned}$$

     2. Remoção de negações a fórmulas estruturadas:

$$\begin{aligned} \neg(\neg(\neg p \vee q) \vee p) \vee p &\equiv \neg((p \wedge \neg q) \vee p) \vee p \\ &\equiv (\neg(p \wedge \neg q) \wedge \neg p) \vee p \\ &\equiv ((\neg p \vee q) \wedge \neg p) \vee p \end{aligned}$$

  2. Cálculo da FNC

$$((\neg p \vee q) \wedge \neg p) \vee p \equiv ((\neg p \vee q \vee p) \wedge (\neg p \vee p)$$

Propriedades da FNC

- A forma rígida das FNCs permite representações simplificadas — uma representação habitual é a de as representarmos como *listas de listas de literais*[1] (as listas interiores representam as cláusulas). Desta forma, a FNC do exemplo apresentado seria $[[p, \neg p, q], [p, \neg p]]$.
- As cláusulas que contenham simultaneamente um símbolo proposicional e a sua negação dizem-se **fechadas**
- A validade de uma FNC não é afectada quando se omitem cláusulas fechadas. Este facto é consequência directa de as cláusulas fechadas representarem disjunções de literais com a tautologia $P \vee \neg P$ (para um dado símbolo proposicional P).
- Como corolário temos que a FNC composta unicamente por cláusulas fechadas é uma **tautologia**. A FNC vazia (sem qualquer cláusula) é denotada por $\top$.

---

[1] Poderiam até ser conjuntos de conjuntos. Mas, numa perspectiva computacional, estes são normalmente representados como listas.

Forma Normal Disjuntiva

De forma dual, também podemos considerar a *forma normal disjuntiva*.

Definição

*Diz-se que uma fórmula é uma **forma normal disjuntiva** (FND) quando está na forma normal negativa e a conectiva $\vee$ nunca é utilizada em argumentos da conectiva $\wedge$.*

- As **cláusulas** das FNDs serão agora conjunções. A cláusula vazia é denotada por $\top$.
- Para, a partir de uma fórmula na FNN, se obter uma equivalente na FND podemos aplicar repetidamente a distributividade do $\wedge$ sobre o $\vee$ para remover as disjunções que ocorrem como argumentos das conjunções.

$$\begin{aligned} A \wedge (B \vee C) &\equiv (A \wedge B) \vee (A \wedge C) \\ (A \vee B) \wedge C &\equiv (A \wedge C) \vee (B \wedge C) \end{aligned}$$

Propriedades da FND

As FNDs exibem propriedades **duais** das FNCs. Nomeadamente,

- A validade de uma FND não é afectada quando se omitem cláusulas fechadas. Este facto é consequência directa de as cláusulas fechadas representarem conjunção de literais com a tautologia $P \wedge \neg P$ (para um dado símbolo proposicional P).
- Como corolário temos que a FND composta unicamente por cláusulas fechadas é uma **contradição**. A FND vazia (sem qualquer cláusula) é denotada por $\perp$.

Doravante consideramos unicamente as propriedades e métodos para a FNC. Todos os resultados podem ser dualizados para a FND.

Cláusulas como sequentes atómicos

- Se, numa cláusula de uma FNC, separarmos os literais negativos (proposições negadas) dos positivos (proposições não negadas), podemos exprimir essa cláusula da seguinte forma:

$$\begin{aligned} &\neg a_1 \vee \ldots \vee \neg a_k \vee a_{k+1} \vee \ldots \vee a_n \\ &\equiv \neg(a_1 \wedge \ldots \wedge a_k) \vee a_{k+1} \vee \ldots \vee a_n \\ &\equiv a_1 \wedge \ldots \wedge a_k \Rightarrow a_{k+1} \vee \ldots \vee a_n \end{aligned}$$

- Por outras palavras, podemos ver um cláusula como um **sequentes atómico** (sequente onde todas as fórmulas são símbolos proposicionais).

$$a_1, \ldots, a_k \vdash a_{k+1}, \ldots, a_n$$

Resolução

- Considerem-se duas cláusulas (vistas como sequentes atómicos), em que numa surge uma dado símbolo proposicional no lado direito e no outra surge o mesmo símbolo no lado esquerdo:

$$C_1 = a_1, \ldots, a_k \vdash a_{k+1}, \ldots, a_m, p$$
$$C_2 = p, b_1, \ldots, b_l \vdash b_{l+1}, \ldots, b_n$$

- Por aplicação da *regra de corte* podemos obter uma nova cláusula $C_r = a_1, \ldots, a_k, b_1, \ldots, b_l \vdash a_{k+1}, \ldots, a_m, b_{l+1}, \ldots, b_n$ em que se elimina o símbolo proposicional p:

$$\frac{a_1, \ldots, a_k \vdash a_{k+1}, \ldots, a_m, p \qquad p, b_1, \ldots, b_l \vdash b_{l+1}, \ldots, b_n}{a_1, \ldots, a_k, b_1, \ldots, b_l \vdash a_{k+1}, \ldots, a_m, b_{l+1}, \ldots, b_n} \text{(Corte)}$$

- Dizemos que C_r é obtida por **resolução** das cláusulas C_1 e C_2 (C_r é por isso o **resolvente** das cláusulas C_1 e C_2).

