INSTITUTO SUPERIOR TÉCNICO

UNIVERSIDADE TÉCNICA DE LISBOA
INSTITUTO SUPERIOR TÉCNICO

Exercícios para as aulas práticas de Representação do Conhecimento

Ana Cardoso-Cachopo

Ano Lectivo 2006/2007

Conteúdo

Prefácio

Estas folhas contêm uma compilação dos exercícios usados nas aulas práticas da disciplina de Representação do Conhecimento no ano lectivo de 2006/2007.

A maior parte deles foi criada por mim especificamente para as aulas práticas ou para as provas de avaliação da disciplina e outros foram tirados de livros ou artigos acerca da matéria em questão.

Existem quatro exercícios cujo enunciado foi feito pelo Professor João Pavão Martins. Esses exercícios estão assinalados com a etiqueta (JPM).

O João Cachopo, para além de discutir comigo alguns dos exercícios, ajudoume a fazer em LaTeX as figuras dos vários capítulos, tornando esta compilação (muito) mais apresentável.

1 Lógica clássica — Argumentos e representação

Sumário:

- Apresentação.
- Devem trazer para as aulas os enunciados dos exercícios para as aulas práticas.
- Argumentos.
- LPO: resolução de exercícios sobre representação.

Resumo:

Na **Representação de Conhecimento** a "passagem" do mundo real para a base de conhecimento é feita por seres humanos. Apenas a inferência é que é feita pelos computadores.

$$\text{Mundo} \xrightarrow[\textit{abstração}]{} \text{Modelo} \xrightarrow[\textit{representação}]{} \text{Base de Conhecimento} \circlearrowright_{\textit{inferência computacional}}$$

Um argumento é representado por um par (Δ, α), em que Δ é um conjunto de proposições (premissas) e α é uma proposição (conclusão).

Uma proposição pode ser *verdadeira* ou *falsa*.

Um argumento é *válido* quando é impossível ter todas as premissas verdadeiras e a conclusão falsa, ou seja, quando todos os modelos das premissas também são modelos da conclusão. É *inválido* caso contrário.

Quando $(\Delta \vdash \alpha)$, o argumento (Δ, α) é *derivável* (em inglês "provable").

Quando $(\Delta \models \alpha)$, o argumento (Δ, α) é *válido*.

Usam-se predicados para representar as proposições, que podem ser *verdadeiras* ou *falsas*.

Usam-se funções para representar objectos.

Existem três tipos de proposições:

- válidas — as que são sempre verdadeiras, independentemente da interpretação usada.
- satisfazíveis — as que podem ser ou não ser verdadeiras, dependendo da interpretação escolhida.
- não satisfazíveis — as que nunca podem ser verdadeiras, independentemente da interpretação escolhida.

Do ponto de vista da RC, as que interessam são as satisfazíveis, pois são as que permitem separar os mundos que as satisfazem dos que não as satisfazem, e assim ajudar a definir o mundo que nos interessa. As válidas são usadas como passos intermédios nas provas, servem para fazer deduções. As não satisfazíveis não interessam nunca, pois nunca podem ser verdadeiras em nenhum mundo.

Uma *conceptualização* é um triplo (D, F, R), em que D é um conjunto de entidades, F é um conjunto de funções e R é um conjunto de relações.

Vantagens da utilização da lógica como formalismo de Representação do Conhecimento:

+ Elevado poder expressivo
+ Solidez
+ Completude
+ Semântica bem definida
+ Existe uma forma (mecânica e) sistemática para associar os objectos existentes na linguagem aos objectos da conceptualização (a interpretação) e de atribuir verdade ou falsidade a fórmulas (noção de satisfação de uma fórmula por uma interpretação)

Desvantagens da utilização da lógica como formalismo de Representação do Conhecimento:

− Não permite quantificar nem falar sobre predicados. A Lógica de segunda ordem já permite, mas não é completa

− É difícil representar que entre três objectos, exactamente dois têm uma dada propriedade. Mas podemos usar abreviaturas, por exemplo, o ${}_n\mathbb{X}_i^j\{A_1, \ldots, A_n\}$ do SNePS

− Falta de estruturação do conhecimento

− Monotonicidade (não permite valores típicos)

− Semi-decidibilidade: se uma fórmula é consequência dum conjunto de fórmulas, então é possível prová-lo. Mas se não for consequência, não há garantia que o algoritmo pare

− Modela apenas o raciocínio dedutivo: apenas torna explícito o que já estava implícito na base de Conhecimento

Outros tipos desejáveis de raciocínio:

- Raciocínio procedimental (Por exemplo, algoritmo da soma)
- Raciocínio por analogia (Por exemplo, tirar conclusões a partir de uma situação parecida)
- Raciocínio abdutivo (Por exemplo, de $A \rightarrow B$ e B inferir A: Se chove, as ruas ficam molhadas. Se vir que as ruas estão molhadas, deduzo que choveu. Mas elas podem estar molhadas por terem sido lavadas)

- Raciocínio indutivo (Por exemplo, a partir de conjuntos de exemplos “inferir” uma regra geral)

De acordo com o livro, a notação usada para a representação de conhecimento usando lógica é:

Funções letra minúscula

Predicados letra maiúscula

Variáveis letra minúscula

Constantes as constantes são funções de zero argumentos, por isso deveriam ser escritas com letra minúscula. No entanto, como eu acho estranho escrever, por exemplo, Bobi com letra minúscula e uma vez que não existe perigo de confundir estas constantes com predicados de zero argumentos, vou usar letras maiúsculas para as constantes, apesar de isso não estar de acordo com as folhas da cadeira.

Exercício 1.1
Usando apenas a informação que está explícita, diga se os seguintes argumentos são válidos ou são inválidos:

1. Peregrino Cinzento é Gandalf
   Mithrandir é Gandalf
   ∴ Peregrino Cinzento é Mithrandir
2. Mithrandir é um feiticeiro
   Mithrandir é Gandalf
   ∴ Gandalf é um feiticeiro
3. Os orcs são feios
   ∴ Os orcs são feios
4. Nemo é um peixe
   Dori é um peixe
   ∴ Nemo é Dori
5. Os tubarões são carnívoros
   Os tubarões não são vegetarianos
   O Bruce é vegetariano
   ∴ O Bruce não é tubarão
6. Os peixes são animais
   ∴ Os tubarões são animais

Resposta:

Pretendemos saber se cada argumento satisfaz a definição de argumento válido: vamos ver se é possível ter a conclusão falsa e as premissas verdadeiras. Se for, o argumento não satisfaz a definição e por isso é inválido; são não for, o argumento satisfaz a definição e por isso é válido.

Podemos resolver este exercício usando diagramas de Venn ou usando a *forma* dos argumentos e partindo do princípio que a noção de igualdade (com as suas propriedades) está definida na lógica que estivermos a usar.

Também podemos fazer as provas "directamente", partindo do princípio que todas as premissas são verdadeiras e testando se a conclusão pode ser falsa; ou por redução ao absurdo, partindo do princípio que a conclusão é falsa e vendo se é possível todas as premissas serem verdadeiras.

Para além destas maneiras de provar se um argumento á válido ou não, também o podemos fazer partindo de um palpite, como está no livro. Se o palpite for que o argumento é válido, tentamos provar que é impossível ter todas as premissas verdadeiras e a conclusão falsa. Se conseguirmos, o argumento é sabido válido. Se não conseguirmos, mudamos o palpite para inválido e tentamos encontar um argumento com a mesma forma mas que nós saibamos que não é válido. Se conseguirmos, o argumento é sabido inválido. Se não conseguirmos provar que o argumento é válido nem inválido, o argumento tem validade desconhecida.

Vamos usar várias alternativas para resolver os exercícios. Normalmente, deve ser usada apenas uma delas.

1. Peregrino Cinzento é Gandalf
   Mithrandir é Gandalf
   ∴ Peregrino Cinzento é Mithrandir
   (a) Diagrama de Venn, prova directa:
       Para Peregrino Cinzento ser Gandalf, ambos têm que ser representados pelo mesmo

ponto. Para Mithrandir ser Gandalf, ambos têm que ser representados pelo mesmo ponto. Logo, é impossível que Peregrino Cinzento não seja Mithrandir. Assim, o argumento é válido.

PC
· G
M

(b) Diagrama de Venn, prova por redução ao absurdo:
Para Peregrino Cinzento não ser Mithrandir, têm que ser representados por pontos diferentes. Para Peregrino Cinzento ser Gandalf, ambos têm que ser representados pelo mesmo ponto. Para Mithrandir ser Gandalf, ambos têm que ser representados pelo mesmo ponto. Como Gandalf não pode ser representado simultaneamente por dois pontos diferentes, é impossível ter simultaneamente a conclusão falsa e todas as premissas verdadeiras, o que significa que o argumento é válido.

PC M
· ·
G G

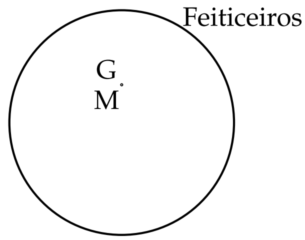
(c) Forma do argumento, prova directa:
Para Peregrino Cinzento ser Gandalf, têm que ser iguais $PC = G$. Para Mithrandir ser Gandalf, têm que ser iguais $G = M$. Logo, $PC = G = M$ e pela transitividade da igualdade, é impossível que Peregrino Cinzento não seja igual a Mithrandir. Assim, o argumento é válido.

(d) Forma do argumento, prova por redução ao absurdo:
Para Peregrino Cinzento não ser Mithrandir, têm que ser diferentes $PC \neq M$. Para Peregrino Cinzento ser Gandalf, têm que ser iguais $PC = G$. Para Mithrandir ser Gandalf, têm que ser iguais $M = G$. Temos $PC \neq M, PC = G, M = G$. Como Gandalf não pode ser diferente de si mesmo, é impossível ter simultaneamente a conclusão falsa e todas as premissas verdadeiras, o que significa que o argumento é válido.

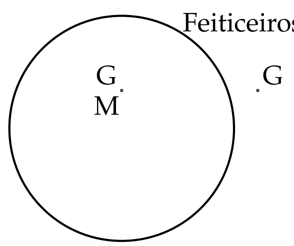
(e) Pelo algoritmo do livro:
Palpite: o argumento é válido.
Resolução: tentar encontrar uma prova. A prova pode ser qualquer uma das apresentadas anteriormente.
Conclusão: como conseguimos encontrar uma prova, o argumento é sabido válido.

2. Mithrandir é um feiticeiro
Mithrandir é Gandalf
$\therefore$ Gandalf é um feiticeiro

(a) Diagrama de Venn, prova directa:
Para Mithrandir ser um feiticeiro, o conjunto dos feiticeiros tem que o conter. Para Mithrandir ser Gandalf, ambos têm que ser representados pelo mesmo ponto. Logo, Gandalf também está obrigatoriamente contido no conjunto dos feiticeiros. Assim, o argumento é válido.



(b) Diagrama de Venn, prova por redução ao absurdo:
Para Gandalf não ser feiticeiro, não pode estar contido no conjunto dos feiticeiros. Para Mithrandir ser um feiticeiro, o conjunto dos feiticeiros tem que o conter. Para Mithrandir ser Gandalf, ambos têm que ser representados pelo mesmo ponto. Como Gandalf não pode ser representado simultaneamente por dois pontos diferentes, é impossível ter simultaneamente a conclusão falsa e todas as premissas verdadeiras, o que significa que o argumento é válido.



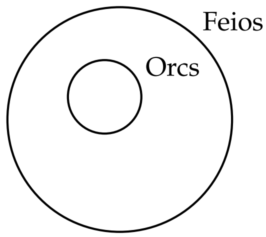
(c) Forma do argumento, prova directa:
Para Mithrandir ser um feiticeiro, $Feiticeiro(M)$. Para Mithrandir ser Gandalf $M = G$. Logo, é possível substituir M por G na primeira fórmula e temos $Feiticeiro(G)$, o que faz com que seja impossível ter $\neg Feiticeiro(M)$. Assim, o argumento é válido.

(d) Forma do argumento, prova por redução ao absurdo:
Para Gandalf não ser feiticeiro, temos $\neg Feiticeiro(G)$. Para Mithrandir ser um feiticeiro, temos $Feiticeiro(M)$. Para Mithrandir ser Gandalf $M = G$ e por isso, $Feiticeiro(G)$. Como isto é contraditório com primeira fórmula, é impossível ter simultaneamente a conclusão falsa e todas as premissas verdadeiras, o que significa que o argumento é válido.

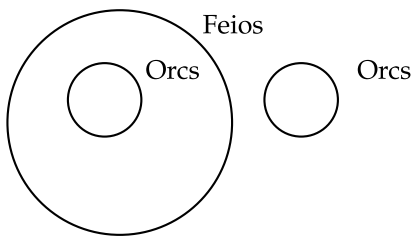
(e) Pelo algoritmo do livro:
Palpite: o argumento é válido.
Resolução: tentar encontrar uma prova. A prova pode ser qualquer uma das apresentadas anteriormente.
Conclusão: como conseguimos encontrar uma prova, o argumento é sabido válido.

3. Os orcs são feios
$\therefore$ Os orcs são feios

(a) Diagrama de Venn, prova directa:
Para os orcs, serem feios, o conjunto que os representa tem que estar contido no conjunto que representa os feios. Para a conclusão ser falsa, é necessário que os orcs não sejam feios, ié, que não estejam contidos no conjunto dos feios. Como isto é impossível dada a primeira frase, o argumento é válido.



(b) Diagrama de Venn, prova por redução ao absurdo:
Vamos supôr que os orcs não são feios, isto é, que o conjunto que os representa não está contido no conjunto que representa os feios. Posto isto, é possível ter a premissa verdadeira, isto é, que os orcs sejam feios, ou seja, que estejam contidos no conjunto dos feios? Como é impossível que o conjunto dos orcs estaja contido e não esteja contido no conjunto dos feios, o argumento é válido.



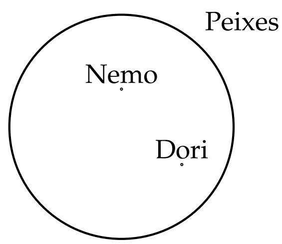
(c) Forma do argumento, prova directa:
Para os orcs serem feios, temos $\forall(x)[Orc(x) \to Feio(x)]$. Logo, é impossível termos a conclusão falsa, ou seja $\neg\forall(x)[Orc(x) \to Feio(x)]$. Assim, o argumento é válido.

(d) Forma do argumento, prova por redução ao absurdo:
Vamos supôr que a conclusão é falsa, isto é, que $\neg\forall(x)[Orc(x) \to Feio(x)]$. Neste caso, é impossível que a premissa $\forall(x)[Orc(x) \to Feio(x)]$ seja verdadeira. Logo, como é impossível ter simultaneamente todas as premissas verdadeiras e a conclusão falsa, o argumento é válido.

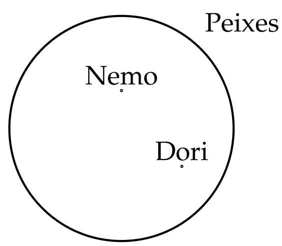
(e) Pelo algoritmo do livro:
Palpite: o argumento é válido.
Resolução: tentar encontrar uma prova. A prova pode ser qualquer uma das apresentadas anteriormente.
Conclusão: como conseguimos encontrar uma prova, o argumento é sabido válido.

4. Nemo é um peixe
Dori é um peixe
$\therefore$ Nemo é Dori

(a) Diagrama de Venn, prova directa:
Para Nemo ser um peixe, tem que estar contido no conjunto dos peixes. Para Dori ser um peixe, tem que estar contida no conjunto dos peixes. Pode ser igual ou diferente do Nemo. Logo, é possível ter todas as premissas verdadeiras e a conclusão falsa, isto é, Nemo não ser Dori. Assim, o argumento não é válido.



(b) Diagrama de Venn, prova por redução ao absurdo:
Vamos supôr que a conclusão é falsa, isto é, que Nemo não é Dori. É possível que ambos sejam peixes? Sim, não há nada que impeça o conjunto dos peixes de conter os dois. Logo, como é possível ter simultaneamente todas as premissas verdadeiras e a conclusão falsa, o argumento não é válido.



(c) Forma do argumento, prova directa:
Para Nemo ser peixe, temos $Peixe(Nemo)$. Para Dori ser peixe, temos $Peixe(Dori)$. Nada nos impede de ter $Nemo \neq Dori$. Logo, como é possível ter simultaneamente todas as premissas verdadeiras e a conclusão falsa, o argumento não é válido.

(d) Forma do argumento, prova por redução ao absurdo:
Vamos supôr que a conclusão é falsa, isto é, que $Nemo \neq Dori$. É possível que ambos sejam peixes? Sim, não há nada que impeça de ter simultaneamente $Peixe(Nemo)$ e $Peixe(Dori)$. Logo, como é possível ter simultaneamente todas as premissas verdadeiras e a conclusão falsa, o argumento não é válido.

(e) Pelo algoritmo do livro:
Palpite: o argumento não é válido.
Resolução: tentar encontrar um contra-argumento, isto é, um argumento com a mesma forma mas que nós saibamos que é inválido.

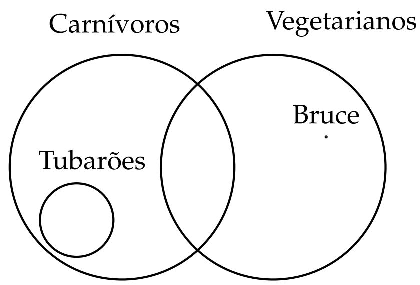
2 é um número
5 é um número
∴ 2 é 5

Nós sabemos que 2 não é 5, logo, este argumento tem todas as premissas verdadeiras e a conclusão falsa, por isso não é válido.
Conclusão: como conseguimos encontrar um contra-argumento, o argumento é sabido inválido.

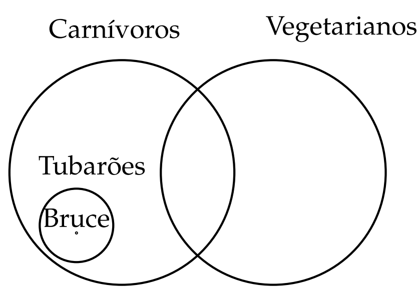
5. Os tubarões são carnívoros
Os tubarões não são vegetarianos
O Bruce é vegetariano
∴ O Bruce não é tubarão

(a) Diagrama de Venn, prova directa:
Para os tubarões serem carnívoros, o conjunto que representa os tubarões tem que estar contido no conjunto que representa os carnívoros. Para os tubarões não serem vegetarianos, o conjunto que representa os tubarões não pode estar contido no conjunto que representa os vegetarianos, embora os carnívoros e os vegetarianos se possam intersectar, sem problemas. Para o Bruce ser vegetariano, tem que estar contido no conjunto que representa os vegetarianos. Com isto, é impossível que o Bruce esteja

contido no conjunto que representa os tubarões. Logo, o argumento e válido.



(b) Diagrama de Venn, prova por redução ao absurdo:
Vamos supôr que o Bruce é um tubarão, o que significa que tem que estar contido no conjunto que representa os tubarões. Para os tubarões serem carnívoros, o conjunto que representa os tubarões tem que estar contido no conjunto que representa os carnívoros. Para os tubarões não serem vegetarianos, o conjunto que representa os tubarões não pode estar contido no conjunto que representa os vegetarianos. Para o Bruce ser vegetariano, tem que estar contido no conjunto que representa os vegetarianos. Isto é impossível, uma vez que ele é um tubarão e os tubarões não são vegetarianos. Logo, como é impossível ter simultaneamente todas as premissas verdadeiras e a conclusão falsa, o argumento é válido.

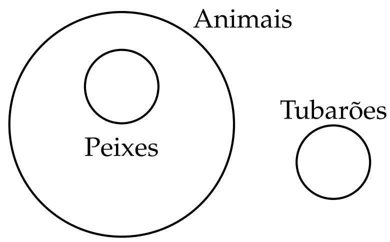


(c) Forma do argumento, prova directa:
Para os tubarões serem carnívoros, temos $\forall(x)[Tubarao(x) \rightarrow Carnivoro(x)]$. Para os tubarões não serem vegetarianos, temos $\forall(x)[Tubarao(x) \rightarrow \neg Vegetariano(x)]$. Para o Bruce ser vegetariano, temos $Vegetariano(Bruce)$. Dadas estas premissas, é obrigatório que a conclusão seja verdadeira, pois se ela fosse falsa e tivéssemos $Tubarao(Bruce)$ ele não seria vegetariano e teríamos um a contradição. Logo, o argumento e válido.

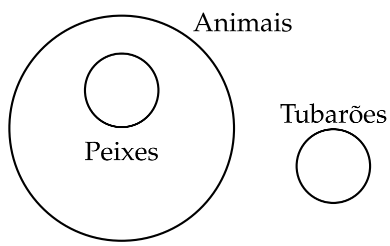
(d) Forma do argumento, prova por redução ao absurdo:
Vamos supôr que a conclusão é falsa, ou seja, que o Bruce é um tubarão $Tubarao(Bruce)$. Para os tubarões serem carnívoros, temos $\forall(x)[Tubarao(x) \rightarrow Carnivoro(x)]$. Para os tubarões não serem vegetarianos, temos $\forall(x)[Tubarao(x) \rightarrow \neg Vegetariano(x)]$. Para o Bruce ser vegetariano, temos $Vegetariano(Bruce)$. Isto é impossível, uma vez que ele é um tubarão e os tubarões não são vegetarianos e por isso também temos $\neg Vegetariano(Bruce)$ que dá uma contradição. Logo, como é impossível ter simultaneamente todas as premissas verdadeiras e a conclusão falsa, o argumento é válido.

(e) Pelo algoritmo do livro:
Palpite: o argumento é válido.
Resolução: tentar encontrar uma prova. A prova pode ser qualquer uma das apresentadas anteriormente.
Conclusão: como conseguimos encontrar uma prova, o argumento é sabido válido.

6. Os peixes são animais
   $\therefore$ Os tubarões são animais
   (a) Diagrama de Venn, prova directa:
   Para os peixes serem animais, têm que estar contidos no conjunto dos animais. Com base nesta premissa, nada nos impede de ter o conjunto dos tubarões fora do conjunto dos animais. Logo, é possível ter todas as premissas verdadeiras e a conclusão falsa, isto é, os tubarões não serem animais. Assim, o argumento não é válido.



   (b) Diagrama de Venn, prova por redução ao absurdo:
   Vamos supôr que a conclusão é falsa, isto é, que os tubarões não são animais. É possível que os peixes sejam animais? Sim, não há nada que impeça o conjunto dos peixes de estar contido no conjunto dos animais. Logo, como é possível ter simultaneamente todas as premissas verdadeiras e a conclusão falsa, o argumento não é válido.



   (c) Forma do argumento, prova directa:
   Para os peixes serem animais, temos $\forall(x)[Peixe(x) \rightarrow Animal(x)]$. Nada nos impede de ter $\neg\forall(x)[Tubarao(x) \rightarrow Animal(x)]$. Logo, como é possível ter simultaneamente todas as premissas verdadeiras e a conclusão falsa, o argumento não é válido.
   (d) Forma do argumento, prova por redução ao absurdo:
   Vamos supôr que a conclusão é falsa, isto é, que $\neg\forall(x)[Tubarao(x) \rightarrow Animal(x)]$. É possível que os peixes sejam animais, isto é, $\forall(x)[Peixe(x) \rightarrow Animal(x)]$? Sim, não há nada que nos impeça de ter simultaneamente estas duas fórmulas. Logo, como é possível ter simultaneamente todas as premissas verdadeiras e a conclusão falsa, o argumento não é válido.
   (e) Pelo algoritmo do livro:
   Palpite: o argumento não é válido.
   Resolução: tentar encontrar um contra-argumento, isto é, um argumento com a mesma forma mas que nós saibamos que é inválido.
   Os peixes são animais
   $\therefore$ Os números são animais
   Nós sabemos que os números não são animais, logo, este argumento tem todas as premissas verdadeiras e a conclusão falsa, por isso não é válido.
   Conclusão: como conseguimos encontrar um contra-argumento, o argumento é sabido inválido.

Exercício 1.2
Considere a seguinte frase: "o Bit é um cão preto". Discuta várias representações possíveis para ela, em lógica de primeira ordem.

Resposta:

Em termos puramente lógicos, a solução mais simples (embora claramente não seja a mais natural) é usar um símbolo de predicado de zero argumentos, por exemplo

$$P()$$

e na interpretação dizer que este predicado corresponde à proposição de que o Bit é um cão preto.

Mas, se tivermos em conta que o conhecimento que representamos deveria ser compreensível por outras pessoas que o quisessem usar ou modificar, deveríamos ter o cuidado de usar nomes mais facilmente compreensíveis (já todos compreendemos bem a necessidade de, num programa, escolher nomes adequados para as variáveis).

Uma alternativa um pouco melhor seria representar esta frase usando o predicado de zero argumentos

$$OBitEUmCaoPreto()$$

dizendo qual o significado deste predicado na nossa interpretação.

O problema com esta representação é que, apesar de se compreender com mais facilidade, não tem partes que sejam reutilizáves.

Por exemplo, não temos nada na nossa representação que nos permita falar do Bit. Assim, faria mais sentido introduzirmos uma constante para representar o Bit. Ficaríamos com:

$$EUmCaoPreto(Bit)$$

Se quisermos dizer, por exemplo que o Pluto é um cão castanho, podemos fazer algo semelhante:

$$EUmCaoCastanho(Pluto)$$

Mas não há nada que relacione estes dois predicados entre si nem com outros que também refiram cães.

Por isso, se quisermos representar informação acerca de outros cães ou de outros objectos pretos ou castanhos, vamos ter que usar outros predicados que não têm nada a ver com estes. Isto torna-se particularmente problemático quando quisermos representar informação acerca dos cães em geral, por exemplo, que são mamíferos.

Tendo em conta a necessidade de termos uma representação facilmente compreensível por outros e suficientemente modular para podermos reutilizar algumas das suas partes, uma boa representação seria:

$$Cao(Bit) \wedge Preto(Bit)$$

Ou se quisermos ter o conceito de cor,

$$Cao(Bit) \wedge Cor(Bit, Preto)$$

Neste caso, podemos ter

$$Cao(Pluto) \wedge Castanho(Pluto)$$

ou

$$Cao(Pluto) \wedge Cor(Pluto, Castanho)$$

e já estamos a usar os mesmos predicados para representar os dois cães.

Se quisermos representar conhecimento acerca da cauda do Bit, devemos usar uma função, pois a cauda é única. Podemos ter, por exemplo:

$$Comprida(cauda(Bit))$$

$$Cor(cauda(Bit), Preto)$$

$$MaisComprida(cauda(Pluto), cauda(Bit))$$

Se quisermos falar acerca das suas orelhas, precisamos de duas funções, uma para a orelha esquerda e outra para a orelha direita. Para as suas quatro patas, já poderá fazer sentido usar funções ou predicados.

Em RC vamos treinar este tipo de escolhas, que se podem aplicar também noutros domínios, como por exemplo, decidir quais as entidades que devem ter uma representação explícita num determinado programa.

Exercício 1.3

Represente em lógica de primeira ordem a afirmação:
"O Nuno ou é polícia ou é ladrão, mas não os dois simultaneamente."

Resposta:

$(Policia(Nuno) \lor Ladrao(Nuno)) \land \neg(Policia(Nuno) \land Ladrao(Nuno))$ ou
$(Policia(Nuno) \land \neg Ladrao(Nuno)) \lor (\neg Policia(Nuno) \land Ladrao(Nuno))$
Se representar usando implicações, atenção à contrapositiva.
Ver a discussão da solução do exercício 2.3.1 dos exercícios resolvidos, acerca do BolaDeNeve, que é cão ou gato, mas não as duas coisas simultaneamente.

Exercício 1.4

Represente em lógica de primeira ordem as afirmações:

1. Nenhum tubarão é pessoa.
2. Nem todos os tubarões são carnívoros.
3. Todos os tubarões têm uma cauda.
4. Qualquer tubarão que esteja vivo pode nadar e morder.
5. Só os homens e as mulheres é que são pessoas.
6. Se alguém consegue esquiar então o Nuno também consegue.
7. Tudo o que alguém consegue fazer o Nuno também consegue.
8. O Pai da Maria é casado com a Mãe da Maria.
9. O João acredita que sabe a idade da Maria.

Resposta:

Nestes exercícios, convém não esquecer de colocar a interpretação que está a ser usada, indicando explicitamente o que é que são predicados, funções e constantes e quais os seus significados.

1. Nenhum tubarão é pessoa.
   $\forall(x)[Tubarao(x) \rightarrow \neg Pessoa(x)]$
   ou
   $\neg(\exists(x)[Tubarao(x) \wedge Pessoa(x)])$
   Estas duas representações são equivalentes:
   $\neg(\exists(x)[Tubarao(x) \wedge Pessoa(x)]) \Leftrightarrow$
   $\forall(x)[\neg(Tubarao(x) \wedge Pessoa(x))] \Leftrightarrow$
   $\forall(x)[\neg Tubarao(x) \vee \neg Pessoa(x)] \Leftrightarrow$
   $\forall(x)[Tubarao(x) \rightarrow \neg Pessoa(x)]$
2. Nem todos os tubarões são carnívoros.
   $\exists(x)[Tubarao(x) \wedge \neg Carnivoro(x)] \wedge \exists(x)[Tubarao(x) \wedge Carnivoro(x)]$ (Segunda parte opcional).
   Se quisermos, podemos fazer a seguinte demonstração:
   $\neg(\forall(x)[Tubarao(x) \rightarrow Carnivoro(x)]) \Leftrightarrow$
   $\exists(x)[\neg(Tubarao(x) \rightarrow Carnivoro(x))] \Leftrightarrow$
   $\exists(x)[\neg(\neg Tubarao(x) \vee Carnivoro(x))] \Leftrightarrow$
   $\exists(x)[Tubarao(x) \wedge \neg Carnivoro(x)]$
3. Todos os tubarões têm uma cauda.
   $\forall(x)[Tubarao(x) \rightarrow \exists(y)[Cauda(y) \wedge Tem(x, y)]]$
   O problema com esta representação é que assim os tubarões podem ter mais do que uma cauda. Assim, e uma vez que sabemos que cada tubarão tem uma e uma só cauda, devemos usar uma função para o representar:
   $\forall(x)[Tubarao(x) \rightarrow Tem(x, cauda(x))]$
4. Qualquer tubarão que esteja vivo pode nadar e morder.
   $\forall(x)[(Tubarao(x) \wedge Vivo(x)) \rightarrow (Pode(x, Nadar) \wedge Pode(x, Morder))]$
5. Só os homens e as mulheres é que são pessoas.
   $\forall(x)[Pessoa(x) \rightarrow (Homem(x) \wedge Mulher(x))]$
   Esta representação está errada! Apesar de a frase em português corrente ter a palavra "e", a conectiva lógica a usar neste caso é a correspondente ao "ou":
   $\forall(x)[Pessoa(x) \rightarrow (Homem(x) \vee Mulher(x))]$
   ou
   $\neg\exists(x)[Pessoa(x) \wedge \neg Homem(x) \wedge \neg Mulher(x)]$
   Mas esta frase apenas diz que não há pessoas que não sejam homens nem mulheres, não diz, por exemplo, que os homens são pessoas. Para ficar completa, precisa das fórmulas seguintes:
   $\forall(x)[Homem(x) \rightarrow Pessoa(x)]$
   $\forall(x)[Mulher(x) \rightarrow Pessoa(x)]$
6. Se alguém consegue esquiar então o Nuno também consegue.
   $(\exists(x)[Pessoa(x) \wedge Consegue(x, Esquiar)]) \rightarrow Consegue(Nuno, Esquiar)$
   Atenção ao âmbito do quantificador $\exists$!!! Não pode ser até ao fim da fórmula. Se fosse até ao fim, teríamos: $(\exists(x)[(Pessoa(x) \wedge Consegue(x, Esquiar)) \rightarrow Consegue(Nuno, Esquiar)])$. Com esta fórmula, desde que exista uma pessoa (por exemplo o Manuel) que não consegue esquiar, esta fórmula é verdadeira, mesmo que as proposições $Consegue(Rui, Esquiar)$ e $\neg Consegue(Nuno, Esquiar)$ sejam verdade. Isto não está de acordo com a frase do enunciado.
   Poderemos ter outra solução correcta, transformando a implicação numa disjunção, alargando o âmbito do quantificador universal e voltando a transformar a disjunção numa implicação. Ficamos com:
   $\forall(x)[\neg Pessoa(x) \vee \neg Consegue(x, Esquiar)] \vee Consegue(Nuno, Esquiar) \Leftrightarrow$
   $\forall(x)[\neg Pessoa(x) \vee \neg Consegue(x, Esquiar) \vee Consegue(Nuno, Esquiar)] \Leftrightarrow$
   $\forall(x)[(Pessoa(x) \wedge Consegue(x, Esquiar)) \rightarrow Consegue(Nuno, Esquiar)]$

7. Tudo o que alguém consegue fazer o Nuno também consegue.
   $\forall(y)[\exists(x)[Pessoa(x) \wedge ConsegueFazer(x,y)] \rightarrow ConsegueFazer(Nuno,y)] \Leftrightarrow$
   $\forall(y)[\forall(x)[\neg Pessoa(x) \vee \neg ConsegueFazer(x,y)] \vee ConsegueFazer(Nuno,y)] \Leftrightarrow$
   $\forall(x,y)[\neg Pessoa(x) \vee \neg ConsegueFazer(x,y) \vee ConsegueFazer(Nuno,y)] \Leftrightarrow$
   $\forall(x,y)[(Pessoa(x) \wedge ConsegueFazer(x,y)) \rightarrow ConsegueFazer(Nuno,y)]$

8. O Pai da Maria é casado com a Mãe da Maria.
   $Casados(pai(Maria), mae(Maria))$

   Neste caso, *pai* e *mae* são funções e os termos $pai(Maria)$ e $mae(Maria)$ representam o Pai e a Mãe da Maria, respectivamente. Desta forma, garantimos que a Maria só tem um Pai e uma Mãe.

   Outra proposta de solução poderia ser:

   $\exists(x,y)[Pai(x, Maria) \wedge Mae(y, Maria) \wedge Casados(x,y)]$ No entanto, com esta representação não garantimos que a Maria só tem um Pai e uma Mãe, porque nada impede que exista mais do que um par de constantes que torne esta fórmula verdadeira.

