SISTEMA DE TABLEAUX PARA LÓGICA PROPOSICIONAL

Linguagem: fórmulas da lógica proposicional

As regras do sistema de tableaux proposicional, aplicadas a uma fórmula α**,** permitem obter, em forma de árvore, uma forma normal disjuntiva de α, isto é $\varphi_1 \vee \varphi_2 \vee .. \vee \varphi_n$ onde φ_i é da forma $\tau_1 \wedge \tau_2 \wedge ... \wedge \tau_k$ e τ_j's são literais. Em cada ramo da árvore aparece os literais que ocorrem em φ_i.
Note que α é insatisfatível sse φ_i é insatisfatível , para i=1…n e para cada i, φ_i é insatisfatível sse contem uma fórmula β e $\neg\, \beta$, para algum β.
Assim, se α é insatisfatível, em cada ramo vai aparecer β e $\neg\, \beta$, para algum β.
Chamaremos de *Tableau* para uma fórmula α uma árvore, com α na raíz, obtida pela aplicação das regras de inferência que daremos a seguir. Antes, porém, daremos as seguintes definições.

Definição
Um ramo ρ de um tableau é <u>fechado</u> se nele aparecem β e $\neg\, \beta$, para alguma fórmula β.

Definição
Um tableau T é <u>fechado</u> se cada ramo de T for fechado.

O sistema de tableaux proposicional tem dois tipos de regras: as que geram árvores degeneradas (ramos), que chamaremos do tipo A, e regras que geram árvores pròpriamente ditas, que chamaremos de regras do tipo B.

Regras de inferência:
Tipo A (gera descendentes lineares)

$$\frac{\alpha \wedge \beta}{\begin{array}{c}\alpha\\ \beta\end{array}} \qquad \frac{\neg\,(\alpha \vee \beta)}{\begin{array}{c}\neg\,\alpha\\ \neg\,\beta\end{array}} \qquad \frac{\neg\,(\alpha \rightarrow \beta)}{\begin{array}{c}\alpha\\ \neg\,\beta\end{array}} \qquad \frac{\neg\,\neg\,\alpha}{\alpha}$$

Tipo B (gera árvores)

$$\frac{\alpha \vee \beta}{\alpha \mid \beta} \qquad \frac{\alpha \rightarrow \beta}{\neg\alpha \mid \beta} \qquad \frac{\neg\,(\alpha \wedge \beta)}{\neg\,\alpha \mid \neg\,\beta} \qquad \frac{\neg\,(\alpha \leftrightarrow \beta)}{\alpha \wedge \neg\,\beta \mid \neg\alpha \wedge \beta}$$

Observe que podemos relacionar a aplicação das regras de inferência com os passos do algoritmo de obtenção de forma normal disjuntiva da seguinte forma: o resultado das regras que bifurcam pode ser considerado como a disjunção das fórmulas sobre às quais as regras são aplicadas; o resultado das regras podem ser considerados como os passos de eliminação dos conectivos $\rightarrow$ e $\leftrightarrow$ e da movimentação do $\neg$ para o interior das fórmulas.

Vejamos através de um exemnplo a relação da aplicação das regras com a obtenção de uma forma normal disjuntiva.

Considere a fórmula $P\rightarrow[(Q\vee R)\wedge S]$. Aplicando-se o algoritmo para obtenção de forma normal disjuntiva podemos obter a fórmula $\neg P \vee (Q \wedge S) \vee (R \wedge S)$.

Agora, começando com a fórmula $P\rightarrow[(Q\vee R)\wedge S]$ e aplicando-se as regras de tableaux podemos obter a seguinte árvore:

1.	$P\rightarrow[(Q\vee R)\wedge S]$		
$\underline{\neg P}$	$(Q\vee R)\wedge S$		regra B em 1
	$Q \vee R$		regra A em $(Q\vee R)\wedge S$
	$\underline{S}$		
	$\underline{Q}$	$\underline{R}$	regra B em $(Q\vee R)$

As fórmulas grafadas em cada ramo correspondem aos literais que ocorrem em cada disjunto da fórmula normal disjuntiva de $\neg P \vee (Q \wedge S) \vee (R \wedge S)$.

O sistema de tableaux vai ser utilizado de forma refutacional, isto é para determinarmos se uma fórmula é válida, iniciamos um tableau com sua negação e teremos a resposta positiva se for possível obtermos um tableau fechado.

Exemplo 1

Determinar que $P \rightarrow (Q \rightarrow P)$ é válida por tableau

1. $\neg(P \rightarrow (Q \rightarrow P))$
2. P (regra tipo A,1)
3. $\neg(Q \rightarrow P)$ (regra tipo A,1)
4. Q (regra tipo A,3)
5. $\neg P$ (regra tipo A,3)

O tableau acima é fechado (só tem um ramo e neste se encontram P e $\neg P$)

Exemplo 2

O mesmo para $P \rightarrow (Q \rightarrow (P \wedge Q))$

1. $\neg (P \rightarrow (Q \rightarrow (P \wedge Q)))$

2 . P

3 $\neg (Q \rightarrow (P \wedge Q)))$

4 Q

5. $\neg(P \wedge Q)))$

/\

6 $\neg P$ $\neg Q$

O tableau acima é fechado (tem dois ramos , num temos P e ¬P, e no outro Q e ¬ ¬Q)