aula 27: Naturalmente

No caderno perdido de Hilbert (que continuou sendo chamado de caderno perdido mesmo depois de ter sido encontrado) havia regras para os conectivos $\wedge$ e $\vee$.

`[H3]` $p \to (q \to (p \wedge q))$

`[H4]` $(p \wedge q) \to p$

`[H5]` $(p \wedge q) \to q$

`[H6]` $p \to (p \vee q)$

`[H7]` $q \to (p \vee q)$

`[H8]` $(p \to r) \to ((q \to r) \to ((p \vee q) \to r)))$

Joé decidiu brincar um pouco com essas regras:

Esses símbolos $\wedge$ e $\vee$ parecem funcionar como "E" e "OU" do português. Então, deixa eu ver se consigo mostrar algumas coisas.

1. $(A \wedge B) \to (A \vee B)$

Essa é fácil. Basta usar a transformação `Trans.` *da transitividade junto com* `H4` *e* `H6`

$$\xrightarrow{\texttt{H4}} (A \wedge B) \to A$$

$$\xrightarrow{\texttt{H6}} A \to (A \vee B)$$

$$\xrightarrow{\texttt{Trans.}} (A \wedge B) \to (A \vee B)$$

Certo. Vamos tentar mais uma.

2. $(A \vee A) \to A$

Também é fácil. Basta ver que essa regra parece com o final de `H8`.

$$\xrightarrow{\texttt{H8}} (A \to A) \to ((A \to A) \to ((A \vee A) \to A)$$

Mas $A \to A$ nós sabemos construir. Então fica assim

$$\xrightarrow{\texttt{H8}} (A \to A) \to ((A \to A) \to ((A \vee A) \to A))$$

$$\xrightarrow{\texttt{R3}} (A \to A) \to ((A \vee A) \to A)$$

$$\xrightarrow{\texttt{R3}} (A \vee A) \to A$$

3. $(A \to B) \to ((A \to C) \to (A \to (B \wedge C)))$

O final dessa regra está me dizendo que de A *consigo chegar em* $(B \wedge C)$*. Mas regra* `H3` *me diz como chegar em uma conjunção. Vou tentar usá-la.*

$$\xRightarrow{\texttt{H3}} B \to (C \to (B \wedge C)))$$

Agora eu tenho que colocar o A *na frente de todo mundo. Vou usar* `L1`*:*

$$\xRightarrow{\texttt{L1}} (A \to B) \to (A \to (C \to (B \wedge C)))$$

Mas falta distribuir o A *na segunda parte. Vou usar* `Dist 2.0`*:*

$$\xRightarrow{\texttt{Dist 2.0}} (A \to B) \to ((A \to C) \to (A \to (B \wedge C)))$$

Cheguei onde eu queria. Gostei dessa regra. Vou chamá-la de `J1`*. Se a Liduína pode ter uma regra, por quê eu não posso?*

Como toda boa regra, deve haver uma transformação correspondente. Vou chamar de `J'`*.*

$$(p \to q) \xRightarrow{\texttt{J'}} (p \to r) \to (p \to (q \wedge r))$$

Mas o que essa transformação me dá é outra regra, que poderiamos desmontar

$$(p \to q) \ e \ (p \to q) \Longrightarrow (p \to (q \wedge t))$$

Isso parece mais intuitivo, vou dar a essa última transformação o mesmo nome da regra que a originou.

$$(p \to q) \ e \ (p \to q) \xRightarrow{\texttt{J1}} (p \to (q \wedge t))$$

Interessante, essa transformação transforma duas regras em uma só. Como o modus ponens. A Liduína tinha umas dessas, mas a maioria transformava uma em outra.

Avante!

4. $(A \wedge (B \wedge C)) \to C$

As regras `H4` *e* `H5` *dizem que posso desmontar uma conjunção (um "E",* $\wedge$*). Mas agora eu tenho uma conjunção dentro de outra conjunção. Será* *que consigo tirar o C daí? Vou tentar demontar por partes. A regra* `H4` *me dá*

$$\xRightarrow{\texttt{H4}} (A \wedge (B \wedge C)) \rightarrow A$$

e `H5` *me dá*

$$\xRightarrow{\texttt{H5}} (A \wedge (B \wedge C)) \rightarrow (B \wedge C)$$

Mas, novamente, `H4` *me dá*

$$\xRightarrow{\texttt{H4}} (B \wedge C) \rightarrow B$$

Aplicando transitividade, temos

$$\xRightarrow{\texttt{Trans.}} (A \wedge (B \wedge C)) \rightarrow B$$

E `H5` *me dá*

$$\xRightarrow{\texttt{H4}} (B \wedge C) \rightarrow C$$

Aplicando transitividade, eu consigo

$$\xRightarrow{\texttt{Trans.}} (A \wedge (B \wedge C)) \rightarrow C$$

Interessante. E consigo desmontar $(A \wedge (B \wedge C))$ completamente. Mas a regra `J1` *pode me ajudar a montar de volta. Por exemplo, de*