   Se quisermos ver o que acontece com um quantificador universal:

   $\forall(x,y)[(Pai(x, Maria) \wedge Mae(y, Maria)) \rightarrow Casados(x,y)]$

   vemos que assim também não conseguimos representar correctamente a informação: desta forma, a Maria pode não ter nenhum Pai nem nenhuma Mãe. Mudámos a conectiva principal de conjunção para implicação porque se não o tivéssemos feito estariamos a obrigar todas as constantes do domínio a serem Pai e Mãe da Maria, inclusivamente ela própria.

9. O João acredita que sabe a idade da Maria.
   Como a Maria tem uma e uma só idade, podemos usar uma função. Uma primeira tentativa seria $Acredita(Joao, Sabe(Joao, idade(Maria)))$. Mas isto seria usar uma lógica de segunda ordem, em que os predicados podem "falar" acerca de proposições. Com estes predicados, esta frase não se pode representar numa Lógica de Primeira Ordem. Mas, se considerarmos que existe o predicado $AcreditaQueSabe$, podemos escrever $AcreditaQueSabe(Joao, idade(Maria))$.

Exercício 1.5

Considere que $PP(x)$ e $P(x)$ representam, respectivamente, "x é um peixe-palhaço" e "x é um peixe" . Represente em lógica de primeira ordem as seguintes frases:

1. Todos os peixes-palhaço são peixes.

2. Alguns peixes são peixes-palhaço.

3. Nem todos os peixes são peixes-palhaço.

Resposta:

1. Todos os peixes-palhaço são peixes.
   $\forall(x)[PP(x) \rightarrow P(x)]$

2. Alguns peixes são peixes-palhaço.
   $\exists(x)[P(x) \wedge PP(x)]$

3. Nem todos os peixes são peixes-palhaço.
   $\exists(x)[P(x) \wedge \neg PP(x)]$ ou
   $\neg\forall(x)[P(x) \rightarrow PP(x)]$

Exercício 1.6
Suponha que $F(x)$ representa o predicado "x é um feiticeiro" e que $H(x)$ representa o predicado "x é humano". Traduza as seguintes fbfs para linguagem comum. Se não conseguir traduzir alguma delas, explique porquê.

1. $\exists(x)[H(x)]$
2. $\neg\exists(x)[F(x)]$
3. $\forall(x)[\neg F(x)]$
4. $\forall(x)[F(x) \rightarrow \neg H(x)]$
5. $\exists(x)[F(x) \wedge H(x)]$
6. $\exists(x)[F(x) \rightarrow H(x)]$

Resposta:

1. $\exists(x)[H(x)]$
   Existe pelo menos um humano.
2. $\neg\exists(x)[F(x)]$
   Não existe nenhum feiticeiro.
3. $\forall(x)[\neg F(x)]$
   Nenhum x é feiticeiro, ou melhor, não existe nenhum feiticeiro. Equivalente à anterior.
4. $\forall(x)[F(x) \rightarrow \neg H(x)]$
   Nenhum feiticeiro é humano.
5. $\exists(x)[F(x) \wedge H(x)]$
   Existe pelo menos um feiticeiro que é humano.
6. $\exists(x)[F(x) \rightarrow H(x)]$
   Esta proposição não tem utilidade, pois basta existir um elemento do domínio que não satisfaça o predicado $F(x)$ para ela ser verdadeira. Está aqui porque é um erro que aparece muito frequentemente nos testes.

2 Lógica clássica — Sistemas sintáctico e semântico

Sumário:

- Sistema dedutivo: regras de inferência e provas.
- Sistema semântico: método das tabelas.

Resumo:

Em RC temos o mundo, através da abstração ficamos com um modelo e através da representação ficamos com a base de conhecimento.

$$Mundo \rightarrow Modelo \rightarrow BaseDeConhecimento$$

Depois de termos a base de conhecimento, fazemos inferência com base no que lá está representado. O mais difícil é chegar à base de conhecimento (por enquanto só os humanos é que o conseguem fazer), a inferência até os computadores são capazes de fazer...

Tratámos da criação da base de conhecimento (representação) na aula anterior, vamos tratar da inferência nesta aula.

Os argumentos continuam a ser representados por um par (Δ, α). Quando uma fórmula α é derivável (usando o sistema sintáctico) a partir de um conjunto de fórmulas Δ, escrevemos $\Delta \vdash \alpha$. Quando uma fórmula α é consequência lógica (usando o sistema semântico) de um conjunto de fórmulas Δ, escrevemos $\Delta \models \alpha$.

A lógica de primeira ordem é sólida e completa. Isto é importante para nos permitir fazer provas pela via sintáctica, com base no conhecimento que temos e sabermos que isso tem uma "correspondência semântica" no mundo que estamos a representar. Enquanto que na lógica proposicional é relativamente simples encontrar algoritmos eficientes para fazer provas pela via semântica, na lógica de primeira ordem isso é mais complicado. Assim, fazemos as provas pela via sintáctica e, pela solidez e completude da lógica, sabemos que seríamos capazes de provar o mesmo pela via semântica.

As regras que se podem usar nas provas pelo sistema sintáctico são as que estão nas tabelas de resumo do livro (e na página seguinte). Também se podem usar as regras derivadas que estão no livro, embora não sejam essenciais. Se se usar uma regra derivada que não esteja no livro, é necessário provar que ela funciona.

Para simplificar as provas, permitimos que se usem algumas das regras com mais graus de liberdade:

$\rightarrow$**E** podemos ter a implicação antes do antecedente, desde que se troquem os índices ao indicar a aplicação da regra, isto é, primeiro aparece o antecedente e depois a implicação.

$\vee$**E** podemos ter a disjunção depois das provas, desde que se troquem os índices ao indicar a aplicação da regra, isto é, primeiro aparece o índice da disjunção, depois os da primeira prova e depois os da segunda.

$\wedge$**E** quando existe uma conjunção com mais do que dois elementos, podemos eliminar a conjunção e ficar com cada um deles individualmente sem ter que passar por conjunções com 2 elementos.

Reit podemos passar para dentro de mais do que um nível de cada vez. Em todas as outras regras temos que respeitar os níveis de prova.

Seguem-se as regras de inferência do sistema dedutivo da lógica clássica.

Lógica de primeira ordem — Sistema dedutivo

Prem

$$\begin{array}{lll} n & \alpha & \text{Prem} \end{array}$$

Hyp

$$\begin{array}{l|ll} n & \alpha & \text{Hyp} \\ \cline{2-2} n+1 & \ldots & \end{array}$$

Rep

$$\begin{array}{lll} n & \alpha & \\ \vdots & \vdots & \\ m & \alpha & \text{Rep}, n \end{array}$$

Reit

$$\begin{array}{l|ll} n & \multicolumn{1}{l}{\alpha} & \\ \vdots & \vdots & \\ m & \alpha & \text{Reit}, n \end{array}$$

$\rightarrow$ I

$$\begin{array}{l|ll} n & \alpha & \text{Hyp} \\ \cline{2-2} \vdots & \vdots & \\ m & \beta & \\ m+1 & \multicolumn{1}{l}{\alpha \rightarrow \beta} & \rightarrow\text{I}, n, m \end{array}$$

$\rightarrow$ E

$$\begin{array}{lll} n & \alpha & \\ m & \alpha \rightarrow \beta & \\ m+1 & \beta & \rightarrow\text{E}, n, m \end{array}$$

$\wedge$I

$$\begin{array}{lll} n & \alpha & \\ n+1 & \beta & \\ n+2 & \alpha \wedge \beta & \wedge\text{I}, n, n+1 \end{array}$$

$\wedge$E

$$\begin{array}{lll} n & \alpha \wedge \beta & \\ n+1 & \alpha & \wedge\text{E}, n \end{array}$$

e

$$\begin{array}{lll} n & \alpha \wedge \beta & \\ n+1 & \beta & \wedge\text{E}, n \end{array}$$

$\vee$I

$$\begin{array}{lll} n & \alpha & \\ n+1 & \alpha \vee \beta & \vee\text{I}, n \end{array}$$

e

$$\begin{array}{lll} n & \alpha & \\ n+1 & \beta \vee \alpha & \vee\text{I}, n \end{array}$$

$\vee$E

$$\begin{array}{ll|ll} n & \multicolumn{2}{l}{\alpha \vee \beta} & \\ o & & \alpha & \text{Hyp} \\ \cline{3-3} \vdots & & \vdots & \\ p & & \gamma & \\ r & & \beta & \text{Hyp} \\ \cline{3-3} \vdots & & \vdots & \\ s & & \gamma & \\ m & \multicolumn{2}{l}{\gamma} & \vee\text{E}, n, o\text{–}p, r\text{–}s \end{array}$$

$\neg$I

$$\begin{array}{l|ll} n & \alpha & \text{Hyp} \\ \cline{2-2} \vdots & \vdots & \\ m & \beta & \\ m+1 & \neg\beta & \\ m+2 & \multicolumn{1}{l}{\neg\alpha} & \neg\text{I}, n, m, m+1 \end{array}$$

$\neg$E

$$\begin{array}{lll} n & \neg\neg\alpha & \\ n+1 & \alpha & \neg\text{E}, n \end{array}$$

$\forall$I

$$\begin{array}{ll|ll} n & x_0 & & \\ \vdots & & \vdots & \\ m & & \alpha(x_0) & \\ m+1 & \multicolumn{2}{l}{\forall x[\alpha(x)]} & \forall\text{I}, n, m \end{array}$$

$\forall$E

$$\begin{array}{lll} n & \forall x[\alpha(x)] & \\ n+1 & \alpha(t) & \forall\text{E}, n \end{array}$$

$\exists$I

$$\begin{array}{lll} n & \alpha(t) & \\ n+1 & \exists x[\alpha(x)] & \exists\text{I}, n \end{array}$$

$\exists$E

$$\begin{array}{ll|ll} n & \multicolumn{2}{l}{\exists x[\alpha(x)]} & \\ m & x_0 & \alpha(x_0) & \text{Hyp} \\ \cline{3-3} \vdots & & \vdots & \\ k & & \beta & \\ k+1 & \multicolumn{2}{l}{\beta} & \exists\text{E}, n, m, k \end{array}$$

Exercício 2.1

(JPM) Demonstre os seguintes teoremas e argumentos usando o sistema de dedução natural da lógica proposicional. Em cada alínea indique se está a demonstrar um teorema ou um argumento.

1. $A \rightarrow (B \rightarrow A)$
2. $(A \wedge \neg A) \rightarrow B$
3. $(\{A \rightarrow B, B \rightarrow \neg A\}, \neg A)$
4. $(\{A\}, B \rightarrow (A \wedge B))$
5. $((A \rightarrow B) \wedge (B \rightarrow C)) \rightarrow (A \rightarrow C)$
6. $(\{\neg(A \vee B)\}, \neg A \wedge \neg B)$
7. $(A \vee \neg B) \rightarrow \neg(\neg A \wedge B)$
8. $(\{\}, \neg(\neg A \vee \neg B) \rightarrow (A \wedge B))$

Resposta:

1. $A \rightarrow (B \rightarrow A)$ — teorema

1	$\mid A$	Hyp
2	$\mid \quad \mid B$	Hyp
3	$\mid \quad \mid A$	Reit, 1
4	$\mid B \rightarrow A$	$\rightarrow$I, 2, 3
5	$A \rightarrow (B \rightarrow A)$	$\rightarrow$I, 1, 4

2. $(A \wedge \neg A) \rightarrow B$ — teorema

1	$\mid A \wedge \neg A$	Hyp
2	$\mid \quad \mid \neg B$	Hyp
3	$\mid \quad \mid A \wedge \neg A$	Reit, 1
4	$\mid \quad \mid A$	$\wedge$E, 3
5	$\mid \quad \mid \neg A$	$\wedge$E, 3
6	$\mid \neg\neg B$	$\neg$I, 2, 4, 5
7	$\mid B$	$\neg$E, 6
8	$(A \wedge \neg A) \rightarrow B$	$\rightarrow$I, 1, 7

3. $(\{A \to B, B \to \neg A\}, \neg A)$ — argumento

1	$A \to B$	Prem
2	$B \to \neg A$	Prem
3	$\quad \vert\ \underline{A}$	Hyp
4	$\quad \vert\ A \to B$	Reit, 1
5	$\quad \vert\ B$	$\to$E, 3, 4
6	$\quad \vert\ B \to \neg A$	Reit, 2
7	$\quad \vert\ \neg A$	$\to$E, 5, 6
8	$\neg A$	$\neg$I, 3, 3, 7

4. $(\{A\}, B \to (A \land B))$ — argumento

1	A	Prem
2	$\quad \vert\ \underline{B}$	Hyp
3	$\quad \vert\ A$	Reit, 1
4	$\quad \vert\ A \land B$	$\land$I, 3, 2
5	$B \to (A \land B)$	$\to$I, 2, 4

5. $((A \to B) \land (B \to C)) \to (A \to C)$ — teorema

1	$\vert\ \underline{(A \to B) \land (B \to C)}$	Hyp
2	$\vert\ A \to B$	$\land$E, 1
3	$\vert\ B \to C$	$\land$E, 1
4	$\vert\ \quad \vert\ \underline{A}$	Hyp
5	$\vert\ \quad \vert\ A \to B$	Reit, 2
6	$\vert\ \quad \vert\ B$	$\to$E, 4, 5
7	$\vert\ \quad \vert\ B \to C$	Reit, 3
8	$\vert\ \quad \vert\ C$	$\to$E, 6, 7
9	$\vert\ A \to C$	$\to$I, 4, 8
10	$((A \to B) \land (B \to C)) \to (A \to C)$	$\to$I, 1, 9

6. $(\{\neg(A \vee B)\}, \neg A \wedge \neg B)$ — argumento

1	$\neg(A \vee B)$	Prem
2	$\mid$ A	Hyp
3	$\mid$ $A \vee B$	$\vee$I, 2
4	$\mid$ $\neg(A \vee B)$	Reit, 1
5	$\neg A$	$\neg$I, 2, 3, 4
6	$\mid$ B	Hyp
7	$\mid$ $A \vee B$	$\vee$I, 6
8	$\mid$ $\neg(A \vee B)$	Reit, 1
9	$\neg B$	$\neg$I, 6, 7, 8
10	$\neg A \wedge \neg B$	$\wedge$I, 5, 9

7. $(A \vee \neg B) \rightarrow \neg(\neg A \wedge B)$ — teorema

1	$\mid$ $A \vee \neg B$	Hyp
2	$\mid$ $\mid$ A	Hyp
3	$\mid$ $\mid$ $\mid$ $\neg A \wedge B$	Hyp
4	$\mid$ $\mid$ $\mid$ $\neg A$	$\wedge$E, 3
5	$\mid$ $\mid$ $\mid$ A	Reit, 2
6	$\mid$ $\mid$ $\neg(\neg A \wedge B)$	$\neg$I, 3, 4, 5
7	$\mid$ $\mid$ $\neg B$	Hyp
8	$\mid$ $\mid$ $\mid$ $\neg A \wedge B$	Hyp
9	$\mid$ $\mid$ $\mid$ B	$\wedge$E, 8
10	$\mid$ $\mid$ $\mid$ $\neg B$	Reit, 7
11	$\mid$ $\mid$ $\neg(\neg A \wedge B)$	$\neg$I, 8, 9, 10
12	$\mid$ $\neg(\neg A \wedge B)$	$\vee$E, 1, 2–6, 7–11
13	$(A \vee \neg B) \rightarrow \neg(\neg A \wedge B)$	$\rightarrow$I, 1, 12

ou

1	$\mid$ $A \vee \neg B$	Hyp
2	$\mid$ $\mid$ $\neg A \wedge B$	Hyp
3	$\mid$ $\mid$ $\neg A$	$\wedge$E, 2
4	$\mid$ $\mid$ B	$\wedge$E, 2
5	$\mid$ $\mid$ $A \vee \neg B$	Reit, 1
6	$\mid$ $\mid$ $\mid$ A	Hyp
7	$\mid$ $\mid$ $\mid$ $\mid$ B	Hyp
8	$\mid$ $\mid$ $\mid$ $\mid$ A	Reit, 6
9	$\mid$ $\mid$ $\mid$ $\mid$ $\neg A$	Reit, 3
10	$\mid$ $\mid$ $\mid$ $\neg B$	$\neg$I, 7, 8, 9
11	$\mid$ $\mid$ $\mid$ $\neg B$	Hyp
12	$\mid$ $\mid$ $\mid$ $\neg B$	Rep, 11
13	$\mid$ $\mid$ $\neg B$	$\vee$E, 5, 6–10, 11–12
14	$\mid$ $\neg(\neg A \wedge B)$	$\neg$I, 2, 4, 13
15	$(A \vee \neg B) \rightarrow \neg(\neg A \wedge B)$	$\rightarrow$I, 1, 14

8. $(\{\}, \neg(\neg A \vee \neg B) \to (A \wedge B)$ — argumento

1	$\neg(\neg A \vee \neg B)$	Hyp
2	$\quad \neg A$	Hyp
3	$\quad \neg A \vee \neg B$	$\vee$I, 2
4	$\quad \neg(\neg A \vee \neg B)$	Reit, 1
5	$\neg\neg A$	$\neg$I, 2, 3, 4
6	A	$\neg$E, 5
7	$\quad \neg B$	Hyp
8	$\quad \neg A \vee \neg B$	$\vee$I, 7
9	$\quad \neg(\neg A \vee \neg B)$	Reit, 1
10	$\neg\neg B$	$\neg$I, 7, 8, 9
11	B	$\neg$E, 10
12	$A \wedge B$	$\wedge$I, 6, 11
13	$\neg(\neg A \vee \neg B) \to (A \wedge B)$	$\to$I, 1, 12

Exercício 2.2
Considere o seguinte conjunto de fórmulas:

$$\{Homem \to Pessoa, Mulher \to Pessoa, Homem \vee Mulher\}$$

1. Mostre quais são os modelos desse conjunto.

2. *Pessoa* é consequência lógica desse conjunto? Porquê?

3. Acrescente $\neg Homem$ ao conjunto. Diga quais são os seus modelos e as suas consequências lógicas.

Resposta:

Considerando que:

- H representa *Homem*,
- M representa *Mulher* e
- P representa *Pessoa*.

1.

H	M	P	$H \to P$	$M \to P$	$H \vee M$	É modelo?
V	V	V	V	V	V	S
V	V	F	F	F	V	N
V	F	V	V	V	V	S
V	F	F	F	V	V	N
F	V	V	V	V	V	S
F	V	F	V	F	V	N
F	F	V	V	V	F	N
F	F	F	V	V	F	N

Um modelo de um conjunto de fórmulas é uma interpretação que satisfaz todas as fbfs desse conjunto, isto é, que as torna todas verdadeiras. Assim, os modelos deste conjunto de fórmulas correspondem às interpretações que atribuem a *Homem*, *Mulher* e *Pessoa* os valores das colunas respectivas, nas linhas em que o valor da última coluna é "S".

Convém notar que num dos modelos temos como consequência lógica $Homem \wedge Mulher$, pois não foi dito nada que o impedisse.

2. As consequências lógicas de um conjunto de fórmulas são as fórmulas que têm o valor V em qualquer modelo do conjunto. Assim, *Pessoa* é consequência lógica deste conjunto, porque tem o valor V em todos os modelos do conjunto.

   Intuitivamente, uma vez que temos $Homem \vee Mulher$, faz sentido que *Pessoa* seja consequência lógica do conjunto.

3. Para este novo conjunto, só interessa a metade inferior da tabela, em que *Homem* tem o valor F. O modelo é o que está assinalado na coluna à direita. As consequências lógicas deste novo conjunto são infinitas e incluem:

   - As fórmulas que estão no conjunto: $\neg Homem$, $Homem \to Pessoa$, $Mulher \to Pessoa$, $Homem \vee Mulher$,
   - as fórmulas atómicas às quais a interpretação atribui o valor verdadeiro: *Mulher* e *Pessoa*,
   - todas as fórmulas deriváveis a partir das fórmulas anteriores: $Mulher \wedge Pessoa$, $Mulher \vee Boneca$, $Homem \to Mulher$, etc.,
   - todos os teoremas da lógica proposicional: $A \to (B \to A)$, $(A \wedge \neg A) \to B$, $A \vee \neg A$, etc.

   Nota: se a pergunta fosse as fórmulas *deriváveis*, ainda seria necessário fazer as provas respectivas pela via sintáctica.

Exercício 2.3

Demonstre os seguintes teoremas usando o sistema de dedução natural da lógica proposicional.

1. $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$
2. $(((A \to (A \to B)) \wedge A) \to B)$
3. $(\neg A \to \neg B) \to (B \to A)$
4. $(A \wedge B) \to \neg(A \to \neg B)$
5. $((A \to B) \wedge \neg B) \to \neg A$
6. $(A \to \neg A) \to \neg A$
7. $(A \vee B) \to (B \vee A)$

8. $((A \lor B) \lor C) \to (A \lor (B \lor C))$

9. $(A \land (B \lor C)) \to ((A \land B) \lor (A \land C))$

10. $((A \land B) \lor A) \to A$

Resposta:

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

1	$\mid A \to (B \to C)$	Hyp
2	$\mid \mid A \to B$	Hyp
3	$\mid \mid \mid A$	Hyp
4	$\mid \mid \mid A \to B$	Reit, 2
5	$\mid \mid \mid B$	$\to$E, 3, 4
6	$\mid \mid \mid A \to (B \to C)$	Reit, 1
7	$\mid \mid \mid B \to C$	$\to$E, 3, 6
8	$\mid \mid \mid C$	$\to$E, 5, 7
9	$\mid \mid A \to C$	$\to$I, 3, 8
10	$\mid (A \to B) \to (A \to C)$	$\to$I, 2, 9
11	$(A \to (B \to C)) \to ((A \to B) \to (A \to C))$	$\to$I, 1, 10

1. $(((A \to (A \to B)) \land A) \to B)$

1	$\mid (A \to (A \to B)) \land A$	Hyp
2	$\mid A \to (A \to B)$	$\land$E, 1
3	$\mid A$	$\land$E, 1
4	$\mid A \to B$	$\to$E, 3, 2
5	$\mid B$	$\to$E, 3, 4
6	$(((A \to (A \to B)) \land A) \to B)$	$\to$I, 1, 5

2. $(\neg A \rightarrow \neg B) \rightarrow (B \rightarrow A)$

1	$\neg A \rightarrow \neg B$	Hyp
2	B	Hyp
3	$\neg A$	Hyp
4	$\neg A \rightarrow \neg B$	Reit, 1
5	$\neg B$	$\rightarrow$E, 3, 4
6	B	Reit, 2
7	$\neg\neg A$	$\neg$I, 3, 6, 5
8	A	$\neg$E, 7
9	$B \rightarrow A$	$\rightarrow$I, 2, 8
10	$(\neg A \rightarrow \neg B) \rightarrow (B \rightarrow A)$	$\rightarrow$I, 1, 9

3. $(A \wedge B) \rightarrow \neg(A \rightarrow \neg B)$

1	$A \wedge B$	Hyp
2	$A \rightarrow \neg B$	Hyp
3	$A \wedge B$	Reit, 1
4	A	$\wedge$E, 3
5	$\neg B$	$\rightarrow$E, 4, 2
6	B	$\wedge$E, 3
7	$\neg(A \rightarrow \neg B)$	$\neg$I, 2, 6, 5
8	$(A \wedge B) \rightarrow \neg(A \rightarrow \neg B)$	$\rightarrow$I, 1, 7

4. $((A \rightarrow B) \wedge \neg B) \rightarrow \neg A$

1	$(A \rightarrow B) \wedge \neg B$	Hyp
2	A	Hyp
3	$(A \rightarrow B) \wedge \neg B$	Reit, 1
4	$A \rightarrow B$	$\wedge$E, 3
5	B	$\rightarrow$E, 2, 4
6	$\neg B$	$\wedge$E, 3
7	$\neg A$	$\neg$I, 2, 5, 6
8	$((A \rightarrow B) \wedge \neg B) \rightarrow \neg A$	$\rightarrow$I, 1, 7

5. $(A \to \neg A) \to \neg A$

1	$A \to \neg A$	Hyp
2	A	Hyp
3	$A \to \neg A$	Reit, 1
4	$\neg A$	$\to$E, 2, 3
5	$\neg A$	$\neg$I, 2, 2, 4
6	$(A \to \neg A) \to \neg A$	$\to$I, 1, 5

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

1	$A \vee B$	Hyp
2	A	Hyp
3	$B \vee A$	$\vee$I, 2
4	B	Hyp
5	$B \vee A$	$\vee$I, 4
6	$B \vee A$	$\vee$E, 1, 2–3, 4–5
7	$(A \vee B) \to (B \vee A)$	$\to$I, 1, 6

7. $((A \vee B) \vee C) \to (A \vee (B \vee C))$

1	$(A \vee B) \vee C$	Hyp
2	$A \vee B$	Hyp
3	A	Hyp
4	$A \vee (B \vee C)$	$\vee$I, 3
5	B	Hyp
6	$B \vee C$	$\vee$I, 5
7	$A \vee (B \vee C)$	$\vee$I, 6
8	$A \vee (B \vee C)$	$\vee$E, 2, 3–4, 5–7
9	C	Hyp
10	$B \vee C$	$\vee$I, 9
11	$A \vee (B \vee C)$	$\vee$I, 10
12	$A \vee (B \vee C)$	$\vee$E, 1, 2–8, 9–11
13	$((A \vee B) \vee C) \to (A \vee (B \vee C))$	$\to$I, 1, 12

8. $(A \wedge (B \vee C)) \rightarrow ((A \wedge B) \vee (A \wedge C))$

1	$A \wedge (B \vee C)$	Hyp
2	A	$\wedge$E, 1
3	$B \vee C$	$\wedge$E, 1
4	$\quad\vert\ B$	Hyp
5	$\quad\vert\ A$	Reit, 2
6	$\quad\vert\ A \wedge B$	$\wedge$I, 5, 4
7	$\quad\vert\ (A \wedge B) \vee (A \wedge C)$	$\vee$I, 6
8	$\quad\vert\ C$	Hyp
9	$\quad\vert\ A$	Reit, 2
10	$\quad\vert\ A \wedge C$	$\wedge$I, 9, 8
11	$\quad\vert\ (A \wedge B) \vee (A \wedge C)$	$\vee$I, 10
12	$(A \wedge B) \vee (A \wedge C)$	$\vee$E, 1, 4–7, 8–11
13	$(A \wedge (B \vee C)) \rightarrow ((A \wedge B) \vee (A \wedge C))$	$\rightarrow$I, 1, 12

9. $((A \wedge B) \vee A) \rightarrow A$

1	$(A \wedge B) \vee A$	Hyp
2	$\quad\vert\ A \wedge B$	Hyp
3	$\quad\vert\ A$	$\wedge$E, 2
4	$\quad\vert\ A$	Hyp
5	$\quad\vert\ A$	Rep, 4
6	A	$\vee$E, 1, 2–3, 4–5
7	$((A \wedge B) \vee A) \rightarrow A$	$\rightarrow$I, 1, 6

Exercício 2.4
Usando o sistema sintáctico ou semântico, conforme o indicado, forneça provas para o seguinte:

1. $\{(A \vee B), \neg A\} \vdash B$
2. $\{\neg A \vee B\} \vdash \neg(A \wedge \neg B)$
3. $\{\neg(A \wedge \neg B)\} \vdash A \rightarrow B$
4. $\{A \vee B\} \vdash \neg(\neg A \wedge \neg B)$
5. $\{\neg(\neg A \wedge \neg B)\} \vdash (A \vee B)$
6. $\{\neg A \vee B, A\} \models B$
7. $\{(A \rightarrow B) \wedge C)\} \models (\neg A \wedge C) \vee (B \wedge C)$

Resposta:

1. $\{(A \vee B), \neg A\} \vdash B$

1	$A \vee B$	Prem
2	$\neg A$	Prem
3	A	Hyp
4	$\neg B$	Hyp
5	A	Reit, 3
6	$\neg A$	Reit, 2
7	$\neg\neg B$	$\neg$I, 4, 5, 6
8	B	$\neg$E, 7
9	B	Hyp
10	B	Rep, 9
11	B	$\vee$E, 1, 3–8, 9–10

2. $\{\neg A \vee B\} \vdash \neg(A \wedge \neg B)$

1	$\neg A \vee B$	Prem
2	$\neg A$	Hyp
3	$A \wedge \neg B$	Hyp
4	A	$\wedge$I, 3
5	$\neg A$	Reit, 2
6	$\neg(A \wedge \neg B)$	$\neg$I, 3, 4, 5
7	B	Hyp
8	$A \wedge \neg B$	Hyp
9	B	Reit, 7
10	$\neg B$	$\wedge$I, 8
11	$\neg(A \wedge \neg B)$	$\neg$I, 8, 9, 10
12	$\neg(A \wedge \neg B)$	$\vee$E, 1, 2–6, 7–11

3. $\{\neg(A \wedge \neg B)\} \vdash A \to B$

1	$\neg(A \wedge \neg B)$	Prem
2	$\mid\ \underline{A}$	Hyp
3	$\mid\ \mid\ \underline{\neg B}$	Hyp
4	$\mid\ \mid\ A$	Reit, 2
5	$\mid\ \mid\ A \wedge \neg B$	$\wedge$I, 4, 3
6	$\mid\ \mid\ \neg(A \wedge \neg B)$	Reit, 1
7	$\mid\ \neg\neg B$	$\neg$I, 3, 5, 6
8	$\mid\ B$	$\neg$E, 7
9	$A \to B$	$\to$I, 2, 8

4. $\{A \vee B\} \vdash \neg(\neg A \wedge \neg B)$

1	$A \vee B$	Prem
2	$\mid\ \underline{A}$	Hyp
3	$\mid\ \mid\ \underline{\neg A \wedge \neg B}$	Hyp
4	$\mid\ \mid\ A$	Reit, 2
5	$\mid\ \mid\ \neg A$	$\wedge$E, 3
6	$\mid\ \neg(\neg A \wedge \neg B)$	$\neg$I, 3, 4, 5
7	$\mid\ \underline{B}$	Hyp
8	$\mid\ \mid\ \underline{\neg A \wedge \neg B}$	Hyp
9	$\mid\ \mid\ B$	Reit, 7
10	$\mid\ \mid\ \neg B$	$\wedge$E, 8
11	$\mid\ \neg(\neg A \wedge \neg B)$	$\neg$I, 8, 9, 10
12	$\neg(\neg A \wedge \neg B)$	$\vee$E, 1, 2–6, 7–11

5. $\{\neg(\neg A \wedge \neg B)\} \vdash (A \vee B)$

1	$\neg(\neg A \wedge \neg B)$	Prem
2	$\quad \neg(A \vee B)$	Hyp
3	$\quad\quad A$	Hyp
4	$\quad\quad A \vee B$	$\vee$I, 3
5	$\quad\quad \neg(A \vee B)$	Reit, 2
6	$\quad \neg A$	$\neg$I, 3, 4, 5
7	$\quad\quad B$	Hyp
8	$\quad\quad A \vee B$	$\vee$I, 7
9	$\quad\quad \neg(A \vee B)$	Reit, 2
10	$\quad \neg B$	$\neg$I, 7, 8, 9
11	$\quad \neg A \wedge \neg B$	$\wedge$I, 6, 10
12	$\quad \neg(\neg A \wedge \neg B)$	Reit, 1
13	$\neg\neg(A \vee B)$	$\neg$I, 2, 11, 12
14	$A \vee B$	$\neg$E, 13

6. $\{\neg A \vee B, A\} \models B$
   tabela

7. $\{(A \rightarrow B) \wedge C)\} \models (\neg A \wedge C) \vee (B \wedge C)$
   tabela

Exercício 2.5
(JPM) Prove os seguintes teoremas, usando o sistema dedutivo da lógica de primeira ordem:

1. $(F(a) \wedge \forall(x)[F(x) \rightarrow G(x)]) \rightarrow G(a)$
2. $(\forall(x)[F(x) \rightarrow G(x)] \wedge \forall(y)[G(y) \rightarrow H(y)]) \rightarrow \forall(z)[F(z) \rightarrow H(z)]$
3. $(\forall(x)[F(x) \rightarrow H(x)] \wedge \exists(y)[F(y)]) \rightarrow \exists(z)[H(z)]$

Resposta:

1. $(F(a) \wedge \forall(x)[F(x) \rightarrow G(x)]) \rightarrow G(a)$

1	$\quad F(a) \wedge \forall(x)[F(x) \rightarrow G(x)]$	Hyp
2	$\quad F(a)$	$\wedge$E, 1
3	$\quad \forall(x)[F(x) \rightarrow G(x)]$	$\wedge$E, 1
4	$\quad F(a) \rightarrow G(a)$	$\forall$E, 3
5	$\quad G(a)$	$\rightarrow$E, 2, 4
6	$(F(a) \wedge \forall(x)[F(x) \rightarrow G(x)]) \rightarrow G(a)$	$\rightarrow$I, 1, 5

2. $(\forall(x)[F(x) \to G(x)] \land \forall(y)[G(y) \to H(y)]) \to \forall(z)[F(z) \to H(z)]$

1	$\forall(x)[F(x) \to G(x)] \land \forall(y)[G(y) \to H(y)]$	Hyp
2	$\forall(x)[F(x) \to G(x)]$	$\land$E, 1
3	$\forall(y)[G(y) \to H(y)]$	$\land$E, 1
4	x_0 $F(x_0)$	Hyp
5	$\forall(x)[F(x) \to G(x)]$	Reit, 2
6	$F(x_0) \to G(x_0)$	$\forall$E, 5
7	$G(x_0)$	$\to$E, 4, 6
8	$\forall(y)[G(y) \to H(y)]$	Reit, 3
9	$G(x_0) \to H(x_0)$	$\forall$E, 8
10	$H(x_0)$	$\to$E, 7, 9
11	$F(x_0) \to H(x_0)$	$\to$I, 4, 10
12	$\forall(z)[F(z) \to H(z)]$	$\forall$I, 4, 11
13	$(\forall(x)[F(x) \to G(x)] \land \forall(y)[G(y) \to H(y)]) \to \forall(z)[F(z) \to H(z)]$	$\to$I, 1, 12

3. $(\forall(x)[F(x) \to H(x)] \land \exists(y)[F(y)]) \to \exists(z)[H(z)]$

1	$\forall(x)[F(x) \to H(x)] \land \exists(y)[F(y)]$	Hyp
2	$\forall(x)[F(x) \to H(x)]$	$\land$E, 1
3	$\exists(y)[F(y)]$	$\land$E, 1
4	x_0 $F(x_0)$	Hyp
5	$\forall(x)[F(x) \to H(x)]$	Reit, 2
6	$F(x_0) \to H(x_0)$	$\forall$E, 5
7	$H(x_0)$	$\to$E, 4, 6
8	$\exists(z)[H(z)]$	$\exists$I, 7
9	$\exists(z)[H(z)]$	$\exists$E, 3, 4–8
10	$(\forall(x)[F(x) \to H(x)] \land \exists(y)[F(y)]) \to \exists(z)[H(z)]$	$\to$I, 1, 9

3 Lógica clássica — Modelação de domínio

Exercício 3.1
Considere a seguinte informação acerca de mamíferos. Com base nela e no conhecimento (de senso comum) que tem acerca do mundo, represente-a usando lógica de primeira ordem. Note que alguma desta informação pode ser muito difícil de representar; se for esse o caso, indique-o explicitamente e proponha uma solução.