Resolução (cont.)

- A adequação da regra de corte garante-nos que C_r será válida sempre que C_1 e C_2 o forem. Assim, se C_1 e C_2 forem cláusulas de uma FNC ϕ, podemos adicionar C_r sem afectar a sua validade, i.e.

$$\phi \equiv \phi \wedge C_r$$

- Em particular, se C_1 e C_2 forem formadas por um único literal (ou seja, $C_1 = p$ e $C_2 = \neg p$), então o seu resolvente será a cláusula vazia $\perp$, e estabelecemos a refutação de ϕ

$$\phi \equiv \phi \wedge \perp \equiv \perp$$

- Este facto fornece-nos uma estratégia para **refutar** uma fórmula ϕ na FNC: *procurar derivar $\perp$ por resolução das cláusulas em ϕ.*

Teorema da Resolução

Teorema

Seja ϕ uma fórmula na FNC e S o conjunto das suas cláusulas (vistas como sequentes). Essa fórmula será uma contradição se e só se for possível derivar o sequente vazio a partir de S utilizando unicamente a regra de corte.

Demonstração.

($\Longleftarrow$) Resulta directamente da adequação da regra de corte.
($\Longrightarrow$) Resultado mais complicado porque corresponde a um resultado de completude. Esboço será apresentado na aula teórica (detalhes podem ser encontrados em *Proof Theory and Automated Deduction, pag. 49–51*). □

Algoritmo de Robinson

Na prática, a aplicação do método da resolução é realizada pelo que é designado por *Algoritmo de Robinson*:

Definição

Seja ϕ uma fórmula na FNC e S_0 o conjunto das suas cláusulas.

1. $S = S_1 = S_0$
2. $i = 1$
3. *Se* $\perp \in S_i$ *então retorna "CONTRADIÇÃO"*
4. $S_{i+1} = \mathrm{Res}(S_i, S) \setminus S$
5. $i = i + 1; S = S \cup S_i$
6. *Se* $S_i \neq \emptyset$ *então volta ao passo* **??**.
7. *retorna "ADMISSÍVEL"*

onde $\mathrm{Res}(X, Y)$ denota o conjunto de resolventes não fechados para fórmulas $x \in X, y \in Y$.

Exemplo de Aplicação (1)

Considere-se o conjunto de cláusulas $S_0 = \{[a, \neg b], [\neg a, b]\}$

1. $S = S_1 = S_0$
2. $S_2 = \mathrm{Res}(S_1, S) = \{\}$

	$[a, \neg b]$	$[\neg a, b]$
$[a, \neg b]$		$[a, \neg a], [b, \neg b]$
$[\neg a, b]$	$[a, \neg a], [b, \neg b]$	

3. $S_2 = \emptyset$, logo Retorna "ADMISSÍVEL"

Exemplo de Aplicação (2)

Considere-se o conjunto de cláusulas $S_0 = \{[a, b], [\neg a, b], [a, \neg b], [\neg a, \neg b]\}$

1. $S = S_1 = S_0$
2. $S_2 = \mathrm{Res}(S_1, S) = \{[a], [b], [\neg a], [\neg b]\}$

	$[a, b]$	$[\neg a, b]$	$[a, \neg b]$	$[\neg a, \neg b]$
$[a, b]$		$[b]$	$[a]$	$[b, \neg b], [a, \neg a]$
$[\neg a, b]$	$[b]$		$[b, \neg b], [a, \neg a]$	$[\neg a]$
$[a, \neg b]$	$[a]$	$[b, \neg b], [a, \neg a]$		$[\neg b]$
$[\neg a, \neg b]$	$[b, \neg b], [a, \neg a]$	$[\neg a]$	$[\neg b]$	

3. $S_3 = \mathrm{Res}(S_2, S_0 \cup S_1 \cup S_2) = \{[]\}$

S_2	$\mathrm{Res}(-, S = S_0 \cup S_1 \cup S_2)$
$[a]$	$\{[b], [\neg b], []\}$
$[b]$	$\{[a], [\neg a], []\}$
$[\neg a]$	$\{[b], [\neg b], []\}$
$[\neg b]$	$\{[a], [\neg a], []\}$

4. $\perp \in S_3$, logo Retorna "CONTRADIÇÃO"

Observações

- O *Algoritmo de Robinson* é muito pouco efectivo como procedimento para determinar a validade de uma FNC. De facto, muito do trabalho realizado é descartado por não trazer informação útil ao procedimento (e.g. cálculo de resolventes fechados ou já pertencentes ao conjunto de cláusulas conhecidas).
- No entanto, veremos adiante que a adaptação deste algoritmo para a lógica de primeira ordem está na base do mecanismo de inferência da linguagem *Prolog*.