$$\Longrightarrow (A \wedge (B \wedge C)) \rightarrow A$$

e

$$\Longrightarrow (A \wedge (B \wedge C)) \rightarrow B$$

conseguimos

$$\xRightarrow{\texttt{J1}} (A \wedge (B \wedge C)) \rightarrow (A \wedge B)$$

E com mais

$$\Longrightarrow (A \wedge (B \wedge C)) \rightarrow C$$

conseguimos

$$\xRightarrow{\texttt{J1}} (A \wedge (B \wedge C)) \rightarrow (A \wedge C)$$

Curioso, isso já me permite observar que eu consigo regras que me permitem mudar a posição dos parênteses. Pois dá para usar `J1` *com essas que já encontrei para construir*

$$\xRightarrow{\texttt{J1}} (A \wedge (B \wedge C)) \rightarrow ((A \wedge B) \wedge C)$$

Mas isso é a associatividade! Há!

E a comutatividade? É fácil. A regra `J1` *me dá, substituindo* p *por* $A \wedge B$,

$$\xRightarrow{\texttt{J1}} ((A \wedge B) \rightarrow B) \rightarrow (((A \wedge B) \rightarrow A) \rightarrow ((A \wedge B) \rightarrow (B \wedge A)))$$

Fazendo modus ponens com $H4$ *e* $H5$ *temos*

$$\xRightarrow{\texttt{MP}} (A \wedge B) \rightarrow (B \wedge A)$$

Mas, será que se eu tiver três letras eu consigo trovar a ordem delas? Para $(A \wedge (B \wedge C))$, *eu já consegui desmontar tudo, então é só juntar do na ordem que eu quiser. De fato, eu já consegui*

$$\Longrightarrow (A \wedge (B \wedge C)) \rightarrow (A \wedge C)$$

e já consegui,

$$\Longrightarrow (A \wedge (B \wedge C)) \rightarrow B$$

`J1` *vai me dar*

$$\xRightarrow{\texttt{J1}} (A \wedge (B \wedge C)) \rightarrow ((A \wedge C) \wedge B)$$

Resumindo, uma vez que eu tenho

$$\Longrightarrow (A \wedge (B \wedge C)) \rightarrow A$$

$$\Longrightarrow (A \wedge (B \wedge C)) \rightarrow B$$

$$\Longrightarrow (A \wedge (B \wedge C)) \rightarrow C$$

posso usar `J1` *para formar todas as conjunções possíveis com duas letras e todas as possíveis com três letras, em qualquer ordem e colocando os parênteses em qualquer posição.*

Ou seja, eu consigo um bocado de regras que me permitem passar de $(A \wedge (B \wedge C))$ *para qualquer arranjo dessas letras.*

Mas, do mesmo jeito que eu consigo desmontar $(A \wedge (B \wedge C))$, eu consigo desmontar os outros arranjos. Por exemplo, considere $(C \wedge (B \wedge A))$. Não é difícil ver que podemos construir as seguintes regras

$$\Longrightarrow (C \wedge (B \wedge A)) \to C$$

$$\Longrightarrow (C \wedge (B \wedge A)) \to B$$

$$\Longrightarrow (C \wedge (B \wedge A)) \to A$$

E daqui é fácil ver, repetindo o que fizemos acima que conseguimos

$$\Longrightarrow (C \wedge (B \wedge A)) \to (A \wedge (B \wedge C))$$

$$\Longrightarrow (C \wedge (B \wedge A)) \to ((B \wedge A) \wedge C))$$

$$\vdots$$

Isto é, de qualquer arrando de A, B e C com conjunção eu consigo formar qualquer outro.

Dá a impressão que, quando temos várias conjunçoões, ordem das coisas e a posição dos parênteses não tem importância...

5. Eu acho que eu percebi uma coisa. Para fazer

$$(A \wedge (B \wedge C)) \to C$$

eu parti disso

$$\xRightarrow{\texttt{H5}} (B \wedge C) \to C$$

e disso

$$\xRightarrow{\texttt{H5}} (A \wedge (B \wedge C)) \to (B \wedge C)$$

e usei transitividade.