Existem aproximadamente 4500 espécies de mamíferos no mundo, e todos eles respiram oxigénio do ar. Os mamíferos também são caracterizados por cuidarem das suas crias enquanto bebés e por as alimentarem de leite materno.

As fêmeas dos mamíferos desenvolvem as crias dentro do ventre e quando chegam ao fim da gestação podem ter entre 1 e 27 crias.

Uma das excepções a esta regra são os monotremas (que incluem os ornitorrincos e as equidnas), que põem ovos e os incubam, para as crias se desenvolverem. Os machos dos ornitorrincos são também caracterizados, entre outras coisas, por terem espigões venenosos nas patas traseiras.

Os humanos, em contrapartida, podem ser caracterizados pelo índice de massa corporal: peso em kilos a dividir pela altura em metros ao quadrado. Este valor é interpretado da seguinte forma:

IMC inferior 18,5	Peso abaixo do normal
IMC de 18,5 a 25	Peso Normal
IMC de 25 a 29,9	Excesso de Peso
IMC superior 30	Obesidade

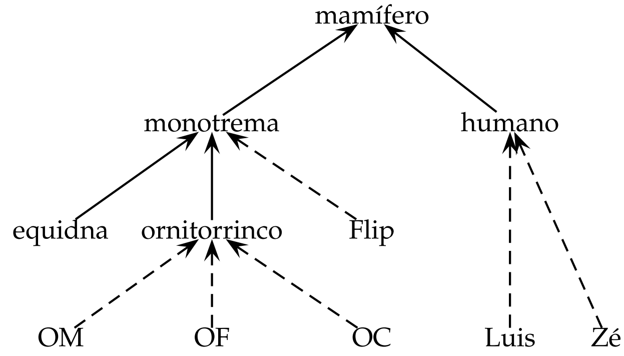
O Luís é um humano que pesa 90Kg e mede 1,90m. O Zé é uma cria do Luis.

A OF é um ornitorrinco fêmea, o OM é um ornitorrinco macho e a OC é uma cria de OF.

Há um mamífero (chamemos-lhe Flip) que ou é um ornitorrinco ou é uma equidna.

Resposta:

Em primeiro lugar, devemos decidir que hierarquia é que vamos usar.



Em que:

$a \longrightarrow b$ significa que todos os *a*s são *b*s
$A \dashrightarrow b$ significa que este A é um *b*

Os factos de ser macho ou fêmea, de cuidar das crias, de desenvolver as crias dentro do ventre, de pôr ovos, etc vão ser representados como propriedades das várias classes e instâncias.

Existem aproximadamente 4500 espécies de mamíferos.
$AproxIgual(cardinDe(EspMam), 4500)$

$\forall(x, y)[(Pertence(x, EspMam) \wedge TemEsp(y, x)) \rightarrow Mam(y)]$

Todos os mamíferos respiram oxigénio do ar.
$\forall(x)[Mam(x) \rightarrow Respira(x, OxigenioAr)]$

Os mamíferos cuidam das suas crias enquanto bebés.
$\forall(x, y)[(Mam(x) \wedge Cria(y, x)) \rightarrow (Mam(y) \wedge (Bebe(y) \rightarrow Cuida(x, y)))]$

> Nota: se quisermos falar acerca das espécies dos animais, por exemplo, para dizer que as crias de um animal têm a mesma espécie que ele, devemos usar $TemEsp(x, Mam)$ em vez de $Mam(x)$. Desta forma, estaríamos a "coisificar" as espécies e poderíamos ter:
> $\forall(x, y, z)[(TemEsp(x, y) \wedge Cria(z, y)) \rightarrow TemEsp(z, y)]$
> Esta abordagem também permitiria dizer que os animais apenas têm uma espécie, coisa que com a representação mais simples não é possível:
> $\forall(x, y, z)[(TemEsp(x, y) \wedge \neg Igual(y, z)) \rightarrow \neg TemEsp(x, z)]$

Os mamíferos (que são fêmeas) alimentam as crias com leite materno.
$\forall(x, y)[(Mam(x) \wedge Femea(x) \wedge Cria(y, x)) \rightarrow Alimenta(x, y, LeiteMaterno)]$

As fêmeas dos mamíferos (que não sejam monotremas) desenvolvem as crias dentro do ventre.
$\forall(x, y)[(Mam(x) \wedge Femea(x) \wedge Cria(y, x) \wedge \neg Monotrema(x)) \rightarrow DesenvolveDentro(x, y, ventreDe(x))]$

> Nota: Esta regra não pode ser aplicada a outros mamíferos, a não ser que seja explicitamente dito que não são monotremas.

As gestações das fêmeas dos mamíferos podem ter entre 1 e 27 crias.
$\forall(x, y)[(Mam(x) \wedge Femea(x) \wedge Gestacao(y, x)) \rightarrow Entre(numCriasDe(y), 1, 27)]$

Os monotremas são mamíferos.
$\forall(x)[Monotrema(x) \rightarrow Mam(x)]$

Os ornitorrincos são monotremas.
$\forall(x)[Ornitorrinco(x) \rightarrow Monotrema(x)]$

As equidnas são monotremas.
$\forall(x)[Equidna(x) \rightarrow Monotrema(x)]$

As fêmeas dos monotremas põem ovos e incubam-nos.
$\forall(x)[(Monotrema(x) \wedge Femea(x)) \rightarrow (Poe(x, ovosDe(x)) \wedge Incuba(x, ovosDe(x)))]$

As fêmeas dos monotremas não desenvolvem as crias dentro do ventre.
$\forall(x, y)[(Monotrema(x) \wedge Femea(x) \wedge Cria(y, x)) \rightarrow \neg DesenvolveDentro(x, y, ventreDe(x))]$

Os machos dos ornitorrincos têm espigões venenosos nas patas traseiras.
$$\forall(x)[(Ornitorrinco(x) \wedge \neg Femea(x)) \rightarrow (Tem(x, espigaoDe(x)) \wedge Venenoso(espigaoDe(x)) \\ \wedge Localizacao(espigaoDe(x), pataTraseiraDe(x)))]$$

Os humanos são mamíferos.
$\forall(x)[Humano(x) \to Mam(x)]$

Os humanos não são monotremas.
$\forall(x)[Humano(x) \to \neg Monotrema(x)]$

> Nota: Esta regra é necessária para podermos aplicar aos humanos as regras que têm como excepção os monotremas.

O IMC dos humanos é igual ao peso a dividir pela altura ao quadrado.
$\forall(x)[Humano(x) \to Igual(imcDe(x), divisaoDe(pesoDe(x), quadradoDe(alturaDe(x))))]$

Os humanos com IMC menor que 18.5 têm peso inferior ao normal.
$\forall(x)[(Humano(x) \wedge Menor(imcDe(x), 18.5)) \to TipoPeso(pesoDe(x), PesoInferiorNormal)]$

Os humanos com IMC entre 18.5 e 25 têm peso normal.
$\forall(x)[(Humano(x) \wedge Entre(imcDe(x), 18.5, 25)) \to TipoPeso(pesoDe(x), PesoNormal)]$

Os humanos com IMC entre 25 e 29.9 têm peso excessivo.
$\forall(x)[(Humano(x) \wedge Entre(imcDe(x), 25, 29.9)) \to TipoPeso(pesoDe(x), PesoExcessivo)]$

Os humanos com IMC maior que 30 sofrem de obesidade.
$\forall(x)[(Humano(x) \wedge Maior(imcDe(x), 30)) \to TipoPeso(pesoDe(x), Obesidade)]$

O Luis é humano, pesa 90 kilos e mede 1.90 metros.
$Humano(Luis)$

$Igual(pesoDe(Luis), 90)$

$Igual(alturaDe(Luis), 1.9)$

O Zé é cria do Luis.
$Cria(Ze, Luis)$

A OF é um ornitorrinco fêmea.
$Ornitorrinco(OF) \wedge Femea(OF)$

O OM é um ornitorrinco macho.
$Ornitorrinco(OM) \wedge \neg Femea(OM)$

A OC é uma cria de OF.
$Cria(OC, OF)$

O Flip ou é um ornitorrinco ou é uma equidna.
$((Ornitorrinco(Flip) \wedge \neg Equidna(Flip)) \vee$
$(\neg Ornitorrinco(Flip) \wedge Equidna(Flip))$

Exercício 3.2
Considere as seguintes definições:

- $\delta_1 = \forall(x)[Mamifero(x) \to Respira(x, OxigenioAr)]$
- $\delta_2 = \forall(x,y)[(Mamifero(x) \wedge Cria(y,x)) \to (Mamifero(y) \wedge Cuida(x,y))]$
- $\delta_3 = \forall(x)[Humano(x) \to Mamifero(x)]$
- $\delta_4 = \forall(x,y)[(Mamifero(x) \wedge Femea(x) \wedge Cria(y,x)) \to Alimenta(x, y, LeiteMaterno)]$
- $\Delta = \{\delta_1, \delta_2, \delta_3, \delta_4\}$

- $\Delta_1 = \Delta \cup \{Humano(Luis)\}$
- $\Delta_2 = \Delta_1 \cup \{Cria(Ze, Luis)\}$

1. Prove os seguintes argumentos, usando o sistema dedutivo da lógica de primeira ordem:

   (a) $(\Delta_1, Respira(Luis, OxigenioAr))$

   (b) $(\Delta_2, Cuida(Luis, Ze))$

2. Consegue provar $(\Delta_2, \neg Alimenta(Luis, Ze, LeiteMaterno))$? Porquê?

3. E $(\Delta_2 \cup \{\neg Femea(Luis)\}, \neg Alimenta(Luis, Ze, LeiteMaterno))$? Porquê?

Resposta:

1. (a) $(\Delta_1, Respira(Luis, OxigenioAr))$

1	$Humano(Luis)$	Prem
2	$\forall(x)[Humano(x) \rightarrow Mamifero(x)]$	Prem
3	$Humano(Luis) \rightarrow Mamifero(Luis)$	$\forall$E, 2
4	$Mamifero(Luis)$	$\rightarrow$E, 1, 3
5	$\forall(x)[Mamifero(x) \rightarrow Respira(x, OxigenioAr)]$	Prem
6	$Mamifero(Luis) \rightarrow Respira(Luis, OxigenioAr)$	$\forall$E, 5
7	$Respira(Luis, OxigenioAr)$	$\rightarrow$E, 4, 6

   (b) $(\Delta_2, Cuida(Luis, Ze))$

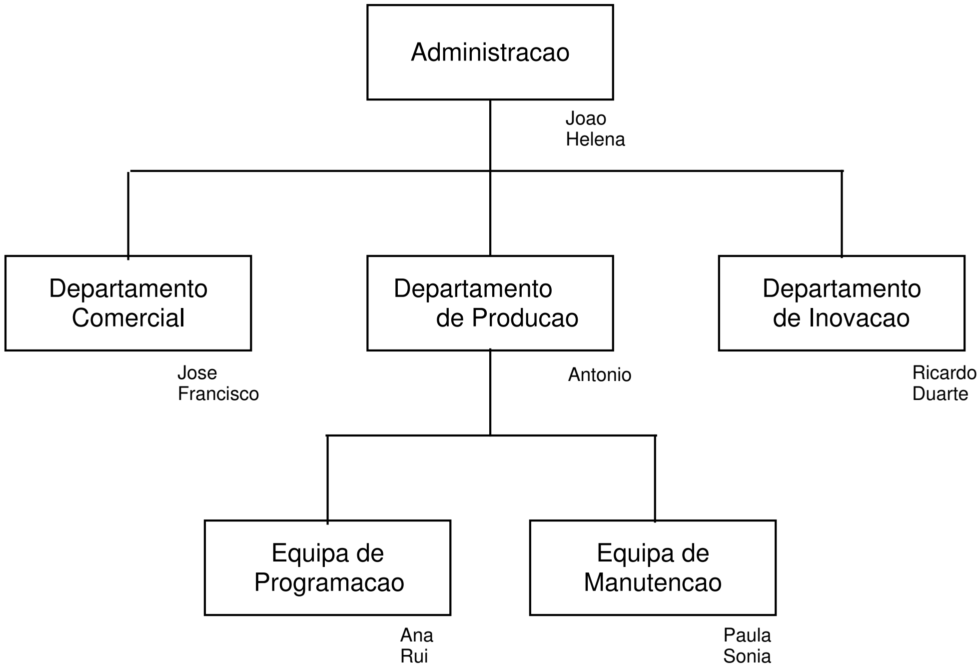
1	$Humano(Luis)$	Prem
2	$\forall(x)[Humano(x) \rightarrow Mamifero(x)]$	Prem
3	$Humano(Luis) \rightarrow Mamifero(Luis)$	$\forall$E, 2
4	$Mamifero(Luis)$	$\rightarrow$E, 1, 3
5	$\forall(x, y)[(Mamifero(x) \wedge Cria(y, x)) \rightarrow (Mamifero(y) \wedge Cuida(x, y))]$	Prem
6	$Cria(Ze, Luis)$	Prem
7	$Mamifero(Luis) \wedge Cria(Ze, Luis)$	$\wedge$I, 4, 6
8	$(Mamifero(Luis) \wedge Cria(Ze, Luis)) \rightarrow (Mamifero(Ze) \wedge Cuida(Luis, Ze))$	$\forall$E, 5
9	$Mamifero(Ze) \wedge Cuida(Luis, Ze)$	$\rightarrow$E, 7, 8
10	$Cuida(Luis, Ze)$	$\wedge$E, 9

2. Não é possível provar $(\Delta_2, \neg Alimenta(Luis, Ze, LeiteMaterno))$ porque não temos informação que diga se o Luis é fêmea.

3. Mesmo sabendo que o Luis não é fêmea, continuamos a não conseguir provar o argumento $(\Delta_2 \cup \{\neg Femea(Luis)\}, \neg Alimenta(Luis, Ze, LeiteMaterno))$ porque o facto de termos a negação do antecedente de uma implicação não nos permite concluir a negação do seu consequente.

   Nenhum destes dois últimos argumentos é válido.

Exercício 3.3
(Adaptado do TPC1 de 2003/04) Existe uma empresa de desenvolvimento de software com o seguinte organigrama:



Cada departamento tem um chefe e cada equipa tem um chefe. Cada chefe depende do chefe da unidade hierarquicamente acima. Os chefes de cada unidade são as pessoas cujo nome aparece em primeiro lugar junto a essa unidade.

Cada pessoa pode ter um conjunto de competências, por exemplo:

- Conhecimento de programação
- Domínio de línguas estrangeiras
- Capacidade de chefia
- etc

Discuta possíveis representações para esta informação, em lógica de primeira ordem. Tenha em consideração o seu conhecimento de senso comum acerca do mundo e suponha que se pretendem seleccionar pessoas para um determinado projecto com base nas suas competências. Em particular, pretende-se uma pessoa para chefiar um projecto na Grã-Bretanha.

Resposta:

Convém ter em atenção que esta resolução se destina a auxiliar a resolução do TPC e que não vai ser completa...

$\forall(x)[Dept(x) \rightarrow \exists(y)(Pessoa(y) \wedge Chefe(y,x))]$

$\forall(x)[Equipa(x) \rightarrow \exists(y)(Pessoa(y) \wedge Chefe(y,x))]$

Podíamos ter uma disjunção no antecedente e ter apenas uma regra, mas assim é mais modular e mais eficiente.

$Dept(Admin)$
$Dept(DComercial)$
$Dept(DProducao)$

$Dept(DInovacao)$
$Equipa(EProgramacao)$
$Equipa(EManutencao)$

$Acima(Admin, DComercial)$
$Acima(Admin, DProducao)$
$Acima(Admin, DInovacao)$
$Acima(DProducao, EProgramacao)$
$Acima(DProducao, EManutencao)$

Transitividade de Acima:
$\forall(x, y, z)[(Acima(x, y) \land Acima(y, z)) \rightarrow Acima(x, z)]$

Algumas relações de chefia: $Chefe(Joao, Admin)$
$Chefe(Antonio, DProducao)$
$Chefe(Ana, EProgramacao)$

Os chefes de pessoas. Estamos a usar o mesmo predicado para representar chefes de unidades e chefes de pessoas. Isto estará correcto?
$\forall(x, y, z, w)[(Chefe(x, y) \land Acima(z, y) \land Chefe(w, z)) \rightarrow Chefe(w, x)]$

Algumas competências, para servirem de exemplo:
$TemComp(Paula, Programacao)$
$TemComp(Paula, CapChefia)$
$TemComp(Paula, Ingles)$
$TemComp(Sonia, Ingles)$
$TemComp(Antonio, CapChefia)$

Discutir acerca da necessidade de incluir também que as pessoas desta empresa falam português. Melhor ainda seria considerar a língua materna, com base na nacionalidade.

Discutir necessidade do "dicionário", em particular quando existem predicados com mais de um argumento, para não existir ambiguidade quanto à ordem dos argumentos.

Solução em SNePSLOG:

```
all(x)(Dept(x) => exists(y)(Pessoa(y) and Chefe(y,x)))
all(x)(Equipa(x) => exists(y)(Pessoa(y) and Chefe(y,x)))
; Podíamos ter uma disjunção no antecedente e ter apenas uma
; regra, mas assim é mais modular e mais eficiente.

Dept(Admin)
Dept(DComercial)
Dept(DProducao)
Dept(DInovacao)
Equipa(EProgramacao)
Equipa(EManutencao)

Acima(Admin, DComercial)
Acima(Admin, DProducao)
Acima(Admin, DInovacao)
Acima(DProducao, EProgramacao)
```

```
Acima(DProducao, EManutencao)

;;;;;;;;;;;;;;;;;;;;
; Transitividade de Acima

all(x,y,z)((Acima(x,y) and Acima(y,z)) => Acima(x,z))

Chefe(Joao, Admin)
Chefe(Antonio, DProducao)
Chefe(Ana, EProgramacao)

all(x,y,z,w)((Chefe(x,y) and Acima(z,y) and Chefe(w,z)) =>
             Chefe(w,x))

; Algumas competencias, para servirem de exemplo
TemComp(Paula, Programacao)
TemComp(Paula, CapChefia)
TemComp(Paula, Ingles)
TemComp(Sonia, Ingles)
TemComp(Antonio, CapChefia)
;...


;;;;;;;;;;;;;;;;;;;;
; Factos e perguntas

Acima(?x,?y)?
; Infere Acima(Admin, EProgramacao) e Acima(Admin, EManutencao)

Chefe(?x, Ana)?
; Infere Chefe(Joao, Ana) e Chefe(Antonio, Ana)
; Estamos a usar o predicado Chefe para ChefeDeUnidade e ChefeDePessoa.
; Possivelmente, deveriam ser predicados diferentes...

; Para chefiar um projecto, deve ter CapChefia.
; Para a Grã-Bretanha, deve falar Ingles
TemComp(?x, CapChefia) and TemComp(?x, Ingles)?
; Infere TemComp(Paula, CapChefia) and TemComp(Paula, Ingles)
```

4 Lógica não clássica — Lógica da implicação relevante

Sumário:

- Revisão das provas para os dois paradoxos da implicação em lógica clássica.
- Lógica da implicação relevante: provas usando o sistema dedutivo.

Resumo:

Revisão do sistema dedutivo da lógica clássica
Provar em lógica clássica os seguintes teoremas, utilizando o sistema de dedução natural.

1. $A \to (B \to A)$ — Primeiro paradoxo da implicação: qualquer proposição implica uma proposição verdadeira.

1	A	Hyp
2	$\quad B$	Hyp
3	$\quad A$	Reit, 1
4	$B \to A$	$\to$I, 2, 3
5	$A \to (B \to A)$	$\to$I, 1, 4

2. $(A \wedge \neg A) \to B$ — Segundo paradoxo da implicação: uma contradição implica qualquer proposição.

1	$A \wedge \neg A$	Hyp
2	$\quad \neg B$	Hyp
3	$\quad A \wedge \neg A$	Reit, 1
4	$\quad A$	$\wedge$E, 3
5	$\quad \neg A$	$\wedge$E, 3
6	$\neg\neg B$	$\neg$I, 2, 4, 5
7	B	$\neg$E, 6
8	$(A \wedge \neg A) \to B$	$\to$I, 1, 7

Tal como em lógica clássica, permitimos que se usem algumas das regras com mais graus de liberdade.

Regras de inferência da lógica da implicação relevante usadas:

Lógica da implicação relevante — Sistema dedutivo

Hyp

$$\begin{array}{l|ll}
n & \underline{\alpha, \{k\}} & \text{Hyp} \\
n+1 & \ldots &
\end{array}$$

Rep

$$\begin{array}{lll}
n & \alpha, a & \\
\vdots & \vdots & \\
m & \alpha, a & \text{Rep}, n
\end{array}$$

Reit

$$\begin{array}{llll}
n & \alpha, a & & \\
\vdots & & \vert\ \vdots & \\
m & & \vert\ \alpha, a & \text{Reit}, n
\end{array}$$

$\Rightarrow$ I

$$\begin{array}{l|ll}
n & \underline{\alpha, \{\boldsymbol{k}\}} & \text{Hyp} \\
\vdots & \vdots & \\
m & \beta, \boldsymbol{a} \cup \{\boldsymbol{k}\} & \\
\end{array}$$
$$\begin{array}{lll}
m+1 & \alpha \Rightarrow \beta, a & \Rightarrow\text{I}, n, m
\end{array}$$

$\Rightarrow$ E

$$\begin{array}{lll}
n & \alpha, a & \\
n+1 & \alpha \Rightarrow \beta, b & \\
n+2 & \beta, a \cup b & \Rightarrow\text{E}, n, n+1
\end{array}$$

$\wedge$I

$$\begin{array}{lll}
n & \alpha, \boldsymbol{a} & \\
n+1 & \beta, \boldsymbol{a} & \\
n+2 & \alpha \wedge \beta, \boldsymbol{a} & \wedge\text{I}, n, n+1
\end{array}$$

$\wedge$E

$$\begin{array}{lll}
n & \alpha \wedge \beta, a & \\
n+1 & \alpha, a & \wedge\text{E}, n
\end{array}$$

e

$$\begin{array}{lll}
n & \alpha \wedge \beta, a & \\
n+1 & \beta, a & \wedge\text{E}, n
\end{array}$$

$\vee$I

$$\begin{array}{lll}
n & \alpha, a & \\
n+1 & \alpha \vee \beta, a & \vee\text{I}, n
\end{array}$$

e

$$\begin{array}{lll}
n & \alpha, a & \\
n+1 & \beta \vee \alpha, a & \vee\text{I}, n
\end{array}$$

$\vee$E

$$\begin{array}{llll}
n & \alpha \vee \beta, a & & \\
o & & \vert\ \underline{\alpha, \{\boldsymbol{k}\}} & \text{Hyp} \\
\vdots & & \vert\ \vdots & \\
p & & \vert\ \gamma, \boldsymbol{b} \cup \{\boldsymbol{k}\} & \\
r & & \vert\ \underline{\beta, \{\boldsymbol{l}\}} & \text{Hyp} \\
\vdots & & \vert\ \vdots & \\
s & & \vert\ \gamma, \boldsymbol{b} \cup \{\boldsymbol{l}\} & \\
m & \gamma, a \cup b & & \vee\text{E}, n, o\text{–}p, r\text{–}s
\end{array}$$

$\neg$I

$$\begin{array}{lll}
n & \alpha \Rightarrow \neg\alpha, a & \\
n+1 & \neg\alpha, a & \neg\text{I}, n
\end{array}$$

$\neg$E

$$\begin{array}{lll}
n & \neg\beta, a & \\
n+1 & \alpha \Rightarrow \beta, b & \\
n+2 & \neg\alpha, a \cup b & \neg\text{E}, n, n+1
\end{array}$$

$\neg\neg$I

$$\begin{array}{lll}
n & \alpha, a & \\
n+1 & \neg\neg\alpha, a & \neg\neg\text{I}, n
\end{array}$$

$\neg\neg$E

$$\begin{array}{lll}
n & \neg\neg\alpha, a & \\
n+1 & \alpha, a & \neg\neg\text{E}, n
\end{array}$$

Exercício 4.1
Prove na lógica da implicação relevante os seguintes teoremas ("correspondentes" aos teoremas, argumentos e provas dos exercícios 2.1, 2.3 e 2.4 da lógica clássica). Caso não consiga provar algum, explique porquê, dizendo qual ou quais as regras que não o permitiram.

1. $A \Rightarrow (B \Rightarrow A)$
2. $(A \wedge \neg A) \Rightarrow B$
3. $A \Rightarrow (B \Rightarrow (A \wedge B))$
4. $\neg(A \vee B) \Rightarrow (\neg A \wedge \neg B)$
5. $(\neg A \Rightarrow \neg B) \Rightarrow (B \Rightarrow A)$
6. $(A \wedge B) \Rightarrow \neg(A \Rightarrow \neg B)$
7. $((A \Rightarrow B) \wedge \neg B) \Rightarrow \neg A$
8. $((\neg A \vee B) \wedge \neg B) \Rightarrow \neg A$
9. $(A \Rightarrow \neg A) \Rightarrow \neg A$
10. $((A \vee B) \vee C) \Rightarrow (A \vee (B \vee C))$
11. $(A \wedge (B \vee C)) \Rightarrow ((A \wedge B) \vee (A \wedge C))$
12. $(A \vee (B \wedge C)) \Rightarrow ((A \vee B) \wedge (A \vee C))$
13. $(\neg A \vee B) \Rightarrow \neg(A \wedge \neg B)$
14. $((A \Rightarrow B) \wedge (B \Rightarrow \neg A)) \Rightarrow \neg A$
15. $((A \Rightarrow B) \wedge (B \Rightarrow C)) \Rightarrow (A \Rightarrow C)$
16. $(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$
17. $(((A \Rightarrow (A \Rightarrow B)) \wedge A) \Rightarrow B)$
18. $(A \vee B) \Rightarrow (B \vee A)$
19. $((A \wedge B) \vee A) \Rightarrow A$
20. $((A \vee B) \wedge \neg A) \Rightarrow B$
21. $\neg(A \wedge \neg B) \Rightarrow (A \Rightarrow B)$
22. $(A \vee B) \Rightarrow \neg(\neg A \wedge \neg B)$
23. $\neg(\neg A \wedge \neg B) \Rightarrow (A \vee B)$
24. $((\neg A \vee B) \wedge A) \Rightarrow B$

Resposta:

1. $A \Rightarrow (B \Rightarrow A)$

1	$A, \{1\}$	Hyp
2	$B, \{2\}$	Hyp
3	$A, \{1\}$	Reit, 1
4	$B \Rightarrow A, \{??\}$	**Impossível**
	$\vdots$	
n	$A \Rightarrow (B \Rightarrow A), \{\}$	$\Rightarrow$I, 1, 4

Este é o primeiro paradoxo da implicação.
Não podemos aplicar a regra $\Rightarrow I$ para ficarmos com $B \Rightarrow A$ porque $\{1\}$ e $\{2\}$ são disjuntos.

Conclusão: Não conseguimos fazer uma prova equivalente à de lógica clássica para este teorema. Apesar disto, não ficou provado que ele não se consegue provar na lógica implicação relevante (poderia haver alguma outra forma de fazer).

2. $(A \land \neg A) \Rightarrow B$

1	$A \land \neg A, \{1\}$	Hyp
2	$\neg B, \{2\}$	Hyp
3	$A \land \neg A, \{1\}$	Reit, 1
4	$A, \{1\}$	$\land$E, 3
5	$\neg A, \{1\}$	$\land$E, 3
	$\vdots$	
n	$\neg\neg B, \{1, 2\}$	**Impossível**
$n+1$	$\neg B \Rightarrow \neg\neg B, \{1\}$	$\Rightarrow$I, 2, n
$n+2$	$\neg\neg B, \{1\}$	$\neg$I, $n+1$
$n+3$	$B, \{1\}$	$\neg\neg$E, $n+2$
$n+4$	$(A \land \neg A) \Rightarrow B, \{\}$	$\Rightarrow$I, 1, $n+3$

Este é o segundo paradoxo da implicação.
Não podemos aplicar a regra $\neg I$ porque não conseguimos derivar $\neg B \Rightarrow \neg\neg B$. Por isso, não conseguimos derivar qualquer proposição a partir duma contradição.

3. $A \Rightarrow (B \Rightarrow (A \land B))$

1	$A, \{1\}$	Hyp
2	$B, \{2\}$	Hyp
3	$A, \{1\}$	Reit, 1
4	$A \land B, \{??\}$	**Impossível**
	$\vdots$	
n	$B \Rightarrow (A \land B), \{1\}$	$\Rightarrow$I, 2, 4
$n+1$	$A \Rightarrow (B \Rightarrow (A \land B)), \{\}$	$\Rightarrow$I, 1, n

Não se pode aplicar a regra da introdução da conjunção porque os conjuntos suporte de A e de B não são iguais.

4. $\neg(A \vee B) \Rightarrow (\neg A \wedge \neg B)$

1	$\neg(A \vee B), \{1\}$	Hyp
2	$\quad A, \{2\}$	Hyp
3	$\quad A \vee B, \{2\}$	$\vee$I, 2
4	$A \Rightarrow (A \vee B), \{\}$	$\Rightarrow$I, 2, 3
5	$\neg A, \{1\}$	$\neg$E, 1, 4
6	$\quad B, \{3\}$	Hyp
7	$\quad A \vee B, \{3\}$	$\vee$I, 6
8	$B \Rightarrow (A \vee B), \{\}$	$\Rightarrow$I, 6, 7
9	$\neg B, \{1\}$	$\neg$E, 1, 8
10	$\neg A \wedge \neg B, \{1\}$	$\wedge$I, 5, 9
11	$\neg(A \vee B) \Rightarrow (\neg A \wedge \neg B), \{\}$	$\Rightarrow$I, 1, 10

5. $(\neg A \Rightarrow \neg B) \Rightarrow (B \Rightarrow A)$

1	$\neg A \Rightarrow \neg B, \{1\}$	Hyp
2	$\quad B, \{2\}$	Hyp
3	$\quad \neg\neg B, \{2\}$	$\neg\neg$I, 2
4	$\quad \neg A \Rightarrow \neg B, \{1\}$	Reit, 1
5	$\quad \neg\neg A, \{2, 1\}$	$\neg$E, 3, 4
6	$\quad A, \{2, 1\}$	$\neg\neg$E, 5
7	$B \Rightarrow A, \{1\}$	$\Rightarrow$I, 2, 6
8	$(\neg A \Rightarrow \neg B) \Rightarrow (B \Rightarrow A), \{\}$	$\Rightarrow$I, 1, 7

6. $(A \wedge B) \Rightarrow \neg(A \Rightarrow \neg B)$

1	$A \wedge B, \{1\}$	Hyp
2	$A, \{1\}$	$\wedge$E, 1
3	$B, \{1\}$	$\wedge$E, 1
4	$\quad A \Rightarrow \neg B, \{2\}$	Hyp
5	$\quad A, \{1\}$	Reit, 2
6	$\quad \neg B, \{1, 2\}$	$\Rightarrow$E, 5, 4
7	$(A \Rightarrow \neg B) \rightarrow \neg B, \{1\}$	$\Rightarrow$I, 4, 6
8	$\neg\neg B, \{1\}$	$\neg\neg$I, 3
9	$\neg(A \Rightarrow \neg B), \{1\}$	$\neg$E, 8, 7
10	$(A \wedge B) \Rightarrow \neg(A \Rightarrow \neg B), \{\}$	$\rightarrow$I, 1, 9

7. $((A \Rightarrow B) \land \neg B) \Rightarrow \neg A$

1	$(A \Rightarrow B) \land \neg B, \{1\}$	Hyp
2	$\neg B, \{1\}$	$\land$E, 1
3	$A \Rightarrow B, \{1\}$	$\land$E, 1
4	$\neg A, \{1\}$	$\neg$E, 2, 3
5	$((A \Rightarrow B) \land \neg B) \Rightarrow \neg A, \{\}$	$\Rightarrow$I, 1, 4

8. $((\neg A \lor B) \land \neg B) \Rightarrow \neg A$

1	$(\neg A \lor B) \land \neg B, \{1\}$	Hyp
2	$\neg A \lor B, \{1\}$	$\land$E, 1
3	$\neg B, \{1\}$	$\land$E, 1
4	$\neg A, \{2\}$	Hyp
5	$\neg A, \{2\}$	Rep, 4
6	$B, \{3\}$	Hyp
	$\vdots$	
n	$\neg A, \{3\}$	**Impossível**
$n+1$	$\neg A, \{1\}$	$\lor$E, 2, 4–5, 6–n
$n+2$	$((\neg A \lor B) \land \neg B) \Rightarrow \neg A, \{\}$	$\Rightarrow$I, 1, $n+1$

Como não conseguimos fazer a prova de B para $\neg A$ em que $\neg A$ dependa apenas de B, não conseguimos fazer esta prova.
Assim, não conseguimos provar $((\neg A \lor B) \land \neg B) \Rightarrow \neg A$, apesar de na alínea anterior termos provado $((A \Rightarrow B) \land \neg B) \Rightarrow \neg A$. Isto porque na LIR $A \Rightarrow B$ não é equivalente a $\neg A \lor B$.