Mas, olhando para

$$\xRightarrow{\texttt{H5}} (B \wedge C) \to C$$

o que eu fiz foi colocar mais uma coisa no lado esquerdo da regra usando uma conjunção.

Será que isso pode ser sempre feito? Isto é, a seguinte regra

$$(p \to q) \to ((r \wedge p) \to q)$$

pode ser construida?

Na verdade, é fácil. É do mesmo jeito, usando transitividade. É só usar H5

$$\xRightarrow{\texttt{H5}} (r \wedge p) \to p$$

depois usamos a regra da transitividade

$$\xRightarrow{\texttt{Trans.}} ((r \wedge p) \to p) \to ((p \to q) \to ((r \wedge p) \to q))$$

e modus ponens

$$\xRightarrow{\texttt{MP}} (p \to q) \to ((r \wedge p) \to q)$$

Ou seja, se eu consigo q a partir de p ($p \to q$), então eu continuo conseguindo q a partir de $r \wedge p$ ($(r \wedge p) \to q$). É simples, gostei dessa regra. Vou chamá-la de Monot.

$$\xRightarrow{\texttt{Monot.}} (p \to q) \to ((r \wedge p) \to q)$$

6. Certo. Acabei de pensar numa interessante

$$(B \to C) \to ((A \wedge B) \to (A \wedge C))$$

O que essa regra me diz é, se eu consigo C a partir de B, então, se eu adiciono A a B, eu continuo obtendo C mais o A ($A \wedge C$).

A regra Monot. *nos dá*

$$\xRightarrow{\texttt{Monot.}} (B \to C) \to ((A \wedge B) \to C)$$

Peraí... H4 *nos dá*

$$\xRightarrow{\texttt{H4}} (A \wedge B) \to A$$

Daria para usar essa regra e o lado esquerdo da regra de cima para, junto com J1*, obter $(A \wedge B) \to (A \wedge C)$. Mas regra de cima tem o $(B \to C)$ na frente.*

Isso tem a cara das transformações 2.0 *da Liduína. Realmente, usando a transformação* H1 *conseguimos*

$$\xRightarrow{\texttt{H1}} (B \to C) \to ((A \wedge B) \to A)$$

Agora, eu acho que preciso de uma `J1 2.0`*. Vou tentar fazer direto, sem fazer a* `J1 2.0`*.*

Tá, primeiro, `J1` *nos dá*

$$\xRightarrow{\texttt{J1}} ((A \wedge B) \to A) \to (((A \wedge B) \to C) \to ((A \wedge B) \to (A \wedge C)))$$

Agora modus ponens com isso e `H4`

$$\xRightarrow{\texttt{MP}} ((A \wedge B) \to C) \to ((A \wedge B) \to (A \wedge C))$$

Agora está fácil, podemos usar `H1` *para colocar algo na frente disso*

$$\xRightarrow{\texttt{H1}} (B \to C) \to (((A \wedge B) \to C) \to ((A \wedge B) \to (A \wedge C)))$$

Ôps! Mas isso não é ainda o que eu queria. Mas está perto. No exercício anterior nós mostramos como contruir

$$\Longrightarrow (B \to C) \to ((A \wedge B) \to C)$$

Eu conseguiria fazer modus ponens com o lado direito dessa e o lado direito da anterior. Mas o $(B \to C)$ está atrapalhando. Vou precisar de um modus ponens `2.0`*.*

Então, a situação é passar de

$$x \to p$$

e

$$x \to (p \to q)$$

para

$$x \to q.$$

Mas isso é fácil usando `H2`*! Pois*

$$\xRightarrow{\texttt{H2}} (x \to (p \to q)) \to ((x \to p) \to (x \to q))$$

Se tivermos

$$\Longrightarrow x \to (p \to q)$$

podemos fazer modus ponnens para chegar em

$$\xRightarrow{\texttt{MP}} (x \to p) \to (x \to q)$$

E com

$$\Longrightarrow x \to p$$

aplicamos `MP` *novamente*

$$\xRightarrow{\texttt{MP}} (x \to q)$$

Voltando ao problema, tínhamos chegado em

$$\xRightarrow{\texttt{H1}} (B \to C) \to (((A \wedge B) \to C) \to ((A \wedge B) \to (A \wedge C)))$$

$$\Longrightarrow (B \to C) \to ((A \wedge B) \to C)$$

agora temos `MP 2.0`*, e obtemos*

$$\xRightarrow{\texttt{MP 2.0}} (B \to C) \to ((A \wedge B) \to (A \wedge C))$$

5. Essa foi uma digressão e tanto. Vou pensar um pouco na disjunção.

As regras da disjução ("OU", $\vee$) não permitem desmontar uma disjunção como é o caso da implicação. Acho que vou começar com a comutatividade, que parece mais simples

$$(A \vee B) \to (B \vee A)$$

Certo, olhando bem, essa é um regra que transforma um "OU" em outro "OU".