9. $(A \Rightarrow \neg A) \Rightarrow \neg A$

1	$A \Rightarrow \neg A, \{1\}$	Hyp
2	$\neg A, \{1\}$	$\neg$I, 1
3	$(A \Rightarrow \neg A) \Rightarrow \neg A, \{\}$	$\Rightarrow$I, 1, 2

10. $((A \vee B) \vee C) \Rightarrow (A \vee (B \vee C))$

1	$(A \vee B) \vee C, \{1\}$	Hyp
2	$A \vee B, \{2\}$	Hyp
3	$A, \{3\}$	Hyp
4	$A \vee (B \vee C), \{3\}$	$\vee$I, 3
5	$B, \{4\}$	Hyp
6	$B \vee C, \{4\}$	$\vee$I, 5
7	$A \vee (B \vee C), \{4\}$	$\vee$I, 6
8	$A \vee (B \vee C), \{2\}$	$\vee$E, 2, 3–4, 5–7
9	$C, \{5\}$	Hyp
10	$B \vee C, \{5\}$	$\vee$I, 9
11	$A \vee (B \vee C), \{5\}$	$\vee$I, 10
12	$A \vee (B \vee C), \{1\}$	$\vee$E, 1, 2–8, 9–11
13	$((A \vee B) \vee C) \Rightarrow (A \vee (B \vee C)), \{\}$	$\Rightarrow$I, 1, 12

11. $(A \wedge (B \vee C)) \Rightarrow ((A \wedge B) \vee (A \wedge C))$

1	$A \wedge (B \vee C), \{1\}$	Hyp
2	$A, \{1\}$	$\wedge$E, 1
3	$B \vee C, \{1\}$	$\wedge$E, 1
4	$B, \{2\}$	Hyp
5	$A, \{1\}$	Reit, 2
	$\vdots$	
n	$A \wedge B, \{??\}$	**Impossível**
$n+1$	$(A \wedge B) \vee (A \wedge C), \{2\}$	$\vee$I, n
$n+2$	$C, \{3\}$	Hyp
$n+3$	$A, \{1\}$	Reit, 2
	$\vdots$	
$n+4$	$A \wedge C, \{??\}$	**Impossível**
$n+5$	$(A \wedge B) \vee (A \wedge C), \{3\}$	$\vee$I, $n+4$
$n+6$	$(A \wedge B) \vee (A \wedge C), \{1\}$	$\vee$E, 1, 4–$n+1$, $n+2$–$n+5$
$n+7$	$(A \wedge (B \vee C)) \Rightarrow ((A \wedge B) \vee (A \wedge C)), \{\}$	$\Rightarrow$I, 1, $n+6$

Não se pode aplicar a regra da introdução da conjunção porque os conjuntos suporte de A e de B e depois de A e de C não são iguais.

12. $(A \vee (B \wedge C)) \Rightarrow ((A \vee B) \wedge (A \vee C))$

1	$A \vee (B \wedge C), \{1\}$	Hyp
2	$\quad A, \{2\}$	Hyp
3	$\quad A \vee B, \{2\}$	$\vee$I, 2
4	$\quad B \wedge C, \{3\}$	Hyp
5	$\quad B, \{3\}$	$\wedge$E, 4
6	$\quad A \vee B, \{3\}$	$\vee$I, 5
7	$A \vee B, \{1\}$	$\vee$E, 1, 2–3, 4–6
8	$\quad A, \{4\}$	Hyp
9	$\quad A \vee C, \{4\}$	$\vee$I, 8
10	$\quad B \wedge C, \{5\}$	Hyp
11	$\quad C, \{3\}$	$\wedge$E, 10
12	$\quad A \vee C, \{3\}$	$\vee$I, 11
13	$A \vee C, \{1\}$	$\vee$E, 1, 8–9, 10–12
14	$(A \vee B) \wedge (A \vee C), \{1\}$	$\wedge$I, 7, 13
15	$(A \vee (B \wedge C)) \Rightarrow ((A \vee B) \wedge (A \vee C)), \{\}$	$\Rightarrow$I, 1, 14

13. $(\neg A \vee B) \Rightarrow \neg(A \wedge \neg B)$

1	$\neg A \vee B, \{1\}$	Hyp
2	$\quad \neg A, \{2\}$	Hyp
3	$\quad\quad A \wedge \neg B, \{3\}$	Hyp
4	$\quad\quad A, \{3\}$	$\wedge$E, 3
5	$\quad (A \wedge \neg B) \Rightarrow A, \{\}$	$\Rightarrow$I, 3, 4
6	$\quad \neg(A \wedge \neg B), \{2\}$	$\neg$E, 2, 5
7	$\quad B, \{4\}$	Hyp
8	$\quad\quad A \wedge \neg B, \{5\}$	Hyp
9	$\quad\quad \neg B, \{5\}$	$\wedge$E, 8
10	$\quad (A \wedge \neg B) \Rightarrow \neg B, \{\}$	$\Rightarrow$I, 8, 9
11	$\quad \neg\neg B, \{4\}$	$\neg\neg$I, 7
12	$\quad \neg(A \wedge \neg B), \{4\}$	$\neg$E, 11, 10
13	$\neg(A \wedge \neg B), \{1\}$	$\vee$E, 1, 2–6, 7–12
14	$(\neg A \vee B) \Rightarrow \neg(A \wedge \neg B), \{\}$	$\Rightarrow$I, 1, 13

14. $((A \Rightarrow B) \wedge (B \Rightarrow \neg A)) \Rightarrow \neg A$

1	$(A \Rightarrow B) \wedge (B \Rightarrow \neg A), \{1\}$	Hyp
2	$A \Rightarrow B, \{1\}$	$\wedge$E, 1
3	$B \Rightarrow \neg A, \{1\}$	$\wedge$E, 1
4	$\quad A, \{2\}$	Hyp
5	$\quad A \Rightarrow B, \{1\}$	Reit, 2
6	$\quad B, \{2, 1\}$	$\Rightarrow$E, 4, 5
7	$\quad B \Rightarrow \neg A, \{1\}$	Reit, 3
8	$\quad \neg A, \{2, 1\}$	$\Rightarrow$E, 6, 7
9	$A \Rightarrow \neg A, \{1\}$	$\Rightarrow$I, 4, 8
10	$\neg A, \{1\}$	$\neg$I, 9
11	$((A \Rightarrow B) \wedge (B \Rightarrow \neg A)) \Rightarrow \neg A, \{\}$	$\Rightarrow$I, 1, 10

15. $((A \Rightarrow B) \wedge (B \Rightarrow C)) \Rightarrow (A \Rightarrow C)$

1	$(A \Rightarrow B) \wedge (B \Rightarrow C), \{1\}$	Hyp
2	$A \Rightarrow B, \{1\}$	$\wedge$E, 1
3	$B \Rightarrow C, \{1\}$	$\wedge$E, 1
4	$\quad A, \{2\}$	Hyp
5	$\quad A \Rightarrow B, \{1\}$	Reit, 2
6	$\quad B, \{2, 1\}$	$\Rightarrow$E, 4, 5
7	$\quad B \Rightarrow C, \{1\}$	Reit, 3
8	$\quad C, \{2, 1\}$	$\Rightarrow$E, 6, 7
9	$A \Rightarrow C, \{1\}$	$\Rightarrow$I, 4, 8
10	$((A \Rightarrow B) \wedge (B \Rightarrow C)) \Rightarrow (A \Rightarrow C), \{\}$	$\Rightarrow$I, 1, 9

16. $(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$

1	$A \Rightarrow (B \Rightarrow C), \{1\}$	Hyp
2	$\quad A \Rightarrow B, \{2\}$	Hyp
3	$\quad\quad A, \{3\}$	Hyp
4	$\quad\quad A \Rightarrow B, \{2\}$	Reit, 2
5	$\quad\quad B, \{3, 2\}$	$\Rightarrow$E, 3, 4
6	$\quad\quad A \Rightarrow (B \Rightarrow C), \{1\}$	Reit, 1
7	$\quad\quad B \Rightarrow C, \{3, 1\}$	$\Rightarrow$E, 3, 6
8	$\quad\quad C, \{3, 2, 1\}$	$\Rightarrow$E, 5, 7
9	$\quad A \Rightarrow C, \{2, 1\}$	$\Rightarrow$I, 3, 8
10	$(A \Rightarrow B) \Rightarrow (A \Rightarrow C), \{1\}$	$\Rightarrow$I, 2, 9
11	$(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)), \{\}$	$\Rightarrow$I, 1, 10

17. $(((A \Rightarrow (A \Rightarrow B)) \wedge A) \Rightarrow B)$

1	$(A \Rightarrow (A \Rightarrow B)) \wedge A, \{1\}$	Hyp
2	$A \Rightarrow (A \Rightarrow B), \{1\}$	$\wedge$E, 1
3	$A, \{1\}$	$\wedge$E, 1
4	$A \Rightarrow B, \{1\}$	$\Rightarrow$E, 3, 2
5	$B, \{1\}$	$\Rightarrow$E, 3, 4
6	$(((A \Rightarrow (A \Rightarrow B)) \wedge A) \Rightarrow B), \{\}$	$\Rightarrow$I, 1, 5

18. $(A \vee B) \Rightarrow (B \vee A)$

1	$A \vee B, \{1\}$	Hyp
2	$\quad A, \{2\}$	Hyp
3	$\quad B \vee A, \{2\}$	$\vee$I, 2
4	$\quad B, \{3\}$	Hyp
5	$\quad B \vee A, \{3\}$	$\vee$I, 4
6	$B \vee A, \{1\}$	$\vee$E, 1, 2–3, 4–5
7	$(A \vee B) \Rightarrow (B \vee A), \{\}$	$\Rightarrow$I, 1, 6

19. $((A \wedge B) \vee A) \Rightarrow A$

1	$(A \wedge B) \vee A, \{1\}$	Hyp
2	$A \wedge B, \{2\}$	Hyp
3	$A, \{2\}$	$\wedge$E, 2
4	$A, \{3\}$	Hyp
5	$A, \{3\}$	Rep, 4
6	$A, \{1\}$	$\vee$E, 1, 2–3, 4–5
7	$((A \wedge B) \vee A) \Rightarrow A, \{\}$	$\Rightarrow$I, 1, 6

20. $((A \vee B) \wedge \neg A) \Rightarrow B$

1	$(A \vee B) \wedge \neg A, \{1\}$	Hyp
2	$A \vee B, \{1\}$	$\wedge$E, 1
3	$\neg A, \{1\}$	$\wedge$E, 1
4	$A, \{2\}$	Hyp
	$\vdots$	
n	$B, \{2\}$	**Impossível**
$n+1$	$B, \{3\}$	Hyp
$n+2$	$B, \{3\}$	Rep
$n+3$	$B, \{1\}$	$\vee$E, 2, 4–n, $n+1$–$n+2$
$n+4$	$((A \vee B) \wedge \neg A) \Rightarrow B, \{\}$	$\Rightarrow$I, 1, $n+3$

Como não conseguimos fazer uma prova para B em que ele dependa exclusivamente de A, não conseguimos fazer esta prova.

21. $\neg(A \wedge \neg B) \Rightarrow (A \Rightarrow B)$

1	$\neg(A \wedge \neg B), \{1\}$	Hyp
2	$A, \{2\}$	Hyp
3	$\neg B, \{3\}$	Hyp
4	$A, \{2\}$	Reit, 2
	$\vdots$	
n	$A \wedge \neg B, \{??\}$	**Impossível**
$n+1$	$\neg B \Rightarrow (A \wedge \neg B), \{2\}$	$\Rightarrow$I, 3, n
$n+2$	$\neg(A \wedge \neg B), \{1\}$	Reit, 1
$n+3$	$\neg\neg B, \{1, 2\}$	$\neg$E, $n+2$, $n+1$
$n+4$	$B, \{1, 2\}$	$\neg\neg$E, $n+3$
$n+5$	$A \Rightarrow B, \{1\}$	$\Rightarrow$I, 2, $n+4$
$n+6$	$\neg(A \wedge \neg B) \Rightarrow (A \Rightarrow B), \{\}$	$\Rightarrow$I, 1, $n+5$

Não se pode aplicar a regra da introdução da conjunção porque os conjuntos suporte de A e de $\neg B$ não são iguais.

22. $(A \vee B) \Rightarrow \neg(\neg A \wedge \neg B)$

1	$A \vee B, \{1\}$	Hyp
2	$A, \{2\}$	Hyp
3	$\neg A \wedge \neg B, \{3\}$	Hyp
4	$\neg A, \{3\}$	$\wedge$E, 3
5	$(\neg A \wedge \neg B) \Rightarrow \neg A, \{\}$	$\Rightarrow$I, 3, 4
6	$\neg\neg A, \{2\}$	$\neg\neg$I, 2
7	$\neg(\neg A \wedge \neg B), \{2\}$	$\neg$E, 6, 5
8	$B, \{4\}$	Hyp
9	$\neg A \wedge \neg B, \{5\}$	Hyp
10	$\neg B, \{5\}$	$\wedge$E, 9
11	$(\neg A \wedge \neg B) \Rightarrow \neg B, \{\}$	$\Rightarrow$I, 9, 10
12	$\neg\neg B, \{4\}$	$\neg\neg$I, 8
13	$\neg(\neg A \wedge \neg B), \{4\}$	$\neg$E, 12, 11
14	$\neg(\neg A \wedge \neg B), \{1\}$	$\vee$E, 1, 2–7, 8–13
15	$(A \vee B) \Rightarrow \neg(\neg A \wedge \neg B), \{\}$	$\Rightarrow$I, 1, 14

23. $\neg(\neg A \wedge \neg B) \Rightarrow (A \vee B)$

1	$\neg(\neg A \wedge \neg B), \{1\}$	Hyp
2	$\neg(A \vee B), \{2\}$	Hyp
3	$A, \{3\}$	Hyp
4	$A \vee B, \{3\}$	$\vee$I, 3
5	$A \Rightarrow (A \vee B), \{\}$	$\Rightarrow$I, 3, 4
6	$\neg A, \{2\}$	$\neg$E, 2, 5
7	$B, \{4\}$	Hyp
8	$A \vee B, \{4\}$	$\vee$I, 7
9	$B \Rightarrow (A \vee B), \{\}$	$\Rightarrow$I, 7, 8
10	$\neg B, \{2\}$	$\neg$E, 2, 9
11	$\neg A \wedge \neg B, \{2\}$	$\wedge$I, 6, 10
12	$\neg(A \vee B) \Rightarrow (\neg A \wedge \neg B), \{\}$	$\Rightarrow$I, 2, 11
13	$\neg\neg(A \vee B), \{1\}$	$\neg$E, 1, 12
14	$A \vee B, \{1\}$	$\neg\neg$E, 13
15	$\neg(\neg A \wedge \neg B) \Rightarrow (A \vee B), \{\}$	$\Rightarrow$I, 1, 14

24. $((\neg A \vee B) \wedge A) \Rightarrow B$

1	$(\neg A \vee B) \wedge A, \{1\}$	Hyp
2	$\neg A \vee B, \{1\}$	$\wedge$E, 1
3	$A, \{1\}$	$\wedge$E, 1
4	$\neg A, \{2\}$	Hyp
	$\vdots$	
n	$B, \{2\}$	**Impossível**
$n+1$	$B, \{3\}$	Hyp
$n+2$	$B, \{3\}$	Rep
$n+3$	$B, \{1\}$	$\vee$E, 2, 4–n, $n+1$–$n+2$
$n+4$	$((\neg A \vee B) \wedge A) \Rightarrow B, \{\}$	$\Rightarrow$I, 1, $n+3$

Como não conseguimos fazer uma prova para B em que ele dependa exclusivamente de $\neg A$, não conseguimos fazer esta prova.

5 Lógica não clássica — Lógica modal

Sumário:

- Representação em lógica Modal.
- Provas no sistema semântico da lógica modal.

Resumo:

A lógica modal extende a lógica clássica com a adição de dois novos operadores, os operadores modais:

- $\Box$ — o operador de necessitação
- $\Diamond$ — o operador de possibilidade
- $\Diamond = \neg\Box\neg$

Estes operadores permitem a representação de conhecimento que não era possível representar na lógica clássica, como por exemplo que algo é necessariamente verdade, que se acredita que seja verdade, que é obrigatório, etc.

A semântica é baseada na noção de mundos possíveis. Cada mundo possível corresponde a um universo e nele avaliam-se as fórmulas como em lógica clássica. A relação de acessibilidade indica quando é que se pode passar de um mundo possível para outro.

Se $\Box\alpha$ é verdade num mundo, então α é verdade em todos os mundos acessíveis a partir desse mundo.

Se $\neg\Box\alpha$ é verdade num mundo, então $\neg\alpha$ é verdade em pelo menos um mundo acessível a partir desse mundo.

Se $\Diamond\alpha$ é verdade num mundo, então α é verdade em pelo menos um mundo acessível a partir desse mundo.

Se $\neg\Diamond\alpha$ é verdade num mundo, então $\neg\alpha$ é verdade em todos os mundos acessíveis a partir desse mundo.

$\Delta \models \alpha$ se, em qualquer mundo w de qualquer modelo $M = (W, R, V)$, temos $M \models_w \alpha$ sempre que $M \models_w \delta$ para qualquer $\delta \in \Delta$.

Uma fórmula α é verdadeira se $\{\} \models \alpha$.

As seguintes fórmulas são sempre verdadeiras:

- $\Diamond\alpha \leftrightarrow \neg\Box\neg\alpha$
- $\neg\Box\alpha \leftrightarrow \Diamond\neg\alpha$
- $\Box(\alpha \wedge \beta) \leftrightarrow \Box\alpha \wedge \Box\beta$
- $\Diamond(\alpha \vee \beta) \leftrightarrow \Diamond\alpha \vee \Diamond\beta$

Axiomas modais mais conhecidos:

- Axioma K: $\Box(\alpha \to \beta) \to (\Box\alpha \to \Box\beta)$ Este axioma é verdadeiro em todas as estruturas de Kripke.
- Axioma T: $\Box\alpha \to \alpha$ Este axioma é verdadeiro em todas as estruturas de Kripke cuja relação de acessibilidade é reflexiva, isto é, que verifica a condição $\forall(w_i \in W)[R(w_i, w_i)]$.
- Axioma D: $\Box\alpha \to \Diamond\alpha$ Este axioma é verdadeiro em todas as estruturas de Kripke cuja relação de acessibilidade é linear, isto é, que verifica a condição $\forall(w_i \in W)[\exists(w_j \in W)[R(w_i, w_j)]]$.
- Axioma 4: $\Box\alpha \to \Box\Box\alpha$ Este axioma é verdadeiro em todas as estruturas de Kripke em que a relação de acessibilidade é transitiva, isto é, que verifica a condição $\forall(w_i, w_j, w_k \in W)[(R(w_i, w_j) \wedge R(w_j, w_k)) \to R(w_i, w_k)]$.
- Axioma 5: $\Diamond\alpha \to \Box\Diamond\alpha$ Este axioma é verdadeiro em todas as estruturas de Kripke cuja relação de acessibilidade é euclideana, isto é, que verifica a condição $\forall(w_i, w_j, w_k \in W)[(R(w_i, w_j) \wedge R(w_i, w_k)) \to R(w_j, w_k)]$.
- Axioma B: $\alpha \to \Box\Diamond\alpha$ Este axioma é verdadeiro em todas as estruturas de Kripke cuja relação de acessibilidade é simétrica, isto é, que verifica a condição $\forall(w_i, w_j \in W)[R(w_i, w_j) \to R(w_j, w_i)]$.

Nome	**Axioma**	**Pp da relação de acessibilidade**
K	$\Box(\alpha \to \beta) \to (\Box\alpha \to \Box\beta)$	—
T	$\Box\alpha \to \alpha$	Reflexiva
D	$\Box\alpha \to \Diamond\alpha$	Linear
4	$\Box\alpha \to \Box\Box\alpha$	Transitiva
5	$\Diamond\alpha \to \Box\Diamond\alpha$	Euclideana
B	$\alpha \to \Box\Diamond\alpha$	Simétrica

Regra de inferência da necessitação: se $\vdash \alpha$ então $\vdash \Box\alpha$. Ou seja, se α é um teorema da lógica clássica, então $\Box\alpha$ é um teorema da lógica modal. Da mesma forma, pela via semântica temos: se $\models \alpha$ então $\models \Box\alpha$.

Lógicas modais para representar conceitos diferentes terão axiomas diferentes, dependendo do significado que for atribuido aos operadores modais.

- Lógica epistémica lida com a noção do conhecimento de um dado agente. $\Box\alpha$ significa que o agente sabe α e $\Diamond\alpha$ significa que o agente não sabe $\neg\alpha$, ou seja, que α é possível. É a lógica KT45, que requer uma relação de acessibilidade reflexiva, transitiva e euclideana.
- Lógica doxástica lida com a noção de crença de um dado agente. $\Box\alpha$ significa que o agente acredita que α é verdade e $\Diamond\alpha$ significa que o agente não acredita em $\neg\alpha$, ou seja, que α é consistente com as crenças do agente. É a lógica K45, que requer uma relação de acessibilidade transitiva e euclideana.

- Lógica deôntica lida com a noção de códigos morais. $\Box\alpha$ significa que α é obrigatório e $\Diamond\alpha$ significa que $\neg\alpha$ não é obrigatório, ou seja, que α é permitido. É a lógica D, que requer uma relação de acessibilidade linear.

- Lógica temporal lida com a noção de tempo. $\Box\alpha$ significa que α vai acontecer sempre no futuro e $\Diamond\alpha$ significa que $\neg\alpha$ não vai acontecer sempre no futuro, ou seja, que α vai acontecer alguma vez no futuro. Uma vez que a relação de acessibilidade significa "depois de", não pode ser reflexiva nem simétrica, logo os axiomas T e B não podem ser considerados.

Regras de inferência do sistema dedutivo

Para além das regras de inferência do sistema dedutivo da lógica de primeira ordem, existem as seguintes regras de inferência.

Resposta:

Resolver exercício 10 do segundo exame de 2004/05.

Exercício 5.1
Considere que os operadores modais $\Box$ e $\Diamond$ têm os seguintes significados:

$\Box$ sempre no futuro

$\Diamond$ alguma vez no futuro

1. Represente as seguintes proposições:
   (a) O Zé há-de ser feliz.
   (b) O Zé será sempre feliz.
   (c) Nunca mais vai haver guerra.
   (d) Existe alguém que nunca vai ser rico.
2. Quais dos axiomas que conhece usaria no seu sistema para raciocinar acerca do tempo? Justifique, utilizando para isso o significado dos axiomas.

   B: $\alpha \to \Box\Diamond\alpha$

   D: $\Box\alpha \to \Diamond\alpha$

   T: $\Box\alpha \to \alpha$

   4: $\Box\alpha \to \Box\Box\alpha$

   5: $\Diamond\alpha \to \Box\Diamond\alpha$

Resposta:

1. (a) $\Diamond Feliz(Ze)$
   (b) $\Box Feliz(Ze)$
   (c) $\Box\neg Ha(Guerra)$ ou $\neg\Diamond Ha(Guerra)$
   (d) $\exists(p)[Pessoa(p) \wedge \Box\neg Rico(p)]$
2. B: se α é verdade agora, então, sempre no futuro, α vai ser verdade alguma vez. Não faz sentido: hoje é o dia 11 de Abril de 2005, o que nunca mais vai ser verdade.

   D: se α vai ser sempre verdade no futuro, então α vai ser verdade alguma vez no futuro. Faz sentido.

   T: se α vai ser sempre verdade no futuro, então α é verdade agora. Não faz sentido: se o Zé acabar o curso amanhã, então ele vai ter o curso acabado sempre no futuro, mas se ainda não o acabou, o curso não está acabado agora.

   4: se α vai ser sempre verdade no futuro, então sempre no futuro α vai ser sempre verdade. Faz sentido (apesar de o português não ser dos melhores).

   5: se α vai ser verdade alguma vez no futuro, então sempre no futuro α vai ser verdade alguma vez no futuro. Não faz sentido: alguma vez no futuro vamos estar no dia 15 de Abril de 2005, mas depois deste dia isso nunca mais vai ser verdade.

Exercício 5.2
Demonstre que o axioma modal 4 ($\Box\alpha \rightarrow \Box\Box\alpha$) é verdadeiro em todas as estruturas em que a relação de acessibilidade é transitiva, isto é, que verifica a condição $\forall(w_i, w_j, w_k \in W)[(R(w_i, w_j) \wedge R(w_j, w_k)) \rightarrow R(w_i, w_k)]$.
Será que este axioma é adequado se interpretarmos o símbolo modal como "eu sei que"? Justifique a sua resposta.

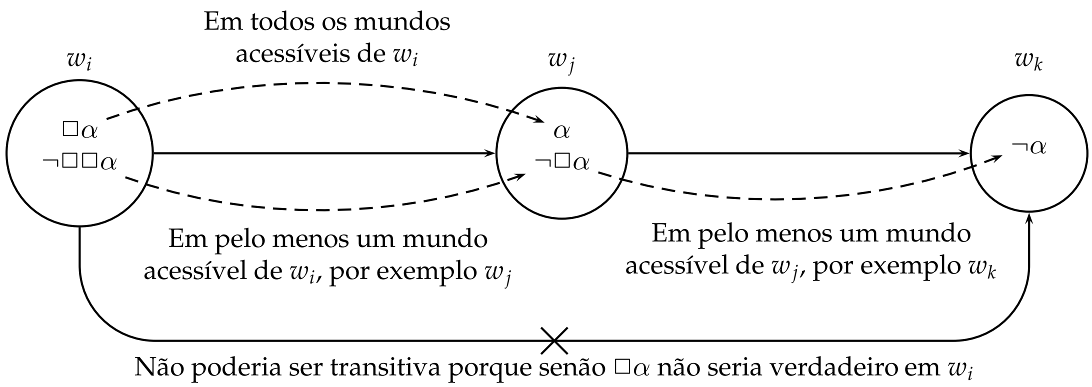
Resposta:

Prova pela contrapositiva (para provar $A \rightarrow B$, provar $\neg B \rightarrow \neg A$):

Para $\Box\alpha \rightarrow \Box\Box\alpha$ ser falso só se

$\Box\alpha$ é verdade e

$\Box\Box\alpha$ é falso.



Se o símbolo modal fosse interpretado como "eu sei que", o axioma 4 significaria: "se eu sei α, então sei que sei α". Seria adequado em sistemas onde se pretendesse usar a capacidade de introspecção.

Exercício 5.3
Demonstre que o axioma modal T ($\Box\alpha \rightarrow \alpha$) é verdadeiro em todas as estruturas em que a relação de acessibilidade é reflexiva, isto é, que verifica a condição $\forall(w_i \in W)[R(w_i, w_i)]$.
Será que este axioma é adequado se interpretarmos o símbolo modal como "eu sei que"? Justifique a sua resposta.

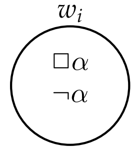
Resposta:

Prova pela contrapositiva (para provar $A \rightarrow B$, provar $\neg B \rightarrow \neg A$):

Para $\Box\alpha \rightarrow \alpha$ ser falso só se

$\Box\alpha$ é verdade e

α é falso.



Se o símbolo modal fosse interpretado como "eu sei que", o axioma T significaria: "se eu sei α, então α é verdade". Seria adequado em sistemas onde o que eu sei fosse sempre verdade no mundo, isto é, onde eu nunca me enganasse...

Exercício 5.4
Considere que o operador modal $\Box$ tem o significado “eu sei que”.

1. Represente as seguintes proposições:
   (a) Eu sei que hoje não chove.
   (b) Eu sei que nos dias de sol não chove.
   (c) Nos dias de chuva eu sei que fico molhado.
2. Com este significado para o operador modal, será que faz sentido introduzir o axioma $\alpha \rightarrow \Box\alpha$?

Resposta:

1. (a) $\Box(\neg Chove(Hoje))$
   (b) $\Box(\forall(d)[(Dia(d) \land EstaSol(d)) \rightarrow \neg Chove(d)])$
   (c) $\forall(d)[(Dia(d) \land EstaChuva(d)) \rightarrow \Box Molhado(Eu)]$
2. O axioma significa: “se α é verdade, então eu sei α”. Fazer este tipo de afirmação só faz sentido se eu for omnisciente . . .

Exercício 5.5
Considere que os operadores modais $\Box$ e $\Diamond$ têm os seguintes significados:

$\Box$ é obrigatório

$\Diamond$ é permitido

1. Represente as seguintes frases:
   (a) Os cidadãos colectados têm que fazer a declaração de rendimentos.
   (b) É proibido matar pessoas.
   (c) As mulheres podem ir à tropa.
   (d) Apenas as pessoas com carta podem conduzir.
2. Quais dos axiomas que conhece usaria no seu sistema para raciocinar acerca de leis? Justifique.

B:	$\alpha \rightarrow \Box\Diamond\alpha$
D:	$\Box\alpha \rightarrow \Diamond\alpha$
T:	$\Box\alpha \rightarrow \alpha$
4:	$\Box\alpha \rightarrow \Box\Box\alpha$
5:	$\Diamond\alpha \rightarrow \Box\Diamond\alpha$

Resposta:

1. (a) $\forall(x)[(Cidadao(x) \wedge Colectado(x)) \rightarrow \Box FazerDeclaracaoRendimentos(x)]$
   (b) $\forall(x)[(Pessoa(x) \wedge Pessoa(y)) \rightarrow \neg\Diamond Matar(x,y)]$
   (c) $\forall(x)[Mulher(x) \rightarrow \Diamond Ir(x, Tropa)]$
   (d) $\forall(x)[(Pessoa(x) \wedge Tem(x, Carta)) \rightarrow \Diamond Conduzir(x)]$
   $\forall(x)[\Diamond Conduzir(x) \rightarrow (Pessoa(x) \wedge Tem(x, Carta))]$
2. B: Se α acontece, então é obrigatório ser permitido α. Não faz sentido: rouba-se, mas é proibido roubar.

   D: Se α é obrigatório, então α é permitido. Faz sentido.

   T: Se α é obrigatório, então α acontece. Depende: é obrigatório parar nos sinais vermelhos, mas nem toda a gente pára. No entanto, gostaríamos que fosse satisfeito. Depende do mundo que estivéssemos a modelar.

   4: Se α é obrigatório, então é obrigatório que α seja obrigatório. Apesar do português estranho, faz sentido que se verifique.

   5: Se α é permitido, então é obrigatório que α seja permitido. Faz sentido.

Exercício 5.6
Prove que o axioma modal B ($\alpha \rightarrow \Box\Diamond\alpha$) é verdadeiro em todas as estruturas cuja relação de acessibilidade é simétrica, isto é, que verifica a condição $\forall(w_i, w_j \in W)[R(w_i, w_j) \rightarrow R(w_j, w_i)]$.

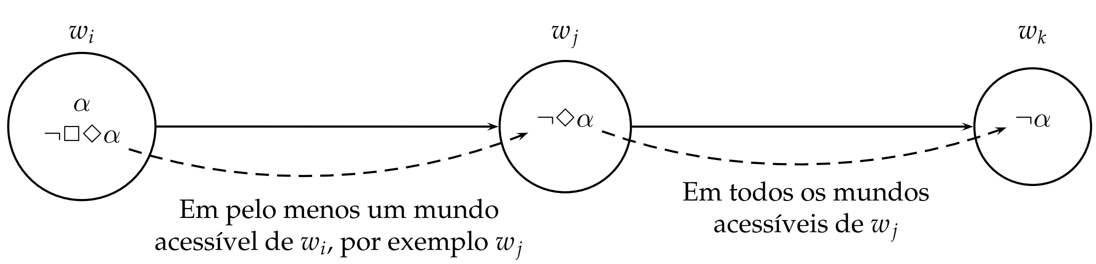
Resposta:

Vamos supor, por redução ao absurdo, que o axioma B não se verifica e verificar se a relação de acessibilidade pode ser simétrica.

Para $\alpha \rightarrow \Box\Diamond\alpha$ ser falso só se

α é verdade e

$\neg\Box\Diamond\alpha$ é falso.



Como α tem que ser verdadeiro em todos os mundos acessíveis de w_j e α não é verdadeiro em w_i, w_j não pode "ver" w_i, logo a relação de acessibilidade não pode ser simétrica.

Exercício 5.7
Prove que o axioma modal D ($\Box\alpha \rightarrow \Diamond\alpha$) é verdadeiro em todas as estruturas cuja relação de acessibilidade é linear, isto é, que verifica a condição $\forall(w_i \in W)[\exists(w_j \in W)[R(w_i, w_j)]]$.

Resposta:

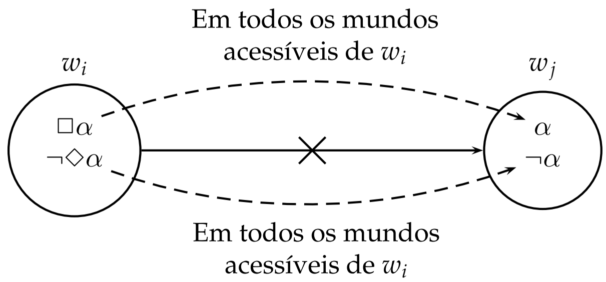
Pretendemos provar que: Ax. D Verd. $\leftrightarrow$ Relação. Acess. Linear, isto é, que qualquer mundo onde o axioma D seja verdadeiro "vê" pelo menos um outro mundo.

Vamos fazer uma prova pela contrapositiva, isto é, que Ax. D Falso $\leftrightarrow$ Relação. Acess. Não Linear:

Para $\Box\alpha \rightarrow \Diamond\alpha$ ser falso só se

$\Box\alpha$ é verdade e

$\Diamond\alpha$ é falso.



Uma vez que qualquer mundo acessível de w_1 é contraditório, w_1 não pode "ver" nenhum mundo, logo, a relação de acessibilidade não pode ser linear.

Exercício 5.8
Prove que o axioma modal 5 ($\Diamond\alpha \rightarrow \Box\Diamond\alpha$) é verdadeiro em todas as estruturas cuja relação de acessibilidade é euclideana, isto é, que verifica a condição $\forall(w_i, w_j, w_k \in W)[(R(w_i, w_j) \wedge R(w_i, w_k)) \rightarrow R(w_j, w_k)]$.

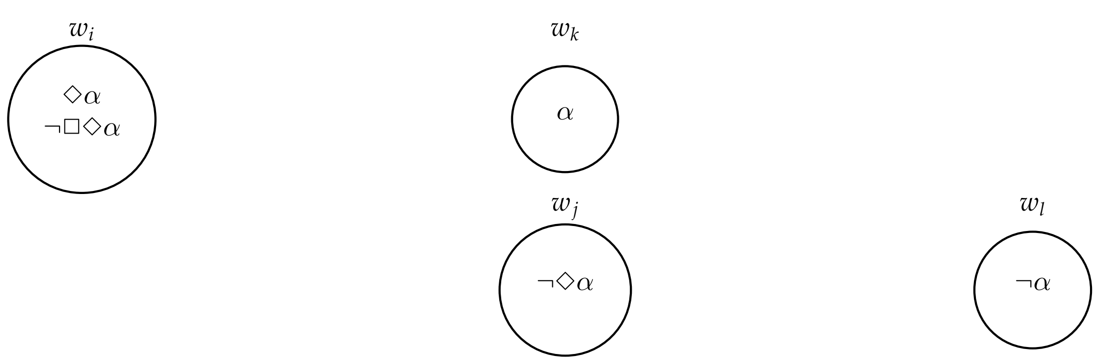
Resposta:

Vamos supor, por redução ao absurdo, que o axioma 5 não se verifica e verificar se a relação de acessibilidade pode ser euclideana.