O finalzinho da regra `H8` *é uma regra que transforma um "OU" em alguma coisa. Será que essa coisa poderia ser outro "OU". Bem, claro que pode, mas não sei se isso ajuda. Vou tentar.*

$$\xRightarrow{\texttt{H8}} (A \to (B \vee A)) \to ((B \to (B \vee A)) \to ((A \vee B) \to (B \vee A)))$$

Mas agora está fácil, pois `H7` *me dá*

$$\xRightarrow{\texttt{H7}} A \to (B \vee A)$$

e `H6` *me dá*

$$\xRightarrow{\texttt{H6}} B \to (B \vee A)$$

e usando modus ponens duas vezes temos

$$\xRightarrow{\texttt{MP}} (B \to (B \vee A)) \to ((A \vee B) \to (B \vee A))$$

$$\xRightarrow{\texttt{MP}} (A \vee B) \to (B \vee A)$$

Estou notando certa semelhança entre as regras do "E" e do "OU". Isto é, uma semelhança e uma diferença. Elas são parecidas, mas diferentes. Enquanto as regras `H4` *e* `H5` *demontam uma conjunção, as regras* `H6` *e* `H7` *montam disjunções.*

As regras de demontar do "E" eu consegui generalizar para o caso de três letras

$$\Longrightarrow (C \wedge (B \wedge A)) \to C$$

$$\Longrightarrow (C \wedge (B \wedge A)) \to B$$

$$\Longrightarrow (C \wedge (B \wedge A)) \to A$$

Será que eu consigo generalizar as regras de montar do "OU" também?

`H6` *me dá fácil*

$$\xRightarrow{\texttt{H6}} A \to (A \vee (B \vee C))$$

e `H7` *me dá*

$$\xRightarrow{\texttt{H7}} (B \vee C) \to (A \vee (B \vee C))$$

Ôpa! Mas também temos

$$\xRightarrow{\texttt{H6}} B \to (B \vee C)$$

$$\xRightarrow{\texttt{H6}} C \to (B \vee C)$$

Agora sai fácil com transitividade

$$\xRightarrow{\texttt{Trans.}} B \to (A \vee (B \vee C))$$

$$\xRightarrow{\texttt{Trans.}} C \to (A \vee (B \vee C))$$

Engraçado, o "OU" a gente desmonta para trás.

Mas, a regra `H8` *diz como colocar um "OU" na frente das coisas. Ou seja, se eu consegui colocar um A na frente de $(A \vee (B \vee C))$ e um C na frente de $(A \vee (B \vee C))$, eu consigo colocar um $A \vee C$. Vejamos*

$$\xRightarrow{\texttt{H8}} (A \to (A \vee (B \vee C))) \to ((C \to (A \vee (B \vee C))) \to ((A \vee C) \to (A \vee (B \vee C))))$$

Mas a parte da frente eu consegui. Aplicando modus ponens

$$\xrightarrow{\texttt{MP}} (C \to (A \vee (B \vee C))) \to ((A \vee C) \to (A \vee (B \vee C)))$$

e de novo

$$\xrightarrow{\texttt{MP}} (A \vee C) \to (A \vee (B \vee C))$$

Eu posso repetir esses últimos passos para conseguir

$$\Longrightarrow (B \vee (A \vee C)) \to (A \vee (B \vee C))$$

Bem, não é difícil ver que eu posso repetir esses passos com qualquer combinação de letras em uma disjunção. Por exemplo, conseguimos construir regras de montagem para $(C \vee (A \vee B))$*:*

$$\Longrightarrow C \to (C \vee (A \vee B))$$

$$\Longrightarrow A \to (C \vee (A \vee B))$$

$$\Longrightarrow B \to (C \vee (A \vee B))$$

E podemos usar `H8` *duas vezes seguidas para construir "OU's" triplos na parte da frente dessas regras, por exemplo*

$$\Longrightarrow (A \vee (B \vee C)) \to (C \vee (A \vee B))$$

E, também, dá pra repetir todo o processo para fazer isso com qualquer ordem de letras e com os parênteses em qualquer lugar. Em particular, daria para fazer

$$\Longrightarrow (A \vee (B \vee C)) \to (A \vee B) \vee C)$$

Que é a distributividade da disjunção.