Para $\Diamond\alpha \rightarrow \Box\Diamond\alpha$ ser falso só se

$\Diamond\alpha$ é verdade e

$\Box\Diamond\alpha$ é falso.



Para a relação de acessibilidade ser euclideana, w_k tem que ser acessível a partir de w_j, e neste caso teria que satisfazer $\neg\alpha$. Como w_k já satisfaz α, seria um mundo contraditório. Logo, a relação de acessibilidade não poderia ser euclideana.

Exercício 5.9

Prove que se $\vdash (\alpha \to \beta)$ então $\vdash (\Box\alpha \to \Box\beta)$, no sistema sintáctico da lógica modal normal (só com axioma K, axiomas da LPO, MP, necessitação).

Resposta:

Pretendemos provar: $\vdash (\alpha \to \beta) \to (\Box\alpha \to \Box\beta)$

Se partirmos da hipótese $\vdash (\alpha \to \beta)$, então, pela regra da necessitação, $\vdash \Box(\alpha \to \beta)$.

Dado o axioma K, $(\vdash \Box(\alpha \to \beta) \to (\Box\alpha \to \Box\beta))$ existente em todas as LM Normais, e como temos que $\vdash \Box(\alpha \to \beta)$, por Modus Ponnens ficamos com $\vdash \Box\alpha \to \Box\beta$.

Como inicialmente tinhamos partido de $\{\alpha \to \beta\}$, pelo teorema da dedução da LPO, temos $\{\} \vdash (\alpha \to \beta) \to (\Box\alpha \to \Box\beta)$

Teorema da dedução: se Δ for um conjunto de fbfs e α e β forem fbfs e se $(\Delta \cup \{\alpha\}) \vdash \beta$ entao $\Delta \vdash (\alpha \to \beta)$

Exercício 5.10

Prove a regra de inferência da necessitação no sistema semântico da lógica modal.

Resposta:

Pretendemos provar que, se $\models \alpha$ (α é um teorema da LPO), entao $\models \Box\alpha$ ($\Box\alpha$ é um teorema da lógica modal).

Se α é um teorema da LPO, então, uma vez que as lógicas modais *estendem* a LPO, α também é um teorema da lógica modal, o que significa que é teorema em todos os mundos existentes.

Uma vez que todos os mundos contêm todos os teoremas da LPO (incluindo α), α vai ser sempre visível em todos os mundos. Logo, temos $\Box\alpha$ em todos os mundos, ou seja, $\Box\alpha$ é um teorema da lógica modal.

Exercício 5.11

Prove, pela via sintáctica, no sistema T o seguinte teorema: $p \to \Diamond p$. Sugestão: use o axioma $(\alpha \to \neg\beta) \to (\beta \to \neg\alpha)$. Axioma T $= \Box\alpha \to \alpha$.

Resposta:

1.	$\Box\neg p \to \neg p$	Axioma T
2.	$(\Box\neg p \to \neg p) \to (p \to \neg\Box\neg p)$	Inst. do axioma: $\alpha = \Box\neg p$ e $\beta = p$
3.	$p \to \neg\Box\neg p$	$MP(1,2)$
4.	$p \to \Diamond p$	Abrev. $\Diamond$

Exercício 5.12
Prove, pela via semântica, no sistema T o seguinte teorema: $p \to \Diamond p$.
Axioma T = $\Box\alpha \to \alpha$. (relação de acessibilidade reflexiva)

Resposta:

Podemos fazer a prova de várias formas.

Forma 1. Fazer uma prova por redução ao absurdo, lembrando que os sistemas que satisfazem o axioma T têm um relação de acessibilidade reflexiva.

Para esta prova, queremos ver se é possível existir um modelo em que a relação de acessibilidade entre os mundos é reflexiva e que não satisfaz o teorema $p \to \Diamond p$.

Para isso, supor que existe um mundo, w_i, que não satisfaz o teorema. Assim, w_i satisfaz p e $\neg\Diamond p$. Usando a abreviatura ($\Diamond = \neg\Box\neg$), $\neg\Diamond p = \Box\neg p$, o que significa que todos os mundos acessíveis de w_i têm que satisfazer $\neg p$.

Uma vez que a relação de acessibilidade entre os mundos é reflexiva, w_i é acessível a partir de si próprio, logo, tem que satisfazer $\neg p$.

Como w_i também satisfaz p, temos um mundo contraditório, logo, não pode existir nenhum modelo em que a relação de acessibilidade é reflexiva e alguns dos mundos não satisfaça o teorema.

Forma 2. Fazer uma prova por redução ao absurdo, ignorando o facto que os sistemas que satisfazem o axioma T têm um relação de acessibilidade reflexiva: Será que pode existir algum mundo que satisfaça o axioma T e não satisfaça o teorema?

Vamos supor que existe e que é o w_i.

w_i satisfaz:	$\Box\alpha \to \alpha$	Axioma T
	p e $\neg\Diamond p$	Para não satisfazer o teorema
	$\Box\neg p$	Pela abreviatura, $\Diamond p = \neg\Box\neg p$
	$\neg p$	Pela aplicação do axioma T

O mundo w_i é inconsistente (porque satisfaz simultaneamente p e $\neg p$), logo, não pode existir. Por isso, todos os mundos que satisfizerem o axioma T também têm que satisfazer o teorema $p \to \Diamond p$.

(Existem mais possibilidades de solução na 6ª aula de 94-95)

Exercício 5.13
Prove, pela via **semântica**, na lógica modal normal, os seguintes teoremas:

1. $\{\Box B\} \vdash \Box(A \to B)$
2. $\{\Box\neg A\} \vdash \Box(A \to B)$
3. $\{A \to \Box B, \Box(B \to C)\} \vdash A \to \Box C$
4. $\Diamond(A \wedge B) \to (\Diamond A \wedge \Diamond B)$
5. $\{\Box A\} \vdash \Box(A \vee B)$

Resposta:

1. Por redução ao absurdo, suponhamos que existe um mundo $\underline{w_i}$ que satisfaz $\Box$B e não satisfaz $\Box$(A$\to$B). Então, deve existir um outro mundo qualquer, $\underline{w_j}$, acessível a partir de $\underline{w_i}$, em que A$\to$B seja falso, ou seja, que satisfaça A e ¬B. Mas, se $\Box$B é satisfeito em $\underline{w_i}$, então B deve ser satisfeito em todos os mundos visíveis de $\underline{w_i}$, como por exemplo, $\underline{w_j}$. Assim, $\underline{w_j}$ deveria satisfazer simultaneamente B e ¬B, o que não é possível. Logo, $\Box$(A$\to$B) tem que se verificar em $\underline{w_i}$.

2. Por redução ao absurdo, suponhamos que existe um mundo $\underline{w_i}$ que satisfaz $\Box\neg$A e não satisfaz $\Box$(A$\rightarrow$B). Então, deve existir um outro mundo qualquer, $\underline{w_j}$, acessível a partir de $\underline{w_i}$, em que A$\rightarrow$B seja falso, ou seja, que satisfaça A e $\neg$B. Mas, se $\Box\neg$A é satisfeito em $\underline{w_i}$, então $\neg$A deve ser satisfeito em todos os mundos visíveis de $\underline{w_i}$, como por exemplo, $\underline{w_j}$. Assim, $\underline{w_j}$ deveria satisfazer simultaneamente A e $\neg$A, o que não é possível. Logo, $\Box$(A$\rightarrow$B) tem que se verificar em $\underline{w_i}$.

3. Por redução ao absurdo, suponhamos que existe um mundo $\underline{w_i}$ que satisfaz A$\rightarrow\Box$B e $\Box$(B$\rightarrow$C), e não satisfaz A$\rightarrow\Box$C. Logo, $\underline{w_i}$ satisfaz A e $\neg\Box$C, ou seja, por abreviatura, satisfaz A e $\Diamond\neg$C. Isso implica que existe um mundo acessível a partir de $\underline{w_i}$, $\underline{w_j}$, em que $\neg$C é satisfeito. Por outro lado, como $\underline{w_i}$ satisfaz A e A$\rightarrow\Box$B, podemos concluir que satisfaz $\Box$B, o que implica que em todos os mundos acessíveis a partir de $\underline{w_i}$ (como por exemplo, $\underline{w_j}$), B é satisfeito. Analogamente, se $\underline{w_i}$ satisfaz $\Box$(B$\rightarrow$C), então em todos os mundos acessíveis a partir de $\underline{w_i}$ (como por exemplo $\underline{w_j}$) (B$\rightarrow$C) é satisfeito. Logo, como em $\underline{w_j}$ B e (B$\rightarrow$C) são satisfeitos, concluimos que C também o é nesse mundo. Isto faria com que $\underline{w_j}$ satisfizesse C e $\neg$C, o que não pode acontecer. Logo, A$\rightarrow\Box$C deve-se verificar em $\underline{w_i}$.

4. Por redução ao absurdo, suponhamos que existe um mundo $\underline{w_i}$ que não satisfaz $\Diamond$(A $\wedge$ B)$\rightarrow$($\Diamond$A $\wedge$ $\Diamond$B), ou seja, que satisfaz $\Diamond$(A $\wedge$ B) e não satisfaz ($\Diamond$A $\wedge$ $\Diamond$B). Se satisfaz $\Diamond$(A $\wedge$ B), deve existir um outro mundo qualquer, $\underline{w_j}$, acessível a partir de $\underline{w_i}$, em que A $\wedge$ B seja satisfeito, ou seja, em que A e B são ambos satisfeitos. Por outro lado, se $\underline{w_i}$ não satisfaz ($\Diamond$A $\wedge$ $\Diamond$B), isso implica que satisfaz $\neg\Diamond$A ou $\neg\Diamond$B (ou ambos). Basta-nos assumir que satisfaz $\neg\Diamond$A. Isso implica que satisfaz (por abreviatura) $\Box\neg$A. Assim, todos os mundos acessíveis a partir de $\underline{w_i}$ (por exemplo $\underline{w_j}$), devem satisfazer $\neg$A. Isto faria com que em $\underline{w_j}$ A e $\neg$A fossem satisfeitos simultaneamente no mesmo mundo, o que é impossível. Logo, $\Diamond$(A $\wedge$ B)$\rightarrow$($\Diamond$A $\wedge$ $\Diamond$B) deve ser satisfeito em $\underline{w_i}$.

6 Lógica não monótona — LOR, Representação

Sumário:

- Lógica da omissão: representação e sistema sintáctico.

Resumo:

Os formalismos não-monótonos foram introduzidos para permitir raciocínio baseado em conhecimento incompleto.

Chamam-se não-monótonos porque com a adição de novo conhecimento, o nosso conjunto de crenças pode diminuir, isto é, podemos deixar de acreditar em algo em que acreditávamos anteriormente.

- Regra de omissão: $\frac{A(x) : B_1(x), \ldots, B_m(x)}{C(x)}$.
  Em que $A(x)$ é a pré-condição, $B_1(x), \ldots, B_m(x)$ é a justificação e $C(x)$ é a conclusão da regra.
  Esta regra significa que, se acreditarmos em $A(x)$ e for consistente assumir cada um dos $B_1(x), \ldots, B_m(x)$, então podemos concluir $C(x)$.
  Para podermos concluir $C(x)$, é preciso provar que não se consegue acreditar em $\neg B_1(x), \ldots, \neg B_m(x)$, nem se vai passar a acreditar a partir de nenhuma consequência desta regra. Este problema é semi-decidível. É possível que $C(x) \rightarrow \neg B_i(x)$, por isso o processo de construção de extensões não pode ser construtivo.

- Regra de omissão normal: $\frac{A(x) : B(x)}{B(x)}$.
  Numa regra de omissão normal, a (única) justificação é equivalente à conclusão.
  As teorias que só têm regras de omissão normais são semi-monótonas, isto é, se o conjunto de regras de omissão da teoria aumentar, para cada extensão da teoria original, vai existir uma extensão da nova teoria que a contém. Cada teoria de omissão normal tem sempre pelo menos uma extensão e existe um mecanismo de decisão mecânico que permite determinar se uma determinada fbf pertence a alguma extensão da teoria.

- Regra de omissão semi-normal: $\frac{A(x) : B(x) \wedge \neg C(x)}{B(x)}$.
  Numa regra de omissão semi-normal, a (única) justificação implica a conclusão.
  As teorias de omissão semi-normais não têm a garantia de extensão a não têm a propriedade da semi-monotonicidade.

- A lógica da omissão de reiter tem alguns problemas. Por exemplo: dadas as regras $\frac{A : B}{B}$ e $\frac{\neg A : B}{B}$, se não soubermos nada acerca de A, não podemos concluir nada acerca de B.

- As extensões de uma teoria de omissão (R, Δ) são os conjuntos de conclusões que podem ser obtidas de Δ, usando não só a lógica de primeira ordem, mas também as regras de omissão em R. Uma teoria de omissão pode definir zero ou mais extensões.

Exercício 6.1
Considere a representação que fez para o exercício dos mamíferos em Lógica De Primeira Ordem.

Diga que alterações é que deveria fazer a essa representação para a passar para a Lógica de Omissão do Reiter.

Resposta:

Segue-se a representação que foi feita para este exercício em LPO.

Existem aproximadamente 4500 espécies de mamíferos.
$AproxIgual(cardinDe(EspMam), 4500)$

$\forall(x, y)[(Pertence(x, EspMam) \wedge TemEsp(y, x)) \rightarrow Mam(y)]$

Todos os mamíferos respiram oxigénio do ar.
$\forall(x)[Mam(x) \rightarrow Respira(x, OxigenioAr)]$

Os mamíferos cuidam das suas crias enquanto bebés.
$\forall(x, y)[(Mam(x) \wedge Cria(y, x)) \rightarrow (Mam(y) \wedge (Bebe(y) \rightarrow Cuida(x, y)))]$

> Nota: se quisermos falar acerca das espécies dos animais, por exemplo, para dizer que as crias de um animal têm a mesma espécie que ele, devemos usar $TemEsp(x, Mam)$ em vez de $Mam(x)$. Desta forma, estaríamos a "coisificar" as espécies e poderíamos ter:
> $\forall(x, y, z)[(TemEsp(x, y) \wedge Cria(z, y)) \rightarrow TemEsp(z, y)]$
> Esta abordagem também permitiria dizer que os animais apenas têm uma espécie, coisa que com a representação mais simples não é possível:
> $\forall(x, y, z)[(TemEsp(x, y) \wedge \neg Igual(y, z)) \rightarrow \neg TemEsp(x, z)]$

Os mamíferos (que são fêmeas) alimentam as crias com leite materno.
$\forall(x, y)[(Mam(x) \wedge Femea(x) \wedge Cria(y, x)) \rightarrow Alimenta(x, y, LeiteMaterno)]$

As fêmeas dos mamíferos (que não sejam monotremas) desenvolvem as crias dentro do ventre.
$\forall(x, y)[(Mam(x) \wedge Femea(x) \wedge Cria(y, x) \wedge \neg Monotrema(x)) \rightarrow DesenvolveDentro(x, y, ventreDe(x))]$

> Nota: Esta regra não pode ser aplicada a outros mamíferos, a não ser que seja explicitamente dito que não são monotremas.

As gestações das fêmeas dos mamíferos podem ter entre 1 e 27 crias.
$\forall(x, y)[(Mam(x) \wedge Femea(x) \wedge Gestacao(y, x)) \rightarrow Entre(numCriasDe(y), 1, 27)]$

Os monotremas são mamíferos.
$\forall(x)[Monotrema(x) \rightarrow Mam(x)]$

Os ornitorrincos são monotremas.
$\forall(x)[Ornitorrinco(x) \rightarrow Monotrema(x)]$

As equidnas são monotremas.
$\forall(x)[Equidna(x) \rightarrow Monotrema(x)]$

As fêmeas dos monotremas põem ovos e incubam-nos.
$\forall(x)[(Monotrema(x) \wedge Femea(x)) \rightarrow (Poe(x, ovosDe(x)) \wedge Incuba(x, ovosDe(x)))]$

As fêmeas dos monotremas não desenvolvem as crias dentro do ventre.
$\forall(x, y)[(Monotrema(x) \wedge Femea(x) \wedge Cria(y, x)) \rightarrow \neg DesenvolveDentro(x, y, ventreDe(x))]$

Os machos dos ornitorrincos têm espigões venenosos nas patas traseiras.
$\forall(x)[(Ornitorrinco(x) \wedge \neg Femea(x)) \rightarrow (Tem(x, espigaoDe(x)) \wedge Venenoso(espigaoDe(x))$
$\wedge\ Localizacao(espigaoDe(x), pataTraseiraDe(x)))]$

Os humanos são mamíferos.
$\forall(x)[Humano(x) \rightarrow Mam(x)]$

Os humanos não são monotremas.
$\forall(x)[Humano(x) \rightarrow \neg Monotrema(x)]$

> Nota: Esta regra é necessária para podermos aplicar aos humanos as regras que têm como excepção os monotremas.

O IMC dos humanos é igual ao peso a dividir pela altura ao quadrado.
$\forall(x)[Humano(x) \rightarrow Igual(imcDe(x), divisaoDe(pesoDe(x), quadradoDe(alturaDe(x))))]$

Os humanos com IMC menor que 18.5 têm peso inferior ao normal.
$\forall(x)[(Humano(x) \wedge Menor(imcDe(x), 18.5)) \rightarrow TipoPeso(pesoDe(x), PesoInferiorNormal)]$

Os humanos com IMC entre 18.5 e 25 têm peso normal.
$\forall(x)[(Humano(x) \wedge Entre(imcDe(x), 18.5, 25)) \rightarrow TipoPeso(pesoDe(x), PesoNormal)]$

Os humanos com IMC entre 25 e 29.9 têm peso excessivo.
$\forall(x)[(Humano(x) \wedge Entre(imcDe(x), 25, 29.9)) \rightarrow TipoPeso(pesoDe(x), PesoExcessivo)]$

Os humanos com IMC maior que 30 sofrem de obesidade.
$\forall(x)[(Humano(x) \wedge Maior(imcDe(x), 30)) \rightarrow TipoPeso(pesoDe(x), Obesidade)]$

O Luis é humano, pesa 90 kilos e mede 1.90 metros.
$Humano(Luis)$

$Igual(pesoDe(Luis), 90)$

$Igual(alturaDe(Luis), 1.9)$

O Zé é cria do Luis.
$Cria(Ze, Luis)$

A OF é um ornitorrinco fêmea.
$Ornitorrinco(OF) \wedge Femea(OF)$

O OM é um ornitorrinco macho.
$Ornitorrinco(OM) \wedge \neg Femea(OM)$

A OC é uma cria de OF.
$Cria(OC, OF)$

O Flip ou é um ornitorrinco ou é uma equidna.
$((Ornitorrinco(Flip) \wedge \neg Equidna(Flip)) \vee$
$(\neg Ornitorrinco(Flip) \wedge Equidna(Flip))$

Fim da representação em LPO

Relativamente a esta representação, devemos alterar a regra que diz que as fêmeas dos mamíferos que não sejam monotremas desenvolvem as crias dentro do ventre (tipo de reprodução é vivíparo), de modo a dizer que *normalmente* as fêmeas dos mamíferos desenvolvem as crias dentro do ventre. Esta regra tem a excepção dos monotremas, que são ovíparos, e cuja regra correspondente já tinha sido escrita na representação em LPO.

$$\frac{Mamifero(x) \wedge Femea(x) \wedge Cria(y,x) : \neg Monotrema(x)}{DesenvolveDentro(x,y,ventre(x))}$$

Mas com esta regra, se dissermos que a Maria é um mamífero fêmea, que o Zé é sua cria e que a Maria não desenvolve o Zé dentro do ventre, continuamos a poder aplicar esta regra de omissão à Maria e ao Zé, o que vai fazer com que o conjunto de crenças resultante seja contraditório, ou seja, não vamos ter nenhuma extensão. Assim, deveríamos ter usado uma regra de omissão semi-normal, escrevendo:

$$\frac{Mamifero(x) \wedge Femea(x) \wedge Cria(y,x) : \neg Monotrema(x) \wedge DesenvolveDentro(x,y,ventre(x))}{DesenvolveDentro(x,y,ventre(x))}$$

Neste caso, faz sentido dizer que os monotremas não desenvolvem as crias dentro do ventre. Isto já devíamos ter dito na LPO, mas na altura a representação que fizémos ficou incompleta:

$$\forall(x,y)[(Monotrema(x) \wedge Cria(y,x)) \rightarrow \neg DesenvolveDentro(x,y,ventre(x))]$$

Para além disso, sabemos que os mamíferos normalmente não põem ovos, mas as fêmeas dos monotremas põem. Em LPO não tínhamos maneira de representar o *normalmente*, e por isso não o fizémos, mas agora podemos ter:

$$\frac{Mamifero(x) : \neg Poe(x,ovos(x))}{\neg Poe(x,ovos(x))}$$

Exercício 6.2

Diga, justificando, se as seguintes regras de omissão fazem ou não sentido, do ponto de vista da Representação do Conhecimento.

1. $\frac{A(x) : B(x)}{B(x)}$
2. $\frac{A(x) : B(x) \wedge C(x)}{B(x)}$
3. $\frac{A(x) : B(x)}{C(x)}$
4. $\frac{A(x) : B(x)}{A(x)}$
5. $\frac{: \neg A(x)}{A(x)}$
6. $\frac{: A(x)}{A(x)}$
7. $\frac{A(x) :}{B(x)}$

Resposta:

O objectivo deste exercício é analizar a forma das regras de omissão que escrevermos quando estivermos a representar conhecimento, para nos dar a capacidade de vermos se elas fazem ou não sentido. Do ponto de vista da LOR, todas elas podem ser usadas, à excepção da última (como a justificação é vazia, não está sintacticamente correcta).

1. $\frac{A(x) : B(x)}{B(x)}$ — Esta é uma regra de omissão normal. Estas são as regras de omissão mais usadas, para tirar partido das propriedades das teorias de omissão normais, como por exemplo a existência de pelo menos uma extensão.

2. $\frac{A(x):B(x)\wedge C(x)}{B(x)}$ — Esta é uma regra de omissão semi-normal. Estas regras de omissão também são muito usadas, apesar de as teorias de omissão que usam regras com esta forma já não terem garantia de extensão.

3. $\frac{A(x):B(x)}{C(x)}$ — As regras com esta forma não são tão usadas como as regras com as duas formas anteriores, mas podem perfeitamente fazer sentido, do ponto de vista da Representação do Conhecimento.

4. $\frac{A(x):B(x)}{A(x)}$ — Não faz sentido: se temos que saber $A(x)$ para podermos aplicar a regra, não adianta nada aplicá-la para voltarmos a concluir $A(x)$.

5. $\frac{:\neg A(x)}{A(x)}$ — Não faz sentido: se for consistente assumir $\neg A(x)$, não faz sentido concluir $A(x)$, até porque $\neg A(x)$ pode já estar na base de conhecimento.

6. $\frac{:A(x)}{A(x)}$ — Faz sentido. É uma regra de omissão normal, mas em que a pré-condição é vazia.

7. $\frac{A(x):}{B(x)}$ — Não faz sentido: uma vez que a justificação da regra é vazia, deveríamos ter usado uma regra universal $\forall(x)[A(x) \rightarrow B(x)]$. Esta regra é a única que não está sintacticamente correcta do ponto de vista da LOR.

Exercício 6.3

Considere as seguintes frases:

- As pessoas simpáticas têm pelo menos uma outra pessoa que é sua amiga.
- Em geral, as pessoas conhecem a sua mãe.
- As pessoas que são honestas normalmente não mentem.
- Tipicamente, as pessoas simpáticas são bem-dispostas, a não ser que estejam zangadas.
- Nem os solteiros nem os padres são casados.
- O Rui é um padre casado.

1. Represente-as, usando uma teoria da Lógica de Omissão de Reiter.

2. Essa teoria tem alguma extensão? Porquê?

3. Que alterações é que faria no conhecimento que foi representado de modo a que a teoria passasse a ter pelo uma extensão não contraditória?

Resposta:

1. Teoria de omissão = (R, Δ)

   $R = \{ \frac{Pessoa(x):Conhece(x,mae(x))}{Conhece(x,mae(x))},$
   $\frac{Pessoa(x)\wedge Honesta(x):\neg Mente(x)}{\neg Mente(x)},$
   $\frac{Pessoa(x)\wedge Simpatico(x):BemDisposto(x)\wedge\neg Zangado(x)}{BemDisposto(x)}\}$

   Nota: Nesta última regra de omissão, temos $BemDisposto(x)$ na justificação porque pode haver pessoas que não estejam zangadas e que mesmo assim não estejam bem dispostas. Por exemplo, se alguém estiver doente não está bem disposto, embora não esteja necessariamente zangado.

   $\Delta = \{\forall(x)[(Pessoa(x) \wedge Simpatico(x)) \rightarrow \exists(y)[Pessoa(y) \wedge Amigo(y,x)]],$
   $\forall(x)[(Solteiro(x) \vee Padre(x)) \rightarrow \neg Casado(x)]$, OU
   $\neg\exists(x)[(Solteiro(x) \vee Padre(x)) \wedge Casado(x)],$
   $Padre(Rui) \wedge Casado(Rui)\ \}$

Nota2: Representámos correctamente o que nos foi pedido e ficámos com uma teoria de omissão inconsistente. Isto não é problema da nossa representação, que está correcta, mas sim do conhecimento que nos pediram para representar. Podemos representar teorias de omissão inconsistentes.

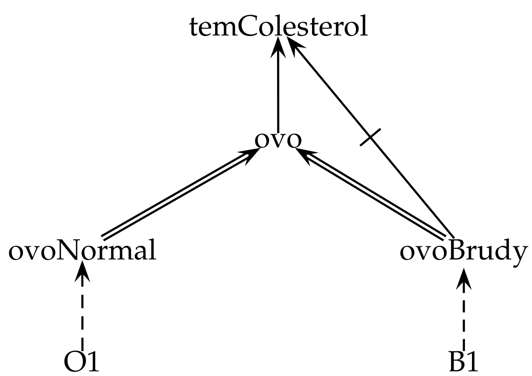
2. Sim, esta teoria tem uma extensão inconsistente, porque o conjunto $th(\Delta)$ é inconsistente. Estes são os únicos casos em que uma teoria pode ter uma extensão que é inconsistente, e na extensão estão todas as fbfs da linguagem da LPO.

3. Deixava de acreditar que não há padres casados e isso passava a ser uma regra de omissão, que pode ter excepções:
   $\frac{Padre(x):\neg Casado(x)}{\neg Casado(x)}$
   Também podia deixar de acreditar que o Rui é padre ou que ele é casado, embora isto fizesse menos sentido.

Exercício 6.4

Considere a seguinte hierarquia, em que:

$a \Longrightarrow b$ significa que todos os as são bs
$a \not\Longrightarrow b$ significa que nenhum a é um b
$a \longrightarrow b$ significa que normalmente os as são bs
$a \not\longrightarrow b$ significa que normalmente os as não são bs
$A \dashrightarrow b$ significa que este A é um b
$A \not\dashrightarrow b$ significa que este A não é um b



1. Represente-a usando uma teoria de omissão da lógica de omissão do Reiter.
2. Diga quais devem ser, intuitivamente, as extensões dessa teoria e o que consegue concluir acerca de cada uma das instâncias.
3. Algumas destas conclusões podem ser invalidadas por nova informação? Porquê?
4. Que alterações teria que fazer à hierarquia anterior para acrescentar o $C1$, que é um ovo de chocolate? Quais as propostas de extensão que fazia sentido testar neste caso?
5. E se acrescentasse outro ovo Brudy, o $B2$?

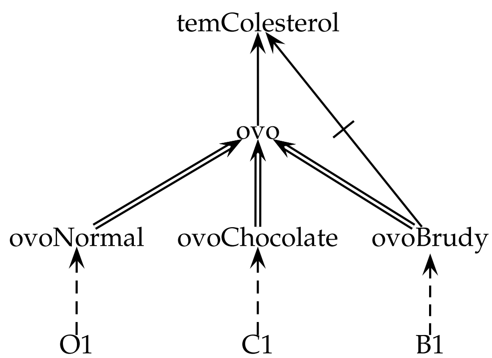
Resposta:

1. Teoria de omissão = (R, Δ)
   $R = \{ \frac{Ovo(x):TemColesterol(x)}{TemColesterol(x)},$
   $\frac{OvoBrudy(x):\neg TemColesterol(x)}{\neg TemColesterol(x)} \}$

$$\Delta = \{\forall(x)[OvoNormal(x) \rightarrow Ovo(x)],$$
$$\forall(x)[OvoBrudy(x) \rightarrow Ovo(x)],$$
$$OvoNormal(O1),$$
$$OvoBrudy(B1)\,\}$$

2. Temos dois conjuntos de conclusões possíveis, correspondentes aos "caminhos" possíveis para cada uma das instâncias. Num deles, tanto o O1 como o B1 têm colesterol; no outro, o O1 tem colesterol mas o B1 não tem. A LOR não nos permite escolher entre eles.

3. Sim. Por exemplo, se a regra que diz que normalmente os ovos Brudy não têm colesterol passar a ser universal, deixamos de poder concluir que B1 tem colesterol. Também podemos deixar a regra como regra de omissão e acrescentar directamente que B1 tem colesterol, etc.
   Basicamente, todas as fbs que foram derivadas com base em regras de omissão podem deixar de ser acreditadas devido à adição de nova informação.

4. 

   Neste caso, fazia sentido testar se são extensão:

   $PExt1 = Th(\Delta \cup \{TemColesterol(O1), TemColesterol(C1), TemColesterol(B1)\})$
   $PExt2 = Th(\Delta \cup \{TemColesterol(O1), TemColesterol(C1), \neg TemColesterol(B1)\})$

5. Se acrescentássemos um outro ovo Brudy $B2$, passaríamos a ter 4 propostas de extensão, com as quatro possíveis combinações de $B1$ e $B2$ terem ou não terem colesterol. Em todas as propostas de extensão $O1$ e $C1$ têm colesterol.

Exercício 6.5
Represente a seguinte informação usando uma teoria da Lógica da Omissão de Reiter:

- Normalmente, os carros são rápidos, a não ser que estejam avariados.
- Em geral, os carros que não estão avariados não são rebocados, a não ser que estejam mal estacionados ou sejam roubados.
- O Herbie é um carro.
- Os Ferraris são carros rápidos.
- Tipicamente os Ferraris são vermelhos, ou pretos, ou amarelos.
- O Nuno tem um carro que é um Ferrari vermelho.

Resposta:

Teoria de omissão = (R, Δ)

$$R = \{\frac{Carro(x) : Rapido(x) \wedge \neg Avariado(x)}{Rapido(x)},$$
$$\frac{Carro(x) \wedge \neg Avariado(x) : \neg Rebocado(x) \wedge \neg MalEstacionado(x) \wedge \neg Roubado(x)}{\neg Rebocado(x)},$$
$$\frac{Ferrari(x) : Vermelho(x) \vee Preto(x) \vee Amarelo(x)}{Vermelho(x) \vee Preto(x) \vee Amarelo(x)}$$

$\Delta = \{Carro(Herbie),$
$\quad \forall(x)[Ferrari(x) \rightarrow (Carro(x) \wedge Rapido(x))],$
$\quad \exists(x)[Carro(x) \wedge Ferrari(x) \wedge Vermelho(x) \wedge Tem(Nuno, x)]\}$

Exercício 6.6
Considere a tradução da solução do primeiro TPC de 2003/2004 para Lógica De Primeira Ordem.

Diga que alterações é que deveria fazer a essa representação para a passar para a Lógica de Omissão do Reiter.

Resposta:

Segue-se a tradução da representação que foi feita para este TPC para LPO.

Factos que indicam as competências de cada trabalhador e as línguas estrangeiras que falam:

$TemComp(Joao, CapAnalise)$
$TemComp(Antonio, CapAnalise)$
$\ldots$
$TemComp(Duarte, Alemao)$
$TemComp(Fausto, Frances)$

$Cargo(Joao, Admin, Chefe)$
$Cargo(Helena, Admin, Secretar)$
$\ldots$
$Cargo(Bill, SubsEU, Programador)$
$Cargo(Stuart, SubsEU, Programador)$

Factos e regras que permitem derivar que os Portugueses falam Português e os Americanos falam Inglês:

$Nacionalidade(Joao, Portuguesa)$
$Nacionalidade(Helena, Portuguesa)$
$\ldots$
$Nacionalidade(Bill, Americana)$
$Nacionalidade(Stuart, Americana)$

$\forall(x)[Nacionalidade(x, Portuguesa) \rightarrow TemComp(x, Portugues)]$

$\forall(x)[Nacionalidade(x, Americana) \rightarrow TemComp(x, Ingles)]$

Factos e regras que permitem derivar as relações hierárquicas entre departamentos:

$Acima(Admin, DepCom)$
$Acima(Admin, DepProd)$
$Acima(Admin, DepInov)$
$Acima(Admin, SubsEU)$
$Acima(DepProd, EqDesenv)$
$Acima(DepProd, EqManut)$

$\forall(x, y, z)[(Acima(x, y) \wedge Acima(y, z)) \rightarrow Acima(x, z)]$

Regras que permitem deduzir as relações hierárquicas entre pessoas, dependendo das suas posições nas unidades da empresa:

$\forall(x,y)[Cargo(x,y,Chefe) \rightarrow ChefeDeUnidade(x,y)]$

$\forall(x,y,z,w)[(ChefeDeUnidade(x,y) \wedge ChefeDeUnidade(z,w) \wedge Acima(y,w)) \rightarrow ChefeDePessoa(x,z)]$

$\forall(x,y,z,w)[(Cargo(x,y,Chefe) \wedge Cargo(z,y,w)) \rightarrow ChefeDePessoa(x,z)]$

Factos e regras que permitem determinar os empregados disponíveis. Nesta solução considera-se que:

- A empresa não deve ficar sem administrador durante o projecto no Brasil, pelo que o chefe da administração não irá integrar a equipa.
- A secretária da administração também não irá fazer parte da equipa a definir, uma vez que, para além de ser a única secretária na empresa, existem outros empregados disponíveis com mais qualificações para orientação ao cliente.
- Os chefes de cada unidade permanecem na empresa para que estas não fiquem sem chefia e para reduzir o número de opções possíveis, dado que existem outras alternativas viáveis com pessoas qualificadas.

$\forall(x,y)[(ChefeDePessoa(y,x) \wedge PodeLibertarRecursos(y)) \rightarrow Disponivel(x)]$

$PodeLibertarRecursos(Jose)$
$PodeLibertarRecursos(Ana)$
$PodeLibertarRecursos(Isabel)$

Regras que permitem derivar quem irá ocupar cada um dos cargos na equipa para o Brasil:

$\forall(x)[(Disponivel(x) \wedge TemComp(x,OrientCliente) \wedge TemComp(x,Portugues)) \rightarrow RespClienteEscolhido(x)]$

$\forall(x)[(Disponivel(x) \wedge TemComp(x,CapAnalise) \wedge TemComp(x,Ingles)) \rightarrow AnalistaEscolhido(x)]$

$\forall(x)[(Disponivel(x) \wedge TemComp(x,CapAnalise) \wedge TemComp(x,Portugues)) \rightarrow AnalistaEscolhido(x)]$

$\forall(x)[(Disponivel(x) \wedge TemComp(x,CapChefia) \wedge TemComp(x,Portugues)) \rightarrow ChefeEscolhido(x)]$

$\forall(x)[(Disponivel(x) \wedge TemComp(x,Programa) \wedge TemComp(x,Portugues)) \rightarrow ProgramadorEscolhido(x)]$

$\forall(x,y,z,w)[(RespClienteEscolhido(x) \wedge AnalistaEscolhido(y) \wedge ChefeEscolhido(z) \wedge ProgramadorEscolhido(w)) \rightarrow Equipa(x,y,z,w)]$

Fim da representação em LPO

Podemos ter uma regra de omissão que represente que tipicamente os empregados estão disponíveis. Só os que são "explicitamente" ditos é que não estão. Neste caso, e para não ficarmos com uma regra do género $\frac{:Disponivel(x)}{Disponivel(x)}$, que poderia ser aplicada a qualquer constante da base de conhecimento, devemos criar um predicado $Empregado$, em que $Empregado(x)$ significa que x é um empregado desta empresa, e acrescentar à base de conhecimento:

$Empregado(Joao)$
$Empregado(Antonio)$
...
$Empregado(Duarte)$
$Empregado(Fausto)$

Nota: Em vez de termos que cada empregado é um empregado, podemos ter uma regra mais geral: $\forall(x,y,z)[Cargo(x,y,z) \rightarrow Empregado(x)]$.

A regra de omissão seria: $\frac{Empregado(x):Disponivel(x)}{Disponivel(x)}$

E, para sabermos quem é que não está disponível, substituíamos o que tínhamos por:

$\forall(x,y)[(ChefeDePessoa(y,x) \wedge TemEquipaOcupada(y)) \rightarrow \neg Disponivel(x)]$

$TemEquipaOcupada(Ricardo)$
$TemEquipaOcupada(Paula)$

Convém ter em atenção que agora os chefes também estão disponíveis, bem como a secretária. Se quisermos, podemos escrever regras específicas para o impedir:

$\forall(x,y)[Cargo(x,y,Chefe) \rightarrow \neg Disponivel(x)]$
$\neg Disponivel(Helena)$

A língua materna também pode ser representada por regras de omissão em vez de regras universais, pois pode haver pessoas com uma determinada nacionalidade que se mudaram para outro país e que por isso não falam a língua correspondente à sua nacionalidade.

$\frac{Nacionalidade(x,Portuguesa):TemComp(x,Portugues)}{TemComp(x,Portugues)}$

$\frac{Nacionalidade(x,Americana):TemComp(x,Ingles)}{TemComp(x,Ingles)}$

7 Lógica não monótona — LOR, Cálculo de extensões

Sumário:

- Lógica da omissão de Reiter: determinação de extensões de teorias pelas vias sintáctica e semântica.

Resumo:

Via sintáctica

Definição de extensão de uma teoria de omissão:

Ω é uma extensão da teoria de omissão (R, Δ) sse $\Gamma(\Omega) = \Omega$, em que $\Gamma(\Omega)$ é o menor conjunto que satisfaz as seguintes condições:

1. $\Delta \subset \Gamma(\Omega)$
2. $\Gamma(\Omega)$ é fechado quanto à derivabilidade, isto é, $th(\Gamma(\Omega)) = \Gamma(\Omega)$
3. Se $\frac{\alpha:\beta}{\gamma} \in R$, $\alpha \in \Gamma(\Omega)$ e $\neg\beta \notin \Omega$ então $\gamma \in \Gamma(\Omega)$

Assim, para Ω ser uma extensão da teoria de omissão (R, Δ), tem que satisfazer as cinco condições enunciadas na definição acima: Ω é um ponto fixo do operador Γ, $\Gamma(\Omega)$ é mínimo, e as condições 1. a 3. são satisfeitas.

Método para verificar se um conjunto de fbfs é extensão de uma teoria de omissão:

Seja $\epsilon \subseteq \mathcal{L}$ um conjunto de fbfs fechadas e seja (R, Δ) uma teoria de omissão fechada. Defina-se:

$$\epsilon_0 = \Delta$$

$$\begin{aligned}\epsilon_{i+1} = \ & th(\epsilon_i) \cup \\ & \{\gamma : \tfrac{\alpha:\beta_1,\ldots,\beta_m}{\gamma} \in R, \text{ em que } \alpha \in \epsilon_i \text{ e } \neg\beta_1, \ldots, \neg\beta_m \notin \boldsymbol{\epsilon}\}\end{aligned}$$

então, ϵ é uma extensão de (R, Δ) sse $\epsilon = \bigcup_{i=0}^{\infty} \epsilon_i$.

Nota: Este não é um método construtivo por causa do ϵ que está a bold na definição, o que implica que é necessário ter à partida uma proposta de extensão.

Como determinar as extensões de uma teoria de omissão pela via sintáctica:

Não sabemos calcular as extensões de uma teoria de omissão.

No entanto, podemos verificar se um conjunto de fbfs Ω é uma extensão da teoria de omissão (R, Δ), quer verificando se ele respeita a definição de extensão, quer através do método de verificação de extensões.

Em qualquer dos casos, o primeiro passo será sempre decidir quais são as propostas de extensão que devemos considerar, tendo em conta que não vale a pena considerar propostas que à partida saibamos que não vão satisfazer a definição de extensão.

Para satisfazerem a condição 1. da definição de extensão, as propostas de extensão têm obrigatoriamente que conter Δ, pois os conjuntos que não contenham Δ nunca vão poder ser extensões da teoria.

Para satisfazerem a condição 2. da definição de extensão, as propostas de extensão têm que ser fechadas quanto à derivabilidade, por isso só vale a pena considerar todos os teoremas que possam ser derivados a partir de conjuntos que contenham Δ.

Para satisfazerem a condição 3. da definição de extensão, as propostas de extensão têm que incluir as concluões de todas as regras de omissão que possam ser aplicadas. Assim, as possíveis extensões devem ser os teoremas de Δ reunido com cada uma das combinações das conclusões de todas as regras de omissão que existirem na teoria.

Não vale a pena considerar extensões que sejam conjuntos que tenham mais fbfs porque esses conjuntos não iriam ser mínimos e por isso não iriam satisfazer a noção de extensão.

Os conjuntos contraditórios também não podem ser extensão de nenhuma teoria de omissão (a não ser que $th(\Delta)$ já fosse contraditório), pois a partir de uma contradição pode derivar-se qualquer coisa. Logo, nos conjuntos contraditórios nunca vai ser possível assumir a justificação de nenhuma regra de omissão, pois tanto a justificação como a sua negação fazem parte do conjunto. Para além disso, contêm fórmulas que não foram derivadas a partir da teoria. No caso em que $th(\Delta)$ é contraditório, já é possível derivar qualquer coisa a partir de Δ, não se pode aplicar qualquer regra de omissão e por isso $th(\Delta)$ vai ser extensão da teoria, apesar de ser contraditório. Convém notar que também teríamos um conjunto de crenças contraditório se estivéssemos a usar apenas a lógica de primeira ordem.

Uma vez que sabemos que as extensões correspondem a conjuntos de fbfs em que se aplicaram todas as regras de omissão que poderiam ser aplicadas (pelo ponto 3. da definição de extensão), convém começar a verificar se são extensão os conjuntos com mais fbfs, pois se um conjunto for extensão, nenhum outro conjunto que esteja contido nele vai ser extensão.

Se tivermos uma teoria de omissão com uma regra de omissão com conclusão A, temos duas propostas de extensão razoáveis: $th(\Delta)$ e $th(\Delta \cup \{A\})$.

Se a teoria tiver mais uma regra de omissão com conclusão B, temos quatro propostas de extensão razoáveis: as duas anteriores mais $th(\Delta \cup \{B\})$ e $th(\Delta \cup \{A, B\})$.

Se a teoria tiver ainda mais uma regra de omissão com conclusão C, temos oito propostas de extensão razoáveis: as quatro anteriores mais $th(\Delta \cup \{C\})$, $th(\Delta \cup \{A, C\})$, $th(\Delta \cup \{B, C\})$ e $th(\Delta \cup \{A, B, C\})$. E assim sucessivamente.

No caso de a teoria de omissão ter regras de omissão com variáveis, é como se cada regra de omissão da teoria correspondesse a várias regras de omissão, correspondendo cada uma delas à aplicação da regra a uma constante.

Via semântica

Pela via semântica, as extensões são calculadas usando modelos de conjuntos de fórmulas. Alguns exemplos:

- $\mathcal{M}_1 = \{M : M \models \{\}\}$
  São consequência lógica deste conjunto de modelos todos os teoremas da LPO, nomeadamente $A \rightarrow (B \rightarrow A)$, $A \vee \neg A$, $(A \wedge \neg A) \rightarrow B$, etc.

- $\mathcal{M}_2 = \{M : M \models \{A\}\}$
  São consequência lógica deste conjunto de modelos todos os teoremas da LPO, mais todas as fórmulas que for possível derivar a partir de A, nomeadamente $C \rightarrow A$, $A \vee FALSO$, $\neg A \rightarrow D$, etc. Este conjunto de modelos tem mais consequências lógicas mas menos modelos do que $\mathcal{M}_1$, pois dos modelos de $\mathcal{M}_1$ foram eliminados todos aqueles em que A é falso.

- $\mathcal{M}_3 = \{M : M \models \{A, B\}\}$
  São consequência lógica deste conjunto de modelos todos os teoremas da LPO, mais todas as fórmulas que for possível derivar a partir de A e de B, nomeadamente $C \rightarrow A$, $A \vee FALSO$, $\neg A \rightarrow D$, $C \rightarrow B$, $B \vee FALSO$, $\neg B \rightarrow D$, $A \rightarrow B$, $B \rightarrow A$, etc. Este conjunto de modelos tem mais consequências lógicas mas menos modelos do que $\mathcal{M}_2$, pois dos modelos de $\mathcal{M}_2$ foram eliminados todos aqueles em que B é falso.

- $\mathcal{M}_4 = \{M : M \models \{A, A \rightarrow B\}\}$
  Este conjunto de modelos é igual a $\mathcal{M}_3$, pois se temos A e $A \rightarrow B$ também temos B, por isso vamos ter exactamente as mesmas consequências lógicas e os mesmos modelos que tínhamos em $\mathcal{M}_3$.

- $\mathcal{M}_5 = \{M : M \models \{A, \neg A\}\}$
  Este conjunto de modelos é vazio, pois não conseguimos encontrar qualquer valoração que dê a A simultaneamente o valor $VERDADEIRO$ e $FALSO$. Mas também tem infinitas consequências lógicas, neste caso todas as fbfs da LPO.

As extensões são calculadas usando a noção de uma regra de omissão preferir um conjunto de modelos relativamente a outro (fazendo as árvores). Quando um conjunto de modelos é máximo e estável, é modelo de uma extensão da teoria.

Um conjunto de modelos é máximo quando não se puder acrescentar mais fórmulas através da aplicação de regras de omissão.

Um conjunto de modelos é estável quando ainda se puderem aplicar as regras que foram aplicadas ao longo de pelo menos um dos caminhos para chegar até ele.

Podemos "aplicar" uma regra de omissão a um conjunto de modelos sse "soubermos" a pré-condição e "for consistente assumir" as justificações, ou seja, se a pré-condição for consequência lógica de todos os modelos do conjunto e cada justificação for consequência lógica de pelo menos um dos modelos do conjunto. Neste caso, podemos passar para um novo conjunto de modelos, em que podemos derivar tudo o que derivávamos anteriormente mais a conclusão da regra de omissão que "aplicámos" para lá chegar.

Nota: Devemos ter em atenção que os conjuntos de modelos de fórmulas contraditórias (estes conjuntos de modelos são vazios, pois é impossível ter uma valoração que dá simultaneamente os valores $VERDADEIRO$ e $FALSO$ a uma determinada fórmula) não são extensão da teoria, pois deixa de ser possível assumir as justificações das regras de omissão que foram aplicadas (se o conjunto de modelos é vazio, então não existe pelo menos um modelo que satisfaça a justificação) e por isso não são estáveis. O único caso em que uma teoria de omissão pode ter uma extensão contraditória é quando as próprias fórmulas de Δ (e os seus teoremas) já dão origem a uma contradição. Neste caso, o conjunto de modelos inicial é vazio, não se consegue aplicar nenhuma regra de omissãoe por isso o conjunto de modelos é máximo e estável.

Exercício 7.1

Considere a teoria de omissão $(\{r_1, r_2\}, \Delta)$, em que:

$r_1 = \frac{Ovo(x) : TemColesterol(x)}{TemColesterol(x)}$, $\quad r_2 = \frac{OvoBrudy(x) : \neg TemColesterol(x)}{\neg TemColesterol(x)}$

$$\begin{aligned}\Delta = &\{\forall(x)[OvoNormal(x) \rightarrow Ovo(x)],\\ &\forall(x)[OvoBrudy(x) \rightarrow Ovo(x)],\\ &OvoNormal(O1),\\ &OvoBrudy(B1)\,\}\end{aligned}$$

1. Calcule as suas extensões pela via sintáctica.

2. Calcule as suas extensões pela via semântica.

Resposta:

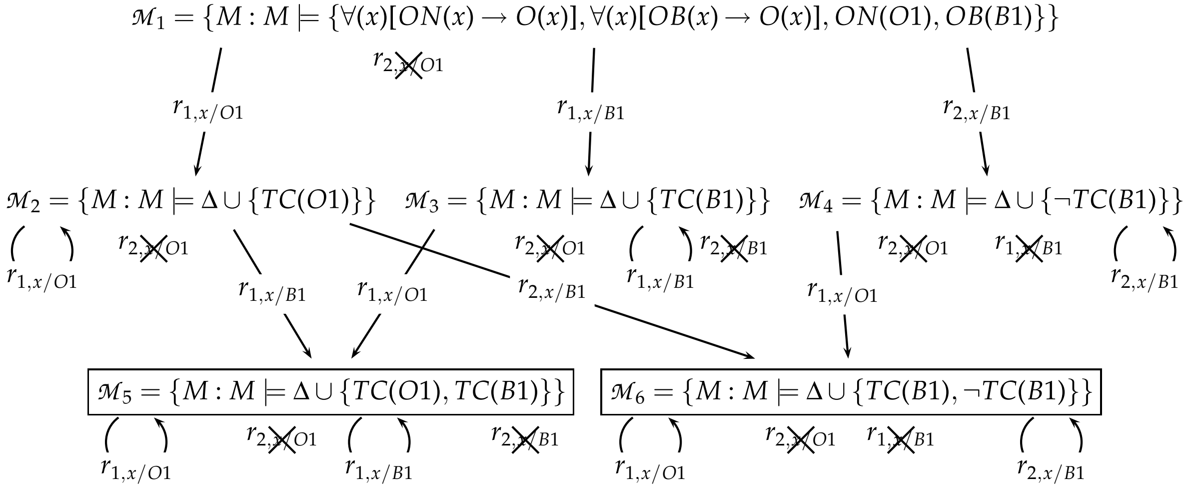
1. Para calcular as extensões da teoria pela via sintáctica, vamos considerar apenas as propostas de extensão que discutimos na aula anterior, que correspondem aos teoremas de Δ reunidos com as combinações das conclusões das regras de omissão da teoria que podem ser aplicadas a cada uma das instâncias.
   $PExt1 = Th(\Delta \cup \{TemColesterol(O1), TemColesterol(B1)\})$
   $PExt2 = Th(\Delta \cup \{TemColesterol(O1), \neg TemColesterol(B1)\})$

   - $\epsilon = PExt1$
     $\epsilon_0 = \Delta$
     $\epsilon_1 = th(\Delta) \cup \{\}$
     $\epsilon_2 = th(\Delta) \cup \{TemColesterol(O1), TemColesterol(B1)\}$
     $\epsilon_3 = th(\Delta \cup \{TemColesterol(O1), TemColesterol(B1)\}) \cup \{\}$
     $\vdots$
     É extensão porque $\epsilon = \bigcup_{i=0}^{\infty} \epsilon_i$

   - $\epsilon = PExt2$
     $\epsilon_0 = \Delta$
     $\epsilon_1 = th(\Delta) \cup \{\neg TemColesterol(B1)\}$
     $\epsilon_2 = th(\Delta \cup \{\neg TemColesterol(B1)\}) \cup \{TemColesterol(O1)\}$
     $\epsilon_3 = th(\Delta \cup \{\neg TemColesterol(B1), TemColesterol(O1)\}) \cup \{\}$
     $\vdots$
     É extensão porque $\epsilon = \bigcup_{i=0}^{\infty} \epsilon_i$

2. Para calcular as extensões da teoria pela via semântica, vamos determinar os modelos das

extensões da teoria.



Como $\mathcal{M}_5$ e $\mathcal{M}_6$ são máximos e estáveis, são modelos de extensões da teoria.
As extensões são
$Ext_1 = th(\Delta \cup \{TC(O1), TC(B1)\})$ e
$Ext_2 = th(\Delta \cup \{TC(B1), \neg TC(B1)\})$.

Exercício 7.2

Considere a seguinte teoria de omissão:

$\mathcal{T} = (\{r_1, r_2, r_3\}, \{P \to \neg Q, Q \to \neg R\})$

$r_1 = \frac{:P}{P}, r_2 = \frac{:Q \wedge R}{Q}, r_3 = \frac{:R}{R}$

1. Calcule as suas extensões pela via sintáctica.
2. Calcule as suas extensões pela via semântica.

Resposta:

1. Existem 8 propostas de extensão, que correspondem aos teoremas de Δ reunido com cada uma das combinações das conclusões de todas as regras de omissão que existirem na teoria.
   $PExt_1 = th(\Delta)$
   $PExt_2 = th(\Delta \cup \{P\})$
   $PExt_3 = th(\Delta \cup \{Q\})$
   $PExt_4 = th(\Delta \cup \{R\})$
   $PExt_5 = th(\Delta \cup \{P, Q\})$
   $PExt_6 = th(\Delta \cup \{P, R\})$
   $PExt_7 = th(\Delta \cup \{Q, R\})$
   $PExt_8 = th(\Delta \cup \{P, Q, R\})$
   Vamos começar por verificar se as propostas de extensão são extensão pelas que "contêm mais conclusões" porque se alguma destas for extensão, nenhuma das que estão contidas nela podem ser extensão.
   - $\epsilon = PExt_8$
     Vamos começar por "simplificar" este conjunto:
     $PExt_8 = th(\Delta \cup \{P, Q, R\})$
     $\quad = th(\{P \to \neg Q, Q \to \neg R, P, Q, R, \neg Q, \neg R\})$
     Este conjunto de fórmulas é contraditório e por isso não pode ser extensão de uma teoria de omissão.

- $\epsilon = PExt_7$
  $PExt_7 = th(\Delta \cup \{Q, R\})$
  $\qquad = th(\{P \rightarrow \neg Q, Q \rightarrow \neg R, Q, R, \neg P, \neg R\})$
  Este conjunto de fórmulas é contraditório e por isso não pode ser extensão de uma teoria de omissão.

- $\epsilon = PExt_6$
  $PExt_6 = th(\Delta \cup \{P, R\})$
  $\qquad = th(\{P \rightarrow \neg Q, Q \rightarrow \neg R, P, R, \neg Q\})$
  Vamos aplicar o método para verificar se este conjunto de fórmulas é extensão desta teoria de omissão.
  $\epsilon_0 = \Delta$
  $\epsilon_1 = th(\Delta) \cup \{P, R\}$
  $\epsilon_2 = th(\Delta \cup \{P, R\}) \cup \{\}$ Nota: $\epsilon_2 \supset \{\neg Q\}$
  $\vdots$
  $\boxed{PExt_6 \text{ é extensão porque } \epsilon = \bigcup_{i=0}^{\infty} \epsilon_i}$

- $\epsilon = PExt_5$
  $PExt_5 = th(\Delta \cup \{P, Q\})$
  $\qquad = th(\{P \rightarrow \neg Q, Q \rightarrow \neg R, P, Q, \neg Q\})$
  Este conjunto de fórmulas é contraditório e por isso não pode ser extensão de uma teoria de omissão.

- $\epsilon = PExt_4$
  $PExt_4 = th(\Delta \cup \{R\})$
  $\qquad = th(\{P \rightarrow \neg Q, Q \rightarrow \neg R, R, \neg Q\})$
  Este conjunto de fórmulas não pode ser extensão desta teoria de omissão porque está contido em $PExt_6$ que é extensão desta teoria. Se quisermos, podemos ver o que acontece aplicando o método:
  $\epsilon_0 = \Delta$
  $\epsilon_1 = th(\Delta) \cup \{P, R\}$
  $\vdots$
  $PExt_4$ não é extensão porque $\epsilon \neq \bigcup_{i=0}^{\infty} \epsilon_i$

- $\epsilon = PExt_3$
  $PExt_3 = th(\Delta \cup \{Q\})$
  $\qquad = th(\{P \rightarrow \neg Q, Q \rightarrow \neg R, Q, \neg P, \neg R\})$
  Vamos aplicar o método para verificar se este conjunto de fórmulas é extensão desta teoria de omissão.
  $\epsilon_0 = \Delta$
  $\epsilon_1 = th(\Delta) \cup \{\}$
  $\vdots$
  $PExt_3$ não é extensão porque $\epsilon \neq \bigcup_{i=0}^{\infty} \epsilon_i$

- $\epsilon = PExt_2$
  $PExt_2 = th(\Delta \cup \{P\})$
  $\qquad = th(\{P \rightarrow \neg Q, Q \rightarrow \neg R, P, \neg Q\})$
  Este conjunto de fórmulas não pode ser extensão desta teoria de omissão porque está contido em $PExt_6$ que é extensão desta teoria. Se quisermos, podemos ver o que acontece aplicando o método:
  $\epsilon_0 = \Delta$
  $\epsilon_1 = th(\Delta) \cup \{P, R\}$
  $\vdots$
  $PExt_2$ não é extensão porque $\epsilon \neq \bigcup_{i=0}^{\infty} \epsilon_i$

- $\epsilon = PExt_1 = th(\Delta)$
  Este conjunto de fórmulas não pode ser extensão desta teoria de omissão porque está contido em $PExt_6$ que é extensão desta teoria.

  Esta teoria tem uma extensão, que corresponde a $PExt_6$:
  $PExt_6 = th(\Delta \cup \{P, R\})$
  $\qquad = th(\{P \to \neg Q, Q \to \neg R, P, R, \neg Q\})$

2. $\mathcal{T} = (\{r_1, r_2, r_3\}, \{P \to \neg Q, Q \to \neg R\})$

   $r_1 = \frac{:P}{P}$, $r_2 = \frac{:Q \wedge R}{Q}$, $r_3 = \frac{:R}{R}$

$$
\begin{tikzcd}
& \mathcal{M}_1 = \{M : M \models \{P \to \neg Q, Q \to \neg R\}\} \arrow[dl, "r_1"'] \arrow[dr, "r_3"] \arrow[loop below, phantom, "\cancel{r_2}"] & \\
\mathcal{M}_2 = \{M : M \models \{P \to \neg Q, Q \to \neg R, P, \neg Q\}\} \arrow[loop below, "r_1"] \arrow[loop right, phantom, "\cancel{r_2}"] \arrow[dr, "r_3"] & & \mathcal{M}_3 = \{M : M \models \{P \to \neg Q, Q \to \neg R, R, \neg Q\}\} \arrow[loop below, "r_3"] \arrow[loop left, phantom, "\cancel{r_2}"] \arrow[dl, "r_1"'] \\
& \boxed{\mathcal{M}_4 = \{M : M \models \{P \to \neg Q, Q \to \neg R, P, \neg Q, R\}\}} \arrow[loop left, "r_1"] \arrow[loop right, "r_3"] \arrow[loop below, phantom, "\cancel{r_2}"] &
\end{tikzcd}
$$

   Como $\mathcal{M}_4$ é máximo e estável, é modelo de uma extensão da teoria.
   A extensão é $th(\{P \to \neg Q, Q \to \neg R, P, \neg Q, R\})$.

Exercício 7.3
Calcule, pela via semântica, as extensões da teoria de omissão $\mathcal{T}_{12} = (\{r_1, r_2, r_3, r_4\}, \{P, S\})$

$r_1 = \frac{P:Q}{Q}$, $r_2 = \frac{S:T}{T}$, $r_3 = \frac{Q \wedge T : R \wedge U}{R \wedge U}$, $r_4 = \frac{:\neg R}{\neg R}$

Resposta:

Exercício 7.4
Considere a seguinte teoria de omissão = $(\{r_1, r_2, r_3\}, \Delta)$

$r_1 = \frac{Mamifero(x) : ProduzLeite(x)}{ProduzLeite(x)}$
$r_2 = \frac{TemPelos(x) : Mamifero(x)}{Mamifero(x)}$
$r_3 = \frac{PoeOvos(x) : \neg Mamifero(x)}{\neg Mamifero(x)}$
$\Delta = \{\forall(x)[Ornitorrinco(x) \to TemPelos(x)],$
$\quad \forall(x)[Ornitorrinco(x) \to PoeOvos(x)],$
$\quad Ornitorrinco(Flip),$
$\quad TemPelos(Lassie)\ \}$

1. Calcule as suas extensões pela via sintáctica.

2. Calcule as suas extensões pela via semântica.

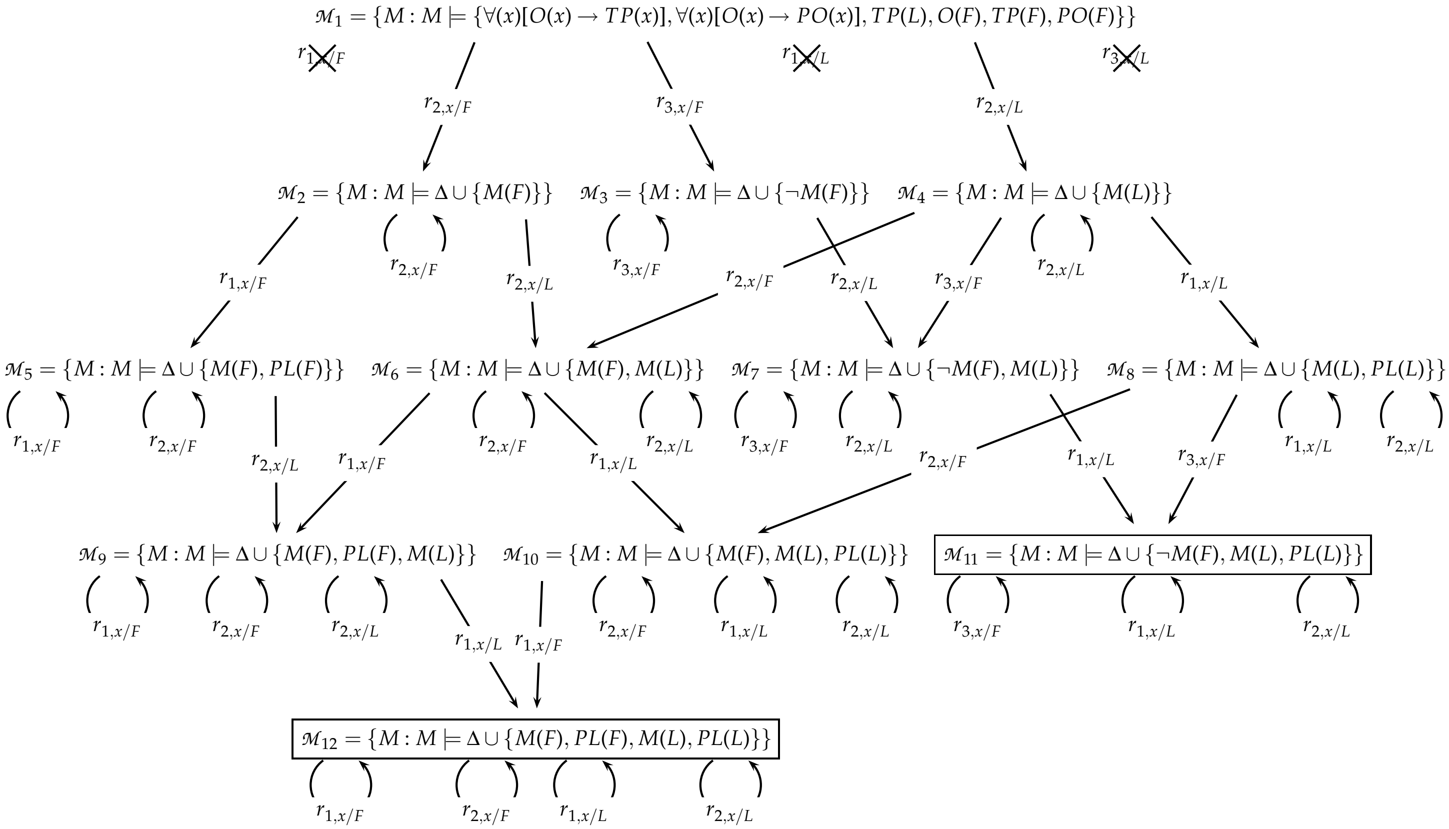
Resposta:

1. Antes de começar a propor extensões, convém ter em atenção que $th(\Delta) \supset \{TemPelos(Flip), PoeOvos(Flip)\}$.
   Por outro lado, $\epsilon_0 = \Delta$, e não $th(\Delta)$, ou seja, devemos ter em atenção que a regra r_2 não é aplicável ao $Flip$ na passagem de ϵ_0 para ϵ_1.
   $$\begin{aligned} PExt_1 = th(\Delta \cup \{ \ & Mamifero(Flip), ProduzLeite(Flip), \\ & Mamifero(Lassie), ProduzLeite(Lassie)\}) \end{aligned}$$
   $PExt_2 = th(\Delta \cup \{\neg Mamifero(Flip), Mamifero(Lassie), ProduzLeite(Lassie)\})$

   - $\epsilon = PExt_1$
     $\epsilon_0 = \Delta$
     $\epsilon_1 = th(\Delta) \cup \{Mamifero(Lassie)\}$
     $\epsilon_2 = th(\Delta \cup \{Mamifero(Lassie)\}) \cup \{Mamifero(Flip), ProduzLeite(Lassie)\}$
     $\epsilon_3 = th(\Delta \cup \{Mamifero(Lassie), Mamifero(Flip), ProduzLeite(Lassie)\}) \cup \{ProduzLeite(Flip)\}$
     $\epsilon_4 = th(\Delta \cup \{Mamifero(Lassie), Mamifero(Flip),$
     $ProduzLeite(Lassie), ProduzLeite(Flip)\}) \cup \{\}$
     $\vdots$
     É extensão porque $\epsilon = \bigcup_{i=0}^{\infty} \epsilon_i$

   - $\epsilon = PExt_2$
     $\epsilon_0 = \Delta$
     $\epsilon_1 = th(\Delta) \cup \{Mamifero(Lassie)\}$
     $\epsilon_2 = th(\Delta \cup \{Mamifero(Lassie)\}) \cup \{\neg Mamifero(Flip), ProduzLeite(Lassie)\}$
     $\epsilon_3 = th(\Delta \cup \{Mamifero(Lassie), \neg Mamifero(Flip), ProduzLeite(Lassie)\}) \cup \{\}$
     $\vdots$
     É extensão porque $\epsilon = \bigcup_{i=0}^{\infty} \epsilon_i$

2. $\mathcal{T} = (\{r_1, r_2, r_3\}, \Delta)$
   $r_1 = \frac{Mamifero(x) : ProduzLeite(x)}{ProduzLeite(x)}$
   $r_2 = \frac{TemPelos(x) : Mamifero(x)}{Mamifero(x)}$
   $r_3 = \frac{PoeOvos(x) : \neg Mamifero(x)}{\neg Mamifero(x)}$
   $$\begin{aligned} \Delta = \{ & \forall(x)[Ornitorrinco(x) \rightarrow TemPelos(x)], \\ & \forall(x)[Ornitorrinco(x) \rightarrow PoeOvos(x)], \\ & Ornitorrinco(Flip), \\ & TemPelos(Lassie) \} \end{aligned}$$



Notas: Neste exercício só no primeiro conjunto de modelos é que se representaram as regras que não são aplicáveis, para facilitar a representação. Para além disso, e uma vez que esta teoria de omissão só tem regras de omissão normais, não é necessário testar as condições de estabilidade para os conjuntos máximos de modelos; isso só foi feito para mostrar que eles realmente são estáveis.

Existem dois conjuntos de modelos máximos e estáveis, $\mathcal{M}_{11}$ e $\mathcal{M}_{12}$. As Extensões são:
$Ext_1 = th(\Delta \cup \{\neg M(F), M(L), PL(L)\})$
$Ext_2 = th(\Delta \cup \{M(F), PL(F), M(L), PL(L)\})$

Exercício 7.5
Determine as extensões das teorias de omissão seguintes pela via semântica:

1. $\mathcal{T}_{13} = (\{r_1, r_2, r_3\}, \{P, S, U\})$
   $r_1 = \frac{P : Q \wedge \neg R}{Q}, r_2 = \frac{S : R \wedge \neg T}{R}, r_3 = \frac{U : T \wedge \neg Q}{T}$
2. $\mathcal{T}_{14} = (\{r_1, r_2, r_3\}, \{P\})$
   $r_1 = \frac{P : Q \wedge R}{Q \wedge R}, r_2 = \frac{P : Q, \neg Q}{R}, r_3 = \frac{P \wedge R : Q}{Q}$
3. $\mathcal{T}_{15} = (\{r_1, r_2, r_3\}, \{P\})$
   $r_1 = \frac{P : Q}{R}, r_2 = \frac{R : P}{\neg Q}, r_3 = \frac{: P \wedge R}{R \wedge \neg Q}$
4. $\mathcal{T}_{16} = (\{r_1, r_2, r_3\}, \{V \to A, V \to C, H, P\})$
   $r_1 = \frac{H : V}{V}, r_2 = \frac{P : \neg A}{\neg A}, r_3 = \frac{P : C}{C}$

5. $\mathcal{T}_{17} = (\{r_1, r_2, r_3\}, \{Q\})$
   $r_1 = \frac{:P, \neg P}{P}$, $r_2 = \frac{Q:R}{R}$, $r_3 = \frac{R:\neg P}{\neg P}$

6. $\mathcal{T}_{18} = (\{r_1, r_2, r_3\}, \{\neg P \vee \neg Q, \neg P \rightarrow R, S\})$
   $r_1 = \frac{S:P \wedge R}{P}$, $r_2 = \frac{:\neg P}{Q}$, $r_3 = \frac{S:Q}{Q}$

7. $\mathcal{T}_{19} = (\{r_1, r_2, r_3\}, \{P \vee Q, P \rightarrow R\})$
   $r_1 = \frac{:P}{\neg R}$, $r_2 = \frac{P:Q}{Q}$, $r_3 = \frac{:R \wedge Q}{R \wedge Q}$

8. $\mathcal{T}_{20} = (\{r_1, r_2\}, \{A, B, C\})$
   $r_1 = \frac{Morcego(x):Fdd(x,Voar)}{Fdd(x,Voar)}$, $r_2 = \frac{Mamifero(x):Fdd(x,Andar)}{Fdd(x,Andar)}$,
   $A = Morcego(Vampy)$,
   $B = \forall(x)[Morcego(x) \rightarrow Mamifero(x)]$,
   $C = \forall(x)[Fdd(x, Andar) \leftrightarrow \neg Fdd(x, Voar)]$

9. $\mathcal{T}_{21} = (\{r_1, r_2\}, \{A, B, C\})$
   $r_1 = \frac{Estudioso(x):Sabio(x)}{Sabio(x)}$, $r_2 = \frac{Politico(x):\neg Sabio(x)}{\neg Sabio(x)}$,
   $A = \forall(x)[Sabio(x) \rightarrow Excentrico(x)]$, $B = Estudioso(Luis)$, $C = Politico(Luis)$

Resposta:

8 Sistemas de revisão de crenças — JTMS e ATMS

Sumário:

- TMSs e redes de dependências
- JTMS
- ATMS

Resumo:

- Tarefas dos TMSs
  - Mantêm a consistência de um conjunto de crenças
  - Só mantêm crenças para as quais tiverem uma razão
  - Explicam as razões para manter cada crença
  - Identificam contradições
  - Identificam os culpados de contradições
  - Evitam o re-aparecimento de contradições conhecidas

  Isto é possível porque os TMSs mantêm um registo das dependências entre as crenças, através da utilização de uma rede de dependências. Nessas redes, os nós representam proposições atómicas ou a sua negação ($\Rightarrow$ não há variáveis nos nós).

- **JTMSs**
  O rótulo de um nó indica se ele é acreditado ou não acreditado.

  Um nó é *acreditado* quando tem pelo menos uma justificação válida.

  Uma justificação é *válida* quando tem todos os seus antecedentes monótonos acreditados e nenhum não monótono acreditado.

  Num ciclo, sempre que se começa a rotular por um determinado nó começa-se com o rótulo NA (pois até esse momento ainda não se encontrou nenhuma justificação válida), a não ser que ele seja uma premissa ou tenha pelo menos uma justificação válida que não dependa do ciclo.

  Ciclos num JTMS — Existem vários tipos de ciclos num JTMS, que podem ser classificados de acordo com o número de justificações não monótonas que atravessam:

  - Os ciclos monótonos não atravessam nenhuma justificação não-monótona e em geral têm apenas uma rotulação estável.
  - Os ciclos pares atravessam um número par de justificações não-monótonas e em geral têm duas rotulações estáveis.
  - Os ciclos ímpares atravessam um número ímpar de justificações não-monótonas e em geral não têm nenhuma rotulação estável.

Remoção de contradições — Depois da detecção da contradição, se for possível removê-la alterando uma escolha feita durante a rotulação dum ciclo ímpar, altera-se; caso contrário, faz-se o retrocesso dirigido pelas dependências, para cada uma das justificações válidas para a contradição:

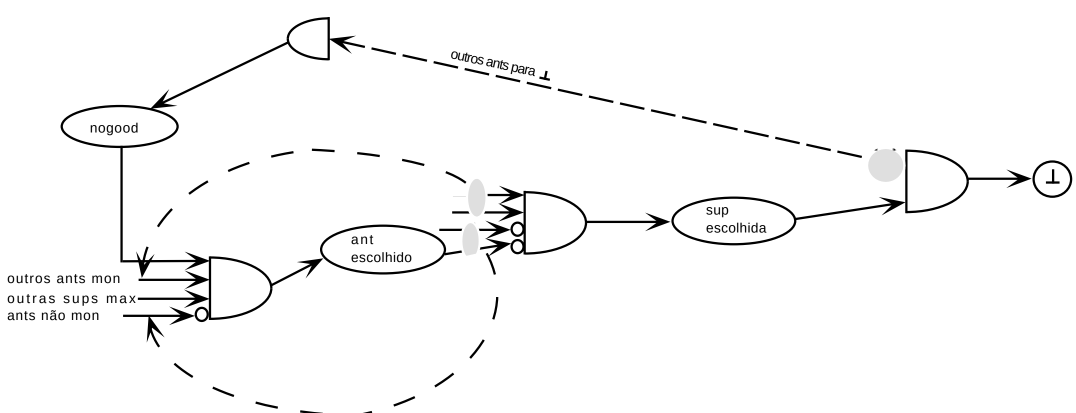
1. Determinar as suposições máximas: suposições usadas para determinar o estado da contradição, mas que não foram usadas para determinar o estado de qualquer outra suposição usada para determinar o estado da contradição, ou seja, são as suposições que estão "mais perto" da contradição. (Uma suposição é um nó com pelo menos uma justificação não monótona.)

2. Se a contradição não depender de nenhuma suposição, é necessário remover uma das premissas subjacentes; neste caso, o JTMS ou escolhe uma ao acaso ou pergunta ao solucionador de problemas qual deve remover. Caso exista mais do que uma suposição máxima na origem da contradição, o JTMS escolhe (ao acaso) uma das suposições máximas para deixar de acreditar.

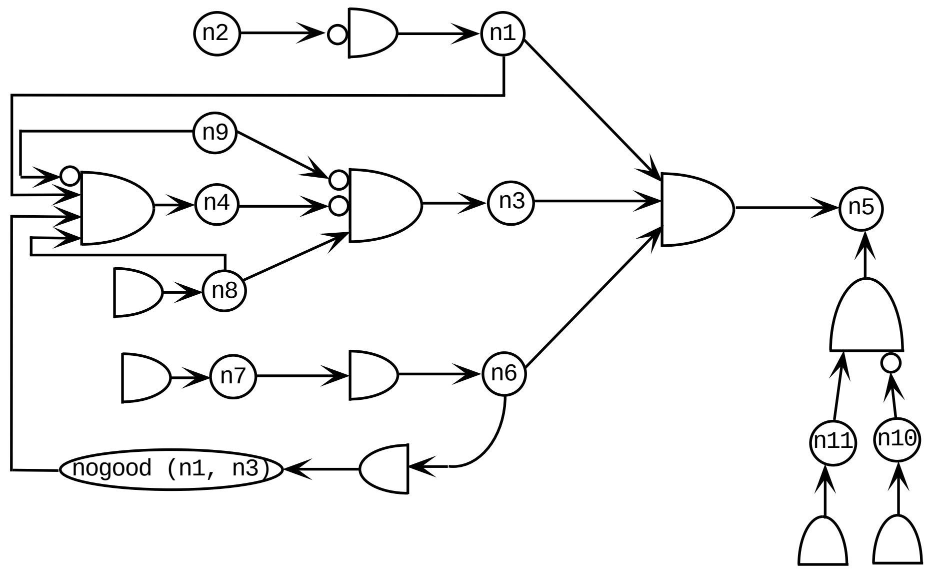
3. Escolher um dos antecedentes não-monótonos da suposição máxima escolhida para passar a ser acreditado. Para isso, vai-lhe ser adicionada uma justificação que só deve ser válida nos casos em que a sua não validade implicaria o aparecimento da contradição.

4. A justificação para o antecedente não-monótono que vai passar a ser acreditado tem como antecedentes monótonos as outras suposições máximas, os antecedentes monótonos da suposição máxima escolhida e o nó nogood; tem como antecedentes não-monótonos os outros antecedentes não-monótonos da suposição máxima escolhida.

5. O nó nogood tem como antecedentes monótonos os antecedentes da justificação para a contradição que não se baseiam em suposições.



Segue-se um exemplo da remoção duma contradição num JTMS, supondo que $n5$ é um nó contradição:



Alterações:

1. Remover a justificação vazia a $n7$
2. Adicionar uma justificação vazia a $n2$
3. Remover a justificação vazia a $n8$

Se $n10$ deixasse de ser acreditado, voltaríamos a ter uma justificação válida para a contradição. Neste caso, seria necessário repetir o processo e criar um novo *nogood*.

- **ATMSs**
  O rótulo de um nó é um conjunto de conjuntos mínimos de nós que devem ser acreditados para o nó em causa ser acreditado.

  Acreditamos nos nós que tenham pelo menos uma justificação válida, isto é, que tenham pelo menos um conjunto de suposições do seu rótulo contido no contexto.

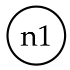
  As contradições são resolvidas evitando que existam rótulos que contenham algum nogood.

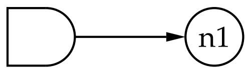
  Os nogoods são os conjuntos que estariam no rótulo do nó contradição e correspondem a conjuntos de nós que nunca devem estar contidos no contexto.

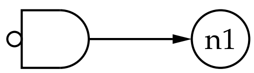
  Embora tenha mais dificuldades em lidar com justificações não-monótonas (o ATMS dado nas aulas nem o consegue fazer), torna muito mais simples a consideração de múltiplas alternativas.

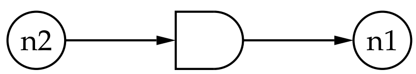
Nota: As duas figuras que estão neste resumo foram passadas para computador pela Elsa Luis Teixeira.

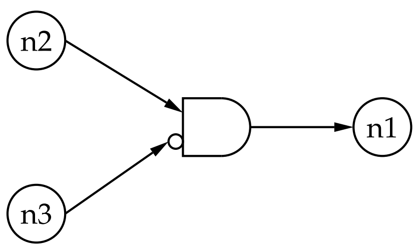
Exercício 8.1
Rotule as redes de dependências seguintes usando um JTMS, tendo em atenção que cada rede pode ter zero, uma ou mais rotulações possíveis. Se alguma delas tiver mais do que uma rotulação, deve mostrá-las todas.

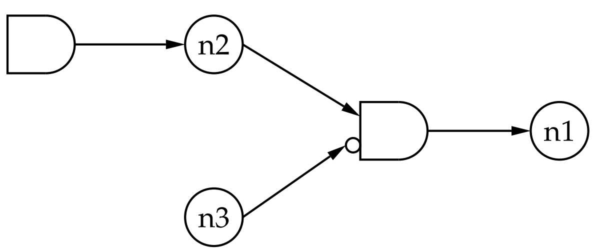
1.


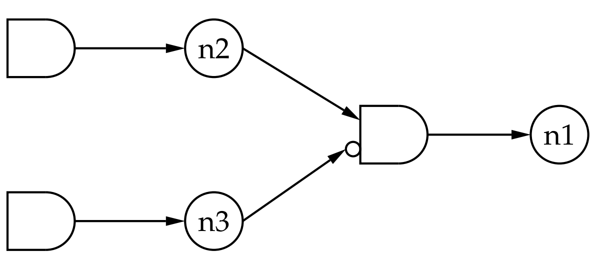
2.


3.


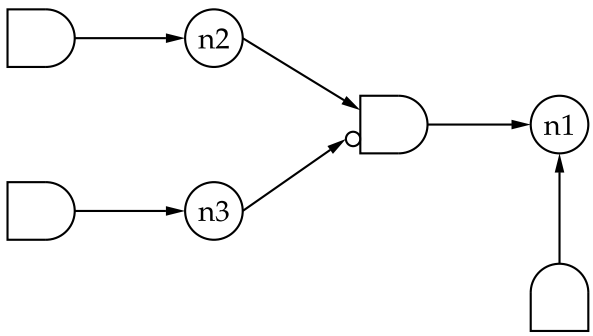
4.


5.


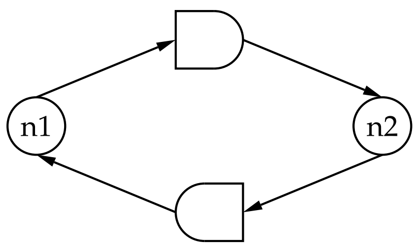
6.


7.


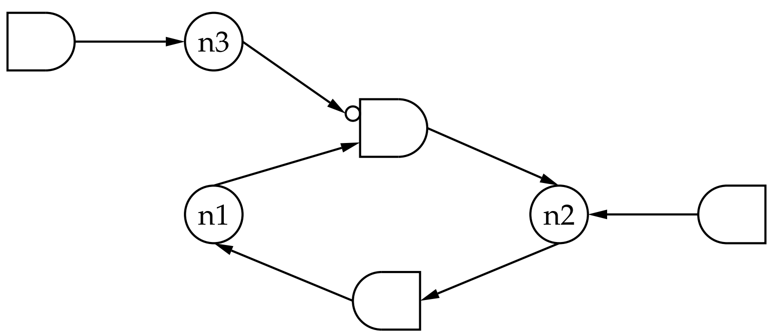
8.



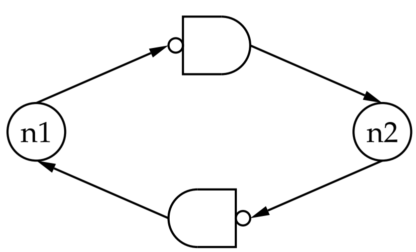
9.



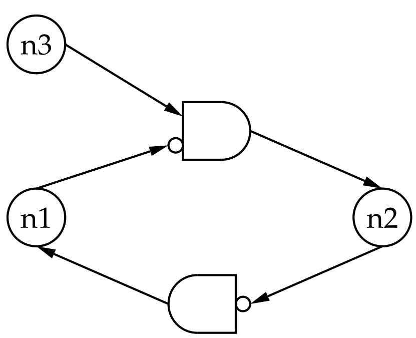
10.



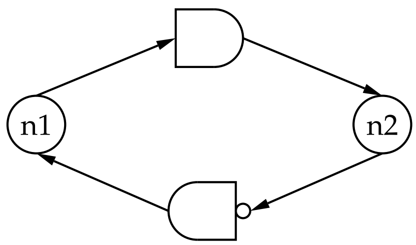
11.



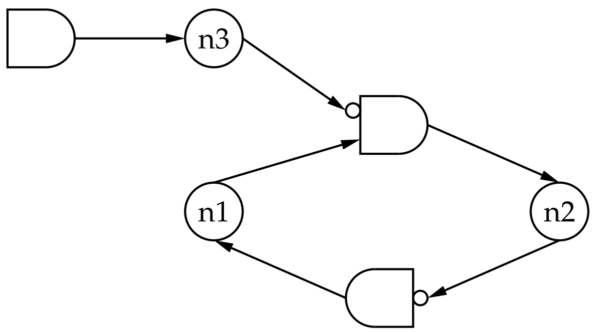
12.



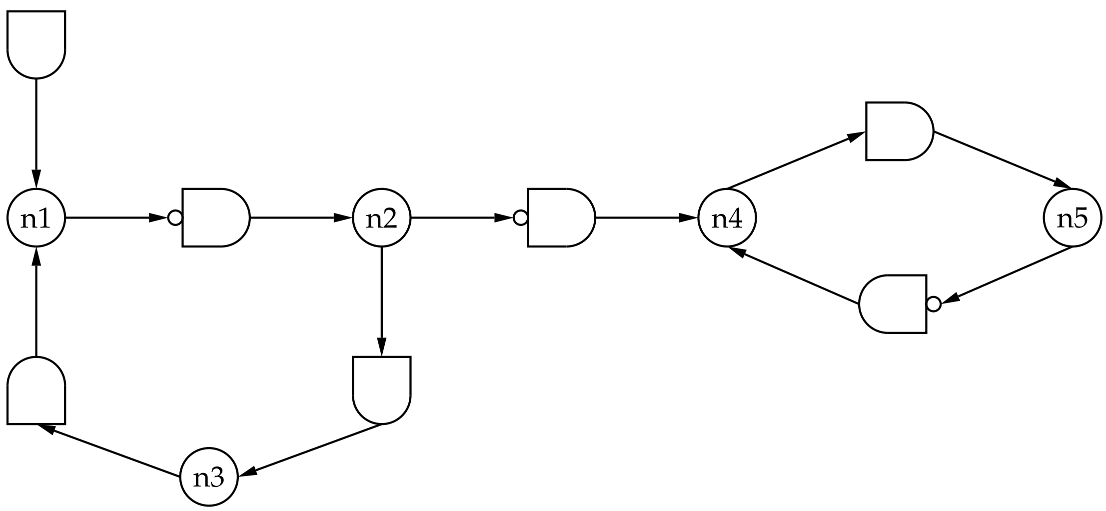
13.



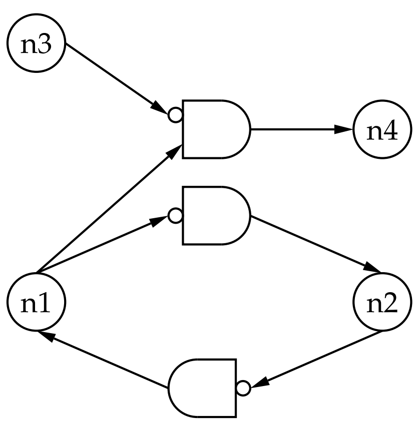
14.



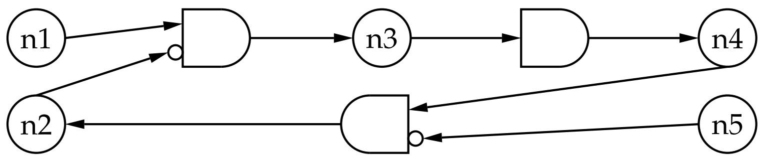
15.



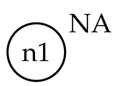
16.

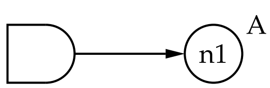


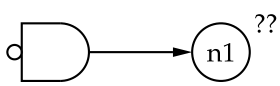
17.

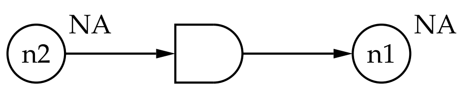


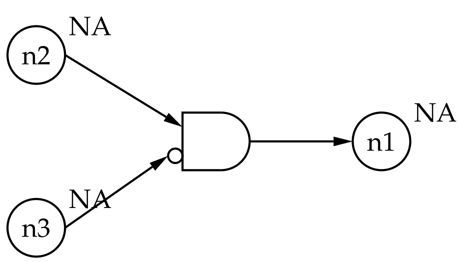
Resposta:

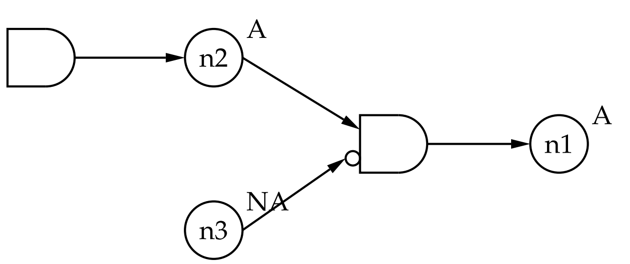
1. 

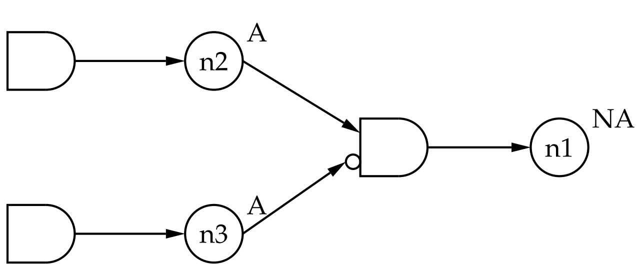
2. 

3. 

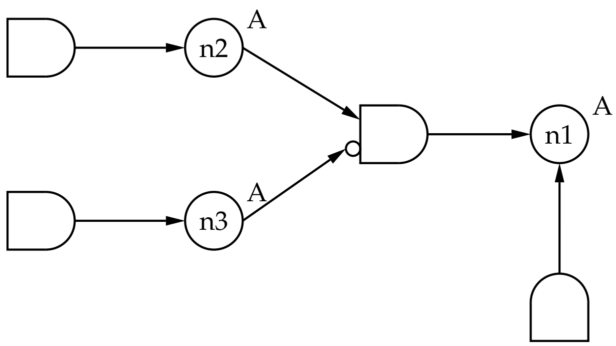
4. 

5. 

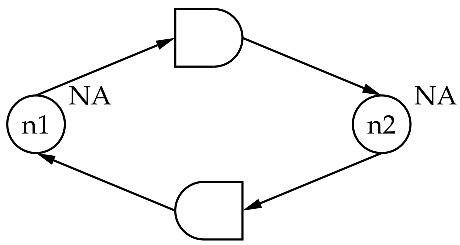
6. 

7. 

8.

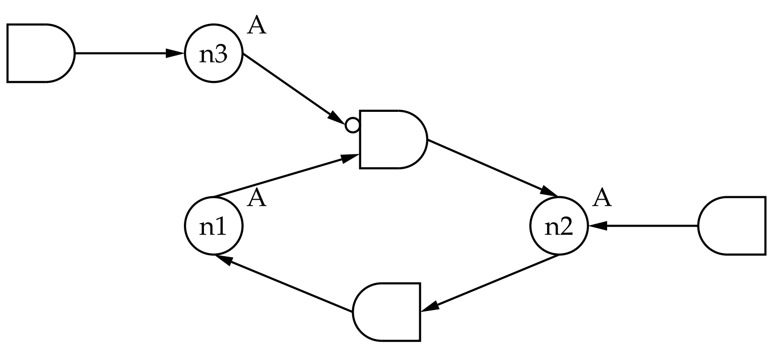


9.



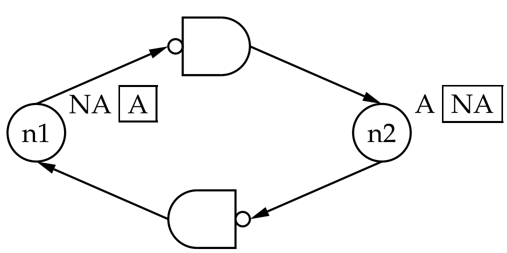
Os ciclos monótonos atravessam apenas justificações monótonas e em geral têm apenas uma rotulação estável em que nenhum nó é acreditado.

10.



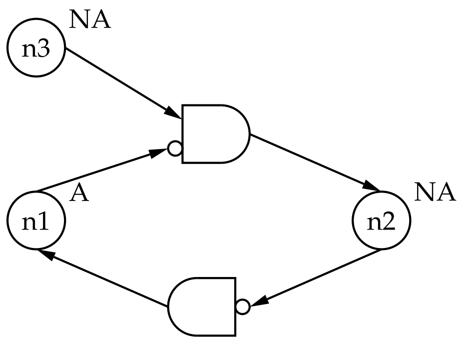
Mas . . . Este é um ciclo monótono em que os nós são acreditados.

11.



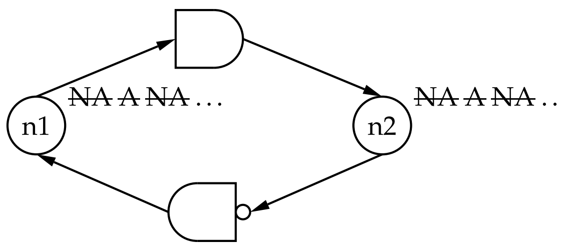
Os ciclos pares atravessam um número par de justificações não monótonas e em geral têm mais do que uma rotulação estável. Se começarmos a rotular por n1 temos a rotulação normal e se começarmos a rotular por n2 temos a rotulação dentro do rectângulo. Não esquecer que cada rotulação atribui um rótulo a cada nó.

12.



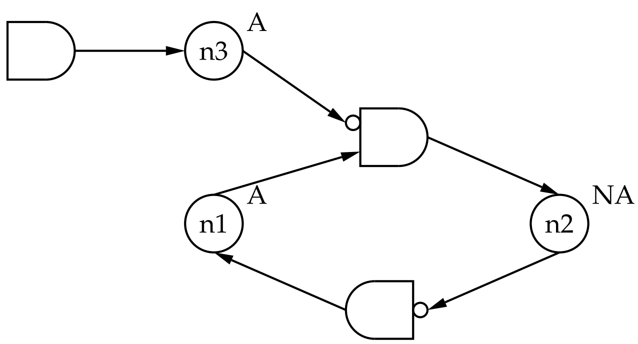
Mas . . . Este é um ciclo par que tem apenas uma rotulação estável.

13.



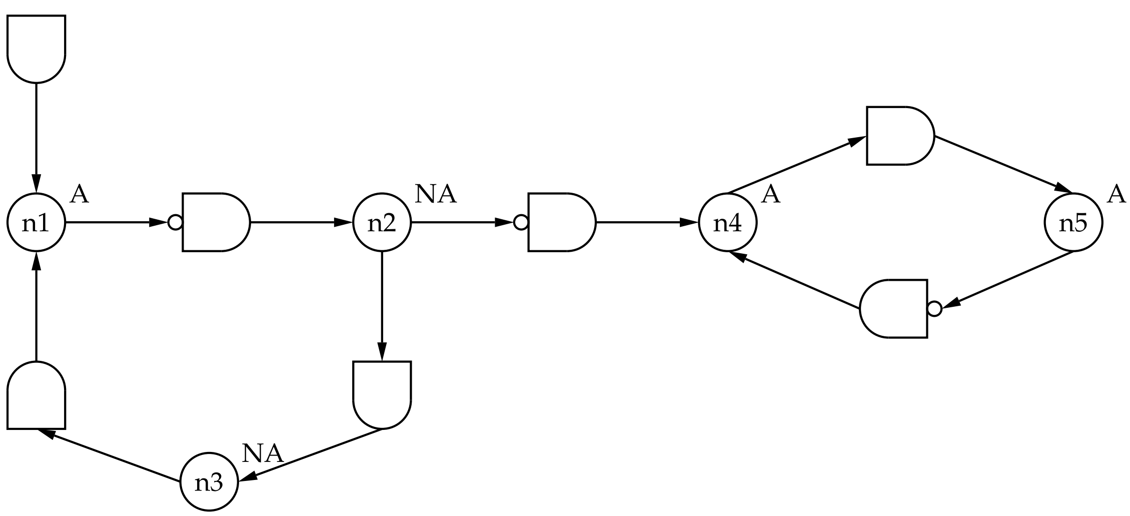
Os ciclos ímpares atravessam um número ímpar de justificações não monótonas e em geral não têm nenhuma rotulação estável.

14.



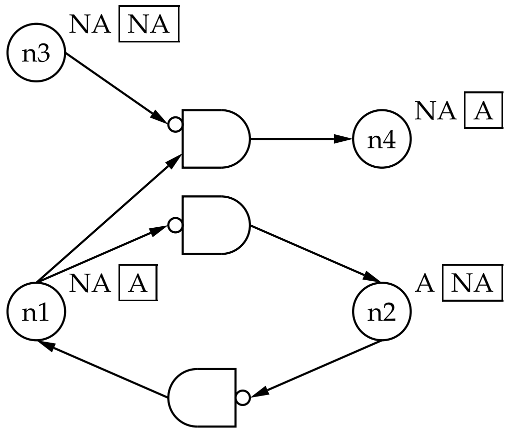
Mas . . . Este é um ciclo ímpar que tem uma rotulação estável.

15.



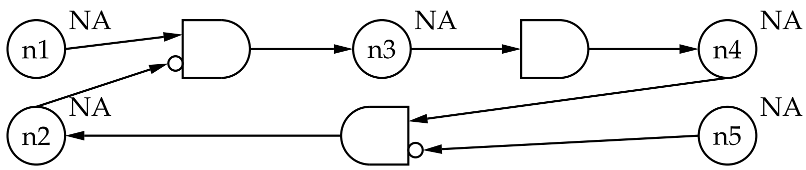
Esta rede tem dois ciclos ímpares mas continua a ter uma rotulação estável.

16.



Esta rede tem duas rotulações estáveis: se começarmos a rotular por n1 temos a rotulação normal e se começarmos a rotular por n2 temos a rotulação dentro do rectângulo.

17.



Nesta rede, não vamos começar a rotular o ciclo por qualquer nó pois sabemos que a justificação para o nó n3 não vai ser válida independentemente do rótulo de n2, uma vez que o nó n1 não é acreditado.

Exercício 8.2

Quais as redes do exercício anterior que consegue representar usando algum ATMS conhecido? Porquê?

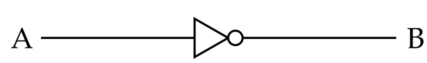
Resposta:

O ATMS de de Kleer dado nesta disciplina apenas permite representar justificações monótonas, ou seja, justificações sem antecedentes não monótonos.

Assim, só conseguimos usar este ATMS para representar as redes que só têm justificações monótonas, nomeadamente os pontos 1, 2, 4 e 9.

Exercício 8.3

Represente o seguinte circuito lógico usando um ATMS.



Resposta:

Podemos tentar representar "directamente" o circuito usando um ATMS, mas para representar o NOT temos que usar uma justificação não monótona, o que não é permitido no ATMS de de Kleer, que só tem justificações monótonas.

Para além disso, não conseguimos representar com um nó (que pode ser acreditado ou não acreditado) os três valores que podem estar num determinado fio. Ou seja, o nó *A* ser acreditado significa que o fio A tem o valor 0, 1 ou desconhecido?

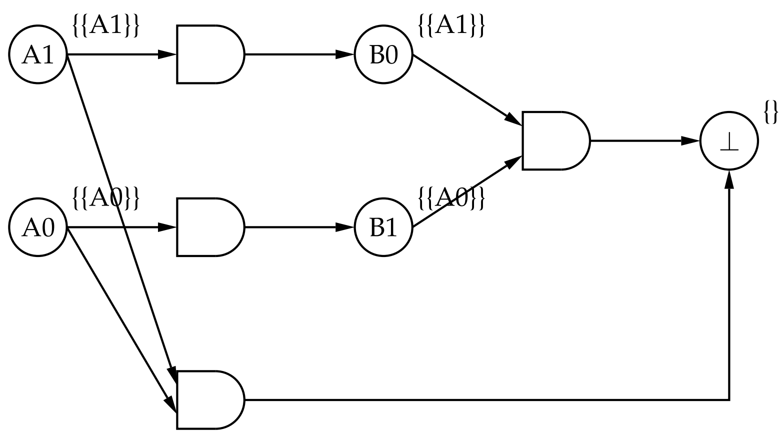
Por estas razões, precisamos de ter dois nós para representar cada fio: um para representar que o fio tem o valor 0 e outro para representar que o fio tem o valor 1.

Convém também não esquecer que um fio não pode ter simultaneamente os valores 0 e 1, por isso é necessário representar o nó contradição (⊥) e encontrar os nogoods.

Alguns pontos importantes a considerar na construção da rede do ATMS são:

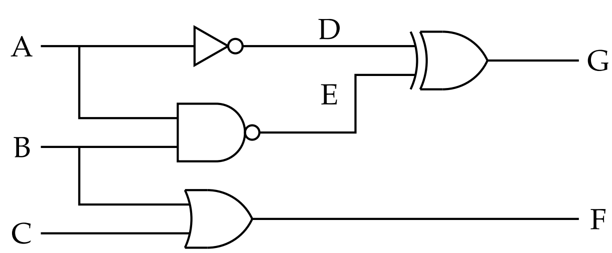
- Os rótulos dos nós contêm apenas conjuntos mínimos de nós. Por isso, depois de determinados os rótulos para os nós da rede, é necessário eliminar os conjuntos que não são mínimos.
- O rótulo do nó ⊥ é sempre {}, pois é a única forma que temos de garantir que ele nunca será acreditado.
- O conjunto dos nogoods é calculado como se estivéssemos a calcular o rótulo para o nó ⊥. Convém reparar que, por o conjunto dos nogoods corresponder a um rótulo, também contém apenas conjuntos mínimos de nós.
- Os rótulos não podem conter nogoods. Por isso, é necessário eliminar os nogoods de cada um dos rótulos anteriormente calculados.

A nova representação encontra-se na figura seguinte. Os rótulos dos nós podem ser representados junto aos nós ou numa tabela à parte, para não complicar demais a figura. Não esquecer de indicar o rótulo do nó contradição e o conjunto dos nogoods.



Nogoods:	{ {A0, A1}, (dos nós A0 e A1) ~~{A1, A0}~~} (dos nós B0 e B1)

Exercício 8.4
Considere o seguinte circuito lógico:

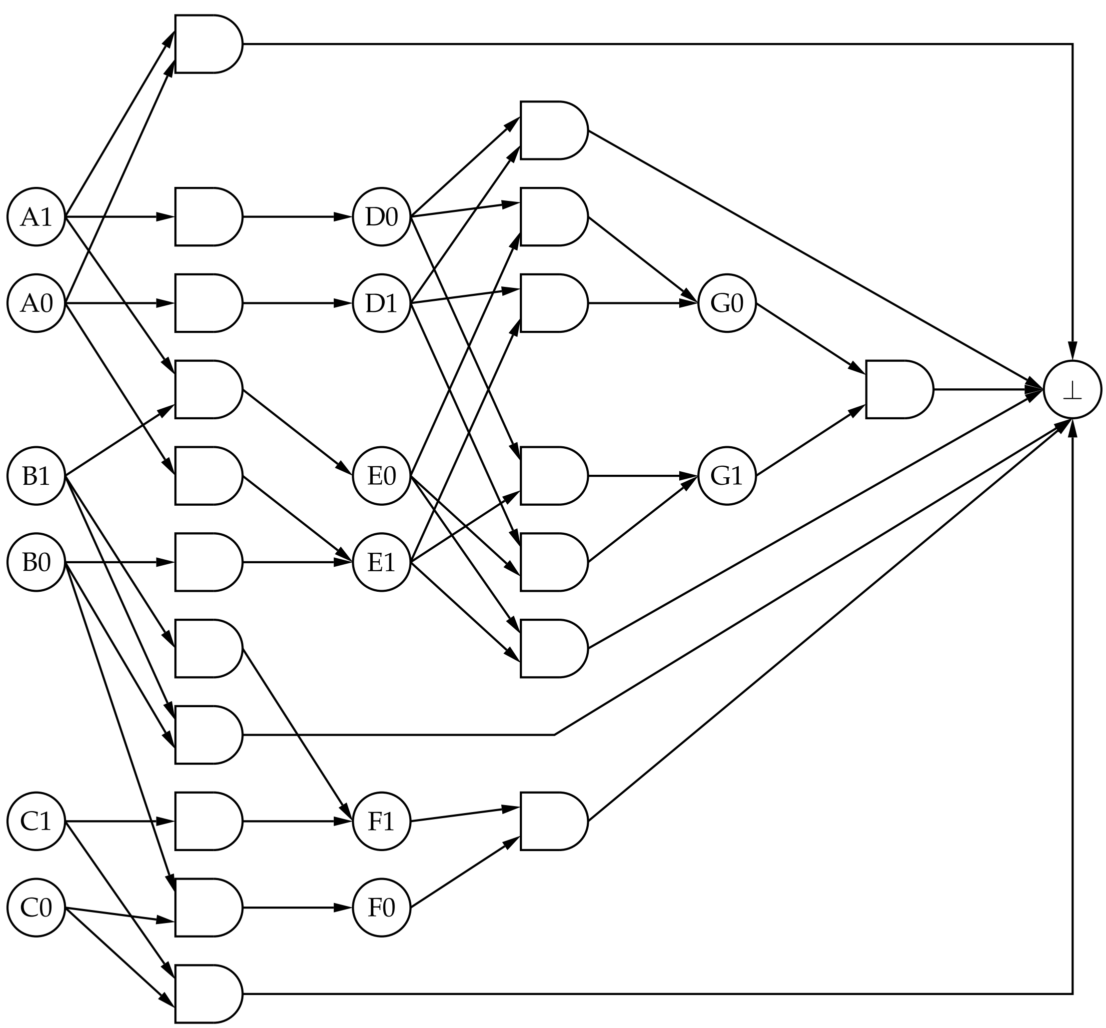


1. Represente-o usando um ATMS.

2. Se $A = 1$, $B = 0$ e $C = 0$, quais são os valores das saídas F e G?
3. E se $A = 0$ e $C = 1$?
4. E se $C = 1$?
5. Quais os valores que têm que ter as entradas A, B e C para ambas as saídas terem o valor 1?
6. E para terem ambas o valor 0?
7. E para pelo menos uma delas ter o valor 1?
8. Quais os valores que têm que ter as entradas A, B e C para E ter o valor 1?
9. E para $E = 0$ e $F = 0$?
10. Se $A = 1$, $B = 0$ e $C = 0$, quais são os fios que têm o valor 1?

Resposta:

1.



Nó	Rótulo
A1	{{A1}}
A0	{{A0}}
B1	{{B1}}
B0	{{B0}}
C1	{{C1}}
C0	{{C0}}
D1	{{A0}}
D0	{{A1}}
E1	{{A0},{B0}}
E0	{{A1, B1}}
F1	{{B1},{C1}}
F0	{{B0, C0}}
G1	{~~{A1, A0}~~, {A1, B0}, ~~{A0, A1, B1}~~} ={{A1, B0}}
G0	{{A1, ~~A1~~, B1},{A0, ~~A0~~}, ~~{A0, B0}~~} = {{A1, B1},{A0}}
$\perp$	{}

Nogoods:	{ {A0, A1},	(dos nós A0 e A1)
	{B0, B1},	(dos nós B0 e B1)
	{C0, C1},	(dos nós C0 e C1)
	~~{A1, A0}~~,	(dos nós D0 e D1)
	~~{A1, B1, A0}~~,~~{A1, B1, B0}~~,	(dos nós E0 e E1)
	~~{B1, B0, C0}~~,~~{C1, B0, C0}~~,	(dos nós F0 e F1)
	~~{A1, B1, A1, B0}~~,~~{A0, A1, B0}~~}	(dos nós G0 e G1)

2. Nestas condições, Contexto=$\{A1, B0, C0\}$.
   Acreditamos nos nós que tenham pelo menos uma justificação válida, isto é, que tenham pelo menos um conjunto de suposições do seu rótulo contido no contexto.
   Queremos saber, dos quatro nós que representam as duas saídas, quais é que são acreditados. Neste caso, acreditamos em $F0$ e em $G1$. Assim, as saídas têm os valores $F = 0$ e $G = 1$.

3. Contexto=$\{A0, C1\}$.
   Dos quatro nós que representam as duas saídas, $F1$ e $G0$ têm um dos conjuntos de suposições do seu rótulo contido no contexto. Assim, sabemos que a saída $F = 1$ e a saída $G = 0$.

4. Contexto=$\{C1\}$.
   Dos quatro nós que representam as duas saídas, só $F1$ é que tem um dos conjuntos de suposições do seu rótulo contido no contexto. Assim, sabemos que a saída $F = 1$ mas não sabemos qual o valor da saída G.

5. Para ambas as saídas terem o valor 1, temos que acreditar simultaneamente em $F1$ e $G1$, ou seja, temos que combinar os seus rótulos e determinar em que contextos é que eles podem estar contidos. Assim, temos que combinar $\{\{B1\}, \{C1\}\}$ com $\{\{A1, B0\}\}$ e ver quando é que podemos ter ambos simultaneamente: $\{\{B1, A1, B0\}\}$ contém um nogood, por isso não pode ser contexto; $\{\{C1, A1, B0\}\}$ pode ser contexto.

   Para ambas as saídas terem o valor 1, as entradas devem ter os valores $A = 1$, $B = 0$ e $C = 1$.

6. Para ambas as saídas terem o valor 0, temos que acreditar simultaneamente em $F0$ e $G0$, ou seja, temos que combinar $\{\{B0, C0\}\}$ com $\{\{A1, B1\}, \{A0\}\}$, obtendo $\{\{B0, C0, A1, B1\}\}$ que contém um nogood e $\{\{B0, C0, A0\}\}$.

   Para ambas as saídas terem o valor 0, as entradas devem ter os valores $A = 0$, $B = 0$ e $C = 0$.

7. Para pelo menos uma das saídas ter o valor 1, temos que acreditar em $F1$ ou $G1$ ou em ambas, ou seja, temos que combinar disjuntivamente $\{\{B1\}, \{C1\}\}$ com $\{\{A1, B0\}\}$.

   Para pelo menos uma das saídas ter o valor 1, as entradas devem ter os valores
   $A = 1$ e $B = 0$, independentemente do valor de C ou
   $B = 1$, independentemente dos valores de A e de C ou
   $C = 1$, independentemente dos valores de A e de B.

8. Para E ter o valor 1, ou seja, para acreditarmos no nó $E1$,um dos conjuntos de suposições do seu rótulo tem que estar contido no contexto, ou seja, ou o contexto contém $\{A0\}$ ou o contexto contém $\{B0\}$. Assim, as entradas devem ter os valores
   $A = 0$, independentemente dos valores de B e de C ou
   $B = 0$, independentemente dos valores de A e de C.
9. Neste caso, é impossível porque ...
10. Se $A = 1$, $B = 0$ e $C = 0$, o contexto é $\{A1, B0, C0\}$, logo, acreditamos nos nós $A1$, $B0$, $C0$, $D0$, $E1$, $F0$ e $G1$.

    Neste caso, os fios que têm o valor 1 são: A, E e G.

Exercício 8.5

Explique qual a diferença entre um nó $n1$ que tem o rótulo $\{\}$ e um nó $n2$ que tem o rótulo $\{\{\}\}$ no ATMS de deKleer.

Resposta:

No ATMS de deKleer, os rótulos dos nós são conjuntos de nós que têm que ser acreditados (isto é, estarem contidos no contexto) para o nó ser acreditado.

O nó $n1$ que tem o rótulo $\{\}$ nunca vai ter nenhum conjunto de nós do seu rótulo contido no contexto, e por isso nunca poderá ser acreditado. Um exemplo deste tipo de nós é o nó contradição.

O nó $n2$ que tem o rótulo $\{\{\}\}$ vai ter sempre um conjunto de nós do seu rótulo contido no contexto, e por isso vai ser sempre acreditado. Um exemplo deste tipo de nós é um nó que represente um teorema, que é sempre verdadeiro.

9 Redes Semânticas — SNePS

Sumário:

- Representação de informação em SNePS e comparação com a lógica de primeira ordem.

Resumo:

- Um nó representa um conceito intensional.
- Princípio da unicidade — cada nó representa um único conceito e cada conceito é representado por um único nó.

  Na realidade, a implementação do SNePS só força o princípio da unicidade para nós que não dominem nenhuma variável. Pode-se considerar (embora isto não esteja correcto) que, em termos globais da rede, cada variável é uma variável nova, por isso é que o SNePS não consegue impor este princípio.
- Os nós podem ser atómicos ou estruturados (moleculares):
  - Atómicos — não dominam nenhum outro nó.
    - Nós atómicos constantes (ou nós base) — representam conceitos não estruturados. Graficamente são representados por rectângulos com o seu nome.
    - Nós atómicos variáveis (ou nós variáveis) — representam variáveis. Graficamente são representados por círculos e o nome começa por V.
  - Estruturados — dominam pelo menos um outro nó.
    - Nós estruturados constantes (ou nós moleculares) — representam proposições ou conceitos estruturados e não dominam variáveis livres. Graficamente são representados por círculos e o seu nome começa por M. Podem ser acreditados ou não acreditados.
    - Nós estruturados não constantes (ou nós padrão) — dominam pelo menos um nó correspondente a uma variável livre. Graficamente são representados por círculos e o seu nome começa por P. Não faz sentido serem acreditados nem negados, por dominarem variáveis livres.
  - Os nós não se podem dominar a si próprios, isto é, não há ciclos compostos apenas por arcos directos.
- Os arcos representam relações não conceptuais entre conceitos. Para cada arco directo, existe o arco inverso, que não é representado explicitamente.
  - Arcos pré-definidos — os únicos arcos pré-definidos em SNePS são os arcos usados para representar as conectivas lógicas, para permitir efectuar inferência: `ant, &ant, cq, arg, min, max, forall`.
  - Arcos definidos pelo utilizador — servem para o utilizador poder definir os case-frames que vai usar na sua representação. Os mais usados são:
    - `mem / class`
    - `sub / super`

        * `obj / pp / valor`
        * `ag / verbo / obj`
        * `rel / arg1 / ...  / argn`
        * `fun / arg1 / ...  / argn`

- Conectivas:
    - Implicação conjuntiva: $\{A_1, \ldots, A_n\} \wedge\!\!\Rightarrow \{C_1, \ldots, C_n\}$
    - Implicação disjuntiva: $\{A_1, \ldots, A_n\} \vee\!\!\Rightarrow \{C_1, \ldots, C_n\}$
    - E/Ou: ${}_n\bowtie_i^j\{A_1, \ldots, A_n\}$
    - Thresh: ${}_n\Theta_i^j\{A_1, \ldots, A_n\}$
- Quantificadores:
    - Universal: $\forall_x[P(x)]$
    - Existencial: $\exists_x[P(x)]$
    - Existencial numérico: ${}_n\exists_i^j$

  Só o universal é que está implementado. O existencial é representado através de funções e constantes de Skolem, dependendo de estar ou não dentro do âmbito de quantificadores universais.
  Skolemização:
  $\exists(x)[P(x)] \rightsquigarrow P(Sk1)$ — quando a variável quantificada existencialmente não está dentro do âmbito de nenhum quantificador universal, é substituida por uma constante de Skolem, ou seja, por uma constante que não apareceu antes na rede.
  $\forall(y)\exists(x)[P(x,y)] \rightsquigarrow \forall(y)[P(Skfun1(y),y)]$ — quando a variável quantificada existencialmente está dentro do âmbito de algum quantificador universal, é substituida por uma função de Skolem, que não apareceu antes na rede e que tem como argumentos as variáveis quantificadas universalmente em cujo âmbito ela se encontra.
- Aspectos a lembrar em qualquer exercício:
    - Princípio da unicidade. Se repetirmos nós, dizer que é apenas para facilitar a representação, mas que na realidade são o mesmo.
    - No mesmo nó, não misturar arcos pré-definidos com arcos definidos pelo utilizador.
    - No mesmo nó, não misturar arcos de case-frames diferentes (por exemplo, *ant* e *cq* com *min* e *max* ou *mem* ou *sub*).
    - No mesmo nó, não misturar quantificadores diferentes. Se o existencial for representado através da Skolemização, isto deixa de ser possível.
    - Não há nós nem arcos sem nome.
    - Não há nós não dominados não acreditados.
    - Só saem setas de nós *M* ou *P* (nunca de nós atómicos).
    - Não faz sentido negar nem acreditar nós atómicos nem um nós padrão.
    - UVBR — Unique Variable Binding Rule: se numa regra há variáveis com nomes diferentes, então vão obrigatoriamente emparelhar com constantes diferentes.
- Inferência

  - Baseada em nós
  - Baseada em caminhos

SNePSLOG
Linguagem de interacção com o SNePS parecida com a LPO.

- É bom fazer um "dicionário" com os predicados e funções e a ordem dos argumentos.
- Variáveis com nomes errados são constantes.
- Ficheiro com as fórmulas em SNePSLOG deve ter no início `clearkb`
- Comentários começam por `;`
- Não se usa `[ ]`
- Conectivas lógicas: `and`, `or`, `=>`, `~`
- Quantificadores: `all(x,y)(...)` ou `exists(x,y)(...)`.
  O existencial é representado internamente pelo SNePSLOG usando Skolemização.
- `??` pergunta sem inferência
- `?` pergunta com inferência
- `?var` variável, só em queries
- `^(setf *infertrace nil)`
- `^(setf *infertrace t)`
- Utilização, depois de entrar no Lisp:

```
(load ''home/cadeiras/rc/Sneps/load-SNePS'')
(snepslog)
demo
8
''home/cadeiras/rc/man.snlog''
<enter>

...

lisp   (para sair do SNePSLOG)
(quit) (para sair do Lisp)
```

SNePSUL
Linguagem de interacção com o SNePS usando explicitamente os nós e os arcos da rede semântica.

- É bom fazer um "dicionário" com os case-frames utilizados.
- `(describe *nodes)` — descreve todos os nós da rede

- `(list-nodes)`
- `(demo “ficheiro”)`
- `(build {relation nodeset})` — cria um nó sem acreditar nele (só para “inner-nodes”).
- `(assert {relation nodeset})` — cria um nó acreditado.
- `(add {relation nodeset})` — cria um nó acreditado e faz inferência para a frente com ele.
- `(resetnet [reset-relations?])`
- `(define {relationset})`
- and-entailment — `(assert &ant (A1, ..., An) cq (C1, ..., Cm))`
- or-entailment — `(assert ant (A1, ..., An) cq (C1, ..., Cm))`
  Mais eficiente que o anterior, deve ser usado quando só houver um antecedente.
- and-or — `(assert min i max j arg (A1, ..., An))`
- quantificador — usa-se em conjunto com as conectivas lógicas. Por exemplo, `(assert forall (x1, ..., xp) &ant (A1(x1, ..., xp), ..., An(x1, ..., xp)) cq (C1(x1, ..., xp), ..., Cm(x1, ..., xp)))`.
  Primeira ocorrência da variável `$var` subsequentes `*var`.
- Utilização, depois de entrar no Lisp:

```
(load ‘‘home/cadeiras/rc/Sneps/load-SNePS’’)
(sneps)

...

lisp   (para sair do SNePS)
(quit) (para sair do Lisp)
```

Exercício 9.1
Represente graficamente em SNePS as seguintes afirmações. Para facilitar a numeração dos nós, considere que cada afirmação é representada numa “rede nova”.

1. O Bit é um cão preto.
2. O Nuno ou é polícia ou é ladrão, mas não os dois simultaneamente.
3. Nenhum tubarão é pessoa.
4. Nem todos os tubarões são carnívoros.
5. Todos os tubarões têm uma cauda.
6. Qualquer tubarão que esteja vivo pode nadar e morder.
7. Só os homens e as mulheres é que são pessoas.
8. Se alguém consegue esquiar então o Nuno também consegue.
9. Tudo o que alguém consegue fazer o Nuno também consegue.
10. O Pai da Maria é casado com a Mãe da Maria.

Resposta:

Para representação gráfica, ver as folhas escritas à mão.

Exercício 9.2
(JPM) Represente graficamente em SNePS as seguintes afirmações:

1. O João acredita que sabe a idade da Maria.
2. O João não sabe a idade da Maria.
3. A Maria tem 22 anos.

Resposta:

Para representação gráfica, ver as folhas escritas à mão.

Exercício 9.3
(JPM) Represente graficamente em SNePS as seguintes propriedades de relações:

1. Transitividade.
2. Simetria.
3. Reflexividade.
4. Equivalência. Uma relação de equivalência é uma relação simétrica, reflexiva e transitiva.

Resposta:

Para representação gráfica, ver as folhas escritas à mão.

Exercício 9.4
Considere a seguinte informação acerca de mamíferos. Com base nela e no conhecimento (de senso comum) que tem acerca do mundo, represente-a graficamente em SNePS.

Existem aproximadamente 4500 espécies de mamíferos no mundo, e todos eles respiram oxigénio do ar. Os mamíferos também são caracterizados por cuidarem das suas crias enquanto bebés e por as alimentarem de leite materno.

As fêmeas dos mamíferos desenvolvem as crias dentro do ventre e quando chegam ao fim da gestação podem ter entre 1 e 27 crias.

Uma das excepções a esta regra são os monotremas (que incluem os ornitorrincos e as equidnas), que põem ovos e os incubam, para as crias se desenvolverem. Os machos dos ornitorrincos são também caracterizados, entre outras coisas, por terem espigões venenosos nas patas traseiras.

Os humanos, em contrapartida, podem ser caracterizados pelo índice de massa corporal: peso em kilos a dividir pela altura em metros ao quadrado. Este valor é interpretado da seguinte forma:

IMC inferior 18,5	Peso abaixo do normal
IMC de 18,5 a 25	Peso Normal
IMC de 25 a 29,9	Excesso de Peso
IMC superior 30	Obesidade

O Luís é um humano que pesa 90Kg e mede 1,90m. O Zé é uma cria do Luis.

A OF é um ornitorrinco fêmea, o OM é um ornitorrinco macho e a OC é uma cria de OF.

Há um mamífero (chamemos-lhe Flip) que ou é um ornitorrinco ou é uma equidna.

Resposta:

Para representação gráfica, ver as folhas escritas à mão.

Ver se faz sentido incluir algumas das perguntas que aparecem no fim das próximas resoluções no exercício da representação gráfica.

Ver que perguntas é que podem ser incluidas para a inferência, nomeadamente para a inferência baseada em caminhos.

Resolução em SNePSLOG

```
clearkb

^ (setf *infertrace* nil)

AproxIgual(cardin(EspMam), 4500)

all(x,y)((Pertence(x, EspMam) and TemEsp(y, x)) =>
         Mam(y))
```

```
all(x)(Mam(x) => Respira(x, OxigenioAr))

all(x,y)((Mam(x) and Cria(y, x)) =>
         (Mam(y) and (Bebe(y) => Cuida(x, y))))

all(x,y)((Mam(x) and Femea(x) and Cria(y, x)) =>
         Alimenta(x, y, LeiteMaterno))

all(x,y)((Mam(x) and Femea(x) and
          Cria(y, x) and ~Monotrema(x)) =>
         DesenvolveDentro(x, y, ventre(x)))
; esta regra nao pode ser aplicada a outros mamiferos,
; a nao ser que seja explicitamente dito
; que nao sao monotremas

all(x,y)((Mam(x) and Femea(x) and Gestacao(y, x)) =>
         Entre(numCrias(y), 1, 27))

all(x)(Monotrema(x) => Mam(x))

all(x)(Ornitorrinco(x) => Monotrema(x))

all(x)(Equidna(x) => Monotrema(x))

all(x)((Monotrema(x) and Femea(x)) =>
       (Poe(x, ovos(x)) and Incuba(x, ovos(x))))

all(x,y)((Monotrema(x) and Femea(x) and Cria(y, x)) =>
         ~DesenvolveDentro(x, y, ventre(x)))

all(x)((Ornitorrinco(x) and ~Femea(x)) =>
       (Tem(x, espigao(x)) and Venenoso(espigao(x)) and
        Localizacao(espigao(x), pataTraseira(x))))

all(x)(Humano(x) => Mam(x))

all(x)(Humano(x) =>
       Igual(imc(x), divisao(peso(x),
                     quadrado(altura(x)))))

all(x)((Humano(x) and Menor(imc(x), 18p5)) =>
       TipoPeso(peso(x), PesoInferiorNormal))

all(x)((Humano(x) and Entre(imc(x), 18p5, 25)) =>
       TipoPeso(peso(x), PesoNormal))

all(x)((Humano(x) and Entre(imc(x), 25, 29p9)) =>
       TipoPeso(peso(x), PesoExcessivo))

all(x)((Humano(x) and Maior(imc(x), 30)) =>
       TipoPeso(peso(x), Obesidade))

;;;;;;;;;;;;;;;;;;;;
; Factos e perguntas

Ornitorrinco(OM) and ~Femea(OM)
```

```
Tem(OM, ?x)?

Ornitorrinco(OF) and Femea(OF)

Cria(OC, OF)!

?p(OC)?

Poe(?x, ?y)?

Humano(Luis)!

Igual(peso(Luis), kilos(90))

Igual(altura(Luis), metros(1p9))

Cria(Ze, Luis)!

Mam(?x)?

Respira(Ze, ?x)?

Alimenta(Luis, Ze, LeiteMaterno)?

~Femea(Luis)

Alimenta(Luis, Ze, LeiteMaterno)?

Humano(Eva) and Femea(Eva) and Cria(Ze, Eva)!

Cuida(?x, Ze)?

Bebe(Ze)!

Cuida(?x, Ze)?

DesenvolveDentro(?x, ?y, ?z)?

all(x)(Humano(x) => ~ Monotrema(x))!

DesenvolveDentro(?x, ?y, ?z)?

Alimenta(?x, ?y, ?z)?
```

Resolução em SNePSUL

```
(resetnet)

^ (setf *infertrace* nil)

(define tempp pp valor)

(describe (assert tempp EspMam pp cardin valor CEM))

(describe (assert tempp CEM pp AproxIgual valor 4500))
```

```
(define mem class)

(describe (assert
  forall ($vx $vy)
  &ant ((build tempp *vx pp Pertence valor EspMam)
         (build tempp *vy pp TemEsp valor *vx))
  cq (build mem *vy class Mam)))

(describe (assert
  forall $vx
  ant (build mem *vx class Mam)
  cq (build tempp *vx pp Respira valor OxigenioAr)))

(define ag rel obj)

(describe (assert
  forall ($vx $vy)
  &ant ((build mem *vx class Mam)
        (build tempp *vx pp cria valor *vy))
  cq ((build mem *vy class Mam)
      (build ant (build mem *vy class Bebe)
             cq (build ag *vx
                       rel Cuidar
                       obj *vy)))))

(define obj1 obj2)

(describe (assert
  forall ($vx $vy)
  &ant ((build mem *vx class Mam)
        (build mem *vx class Femea)
        (build tempp *vx pp Cria valor *vy))
  cq (build ag *vx
            rel Alimentar
            obj1 *vy
            obj2 LeiteMaterno)))

(define fun arg1 arg2)

(describe (assert
  forall ($vx $vy)
  &ant ((build mem *vx class Mam)
        (build mem *vx class Femea)
        (build min 0 max 0
              arg (build mem *vx class Monotrema))
        (build tempp *vx pp Cria valor *vy))
  cq (build ag *vx
            rel DesenvolverDentro
            obj1 *vy
            obj2 (build fun VentreDe arg1 *vx))))

(describe (assert
  forall ($vx $vy)
  &ant ((build mem *vx class Mam)
        (build mem *vx class Femea)
```

```
      (build tempp *vx pp Gestacao valor *vy))
  cq (build ag (build fun NumCriasDe arg1 *vy)
            rel Entre obj1 1 obj2 27)))

(describe (assert
  forall $vx
  ant (build mem *vx class Monotrema)
  cq (build mem *vx class Mam)))

(describe (assert
  forall $vx
  ant (build mem *vx class Ornitorrinco)
  cq (build mem *vx class Monotrema)))

(describe (assert
  forall $vx
  ant (build mem *vx class Equidna)
  cq (build mem *vx class Monotrema)))

(describe (assert
  forall $vx
  &ant ((build mem *vx class Monotrema)
        (build mem *vx class Femea))
  cq ((build ag *vx rel Por
             obj (build fun OvosDe arg1 *vx)
                 = OvosDeX)
      (build ag *vx rel Incubar obj *OvosDeX))))

(describe (assert
  forall ($vx $vy)
  &ant ((build mem *vx class Monotrema)
        (build mem *vx class Femea)
        (build tempp *vx pp Cria valor *vy))
  cq (build min 0 max 0
            arg (build ag *vx
                       rel DesenvolverDentro
                       obj1 *vy
                       obj2 (build fun VentreDe
                                   arg1 *vx)))))

(describe (assert
  forall $vx
  &ant ((build mem *vx class Ornitorrinco)
        (build min 0 max 0
               arg (build mem *vx class Femea)))
  cq ((build tempp *vx
             pp Ter
             valor (build fun EspigaoDe arg1 *vx)
                   = EspDeX)
      (build tempp *EspDeX pp Venenoso)
      (build tempp *EspDeX
             pp Localizacao
             valor (build fun PataTraseiraDe
                          arg1 *vx)))))

(describe (assert
```

```
  forall $vx
  ant (build mem *vx class Humano)
  cq (build mem *vx class Mam)))

(describe (assert
  forall $vx
  ant (build mem *vx class Humano)
  cq (build
       rel Igual
       obj1 (build fun IMCDe arg1 *vx)
       obj2 (build
              fun DivisaoDe
              arg1 (build fun PesoDe
                          arg1 *vx)
              arg2 (build fun QuadradoDe
                          arg1 (build fun AlturaDe
                                      arg1 *vx))))))

(describe (assert
  forall $vx
  &ant ((build mem *vx class Humano)
        (build rel Menor
               obj1 (build fun IMCDe arg1 *vx)
               obj2 18p5))
  cq (build tempp (build fun PesoDe arg1 *vx)
            pp TipoPeso
            valor InferiorNormal)))

(describe (assert
  forall $vx
  &ant ((build mem *vx class Humano)
        (build ag (build fun IMCDe arg1 *vx)
               rel Entre
               obj1 18p5
               obj2 25))
  cq (build tempp (build fun PesoDe arg1 *vx)
            pp TipoPeso
            valor Normal)))

(describe (assert
  forall $vx
  &ant ((build mem *vx class Humano)
        (build ag (build fun IMCDe arg1 *vx)
               rel Entre
               obj1 25
               obj2 29p9))
  cq (build tempp (build fun PesoDe arg1 *vx)
            pp TipoPeso
            valor ExcessoPeso)))

(describe (assert
  forall $vx
  &ant ((build mem *vx class Humano)
        (build rel Menor
               obj1 30))
               obj2 (build fun IMCDe arg1 *vx)
```

```
  cq (build tempp (build fun PesoDe arg1 *vx)
            pp TipoPeso
            valor Obesidade)))



;;;;;;;;;;;;;;;;;;;;
; Factos e perguntas

(dc-set-pause t)

(describe (assert
  min 2 max 2
  arg ((build mem OM class Ornitorrinco)
       (build min 0 max 0
              arg (build mem OM class Femea)))))

(describe (deduce tempp OM pp Ter valor $vy))

(describe (add
  min 2 max 2
  arg ((build mem OF class Ornitorrinco)
       (build mem OF class Femea))))

(describe (add tempp OF pp Cria valor OC))

(describe (deduce ag $vx rel Por obj $vy))

(add mem Luis class Humano)

(add rel Igual
     obj1 (build fun PesoDe arg1 Luis)
     obj2 (build rel Unidade obj1 Kilos obj2 90))

(add rel Igual
     obj1 (build fun AlturaDe arg1 Luis)
     obj2 (build rel Unidade obj1 Metros obj2 1p90))

(describe (add tempp Luis pp Cria valor Ze))

(describe (deduce mem $vx class Mam))

(describe (deduce tempp Ze pp Respira valor $vx))

(describe (deduce ag Luis rel Alimentar
                  obj1 Ze obj2 LeiteMaterno))

(add min 0 max 0 arg (build mem Luis class Femea))

(describe (deduce ag Luis rel Alimentar
                  obj1 Ze obj2 LeiteMaterno))

(describe (add
  min 3 max 3
  arg ((build mem Eva class Humano)
       (build mem Eva class Femea)
```

```
       (build tempp Eva pp Cria valor Ze))))

(describe (deduce ag $vx rel Cuidar obj Ze))

(describe (add mem Ze class Bebe))

(describe (deduce ag $vx rel Cuidar obj Ze))

(describe (deduce ag $vx rel DesenvolverDentro
                  obj1 $vy obj2 $vz))

(add forall $vx
     ant (build mem *vx class Humano)
     cq (build min 0 max 0
               arg (build mem *vx class Monotrema)))

(describe (deduce ag $vx rel DesenvolverDentro
                  obj1 $vy obj2 $vz))

(describe (deduce ag $vx rel Alimentar
                  obj1 $vy obj2 $vz))
```

10 Sistemas de Enquadramentos — KEE

Sumário:

- Representação de conhecimento em KEE.
- Comparação com as várias lógicas estudadas e com o SNePS.

Resumo:

Os enquadramentos servem para representar hierarquias de conceitos. Algumas das noções importantes são:

- Base de conhecimento — conjunto de unidades que pertencem a um determinado domínio e que estão organizadas hierarquicamente (em uma ou mais hierarquias)
- Unidades — representam os objectos do domínio
  - Classe
  - Instância
- Atributos — representam as propriedades das unidades
  - De membro — são propagados para as subclasses e para as instâncias (neste caso passam a ser próprios). Ex: para a classe das aves, os atributos "número de patas" e "cor" seriam atributos de membro; o primeiro teria por omissão o valor 2 e a cor não teria valor por omissão, porque existem aves de muitas cores diferentes.
  - Próprios — pertencem a uma determinada unidade e não são propagados. Ex: na classe das aves, o atributo "maior ave"; todos os atributos das instâncias são atributos próprios.
- Facetas — são características dos atributos
  - Forma de herança
    - `OVERRIDE` — Aplica-se a atributos que só podem ter um valor. Quando não existe valor próprio, fica com o valor herdado do primeiro pai. Um valor mais específico (ou local) sobrepõe-se a um valor mais geral. Quando há conflito, seguem-se as superclasses pela ordem especificada (pelo utilizador).
    - `OVERRIDE.VALUES` — Similar ao anterior, mas aplica-se a atributos que podem ter mais que um valor.
    - `UNION` — União dos valores herdados (de todos os pais) com o valor local.
    - `UNIQUE` — Bloqueia a herança.
    - `METHOD` — Pode-se activar um procedimento para determinar o resultado da herança.
  - Classe de valores
    - `(ONE.OF <obj 1> ... <obj n>)`
      Um **ou mais** dos objectos indicados.

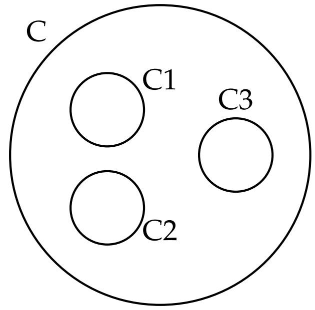
    * `(SUBCLASS.OF <classe>)`
    * `(MEMBER.OF <classe>)`
    * `(UNION <classe 1> ... <classe n>)`
    * `(INTERSECTION <classe 1> ... <classe n>)`
  - Cardinalidade
    * Máxima
    * Mínima
  - Valor
    * Valor pertencente à classe de valores
    * `UNKNOWN` ou — quando não se conhece o valor
    * `NIL` quando não tem valor

A linguagem TellAndAsk:

- Expressões básicas
  - `(SUBCLASS.OF <subclasse> <superclasse>)`
  - `(IN.CLASS <membro> <classe>)`
  - `(MEMBER.VALUE <nome atributo> <nome unidade> <valor>)`
  - `(OWN.VALUE <nome atributo> <nome unidade> <valor>)`
  - `(MEMBER.VALUE.IN.CLASS <nome atributo> <nome unidade> <classe>)`
  - `(OWN.VALUE.IN.CLASS <nome atributo> <nome unidade> <classe>)`
  - `(MEMBER.MIN.CARD <nome atributo> <nome unidade> <inteiro>)`
  - `(OWN.MIN.CARD <nome atributo> <nome unidade> <inteiro>)`
  - `(MEMBER.MAX.CARD <nome atributo> <nome unidade> <inteiro>)`
  - `(OWN.MAX.CARD <nome atributo> <nome unidade> <inteiro>)`
- Expressões compostas — são definidas à custa das expressões básicas, combinadas através da utilização de operadores lógicos (`AND`, `OR`, `NOT` e `EQUAL`)
- Comandos
  - `ASSERT`
  - `QUERY`
  - `RETRACT`

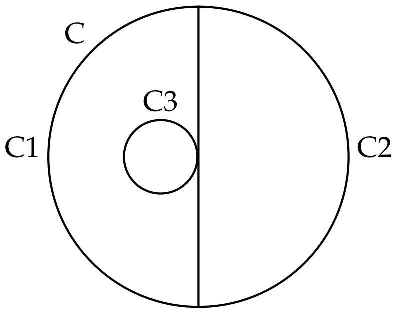
  Outros comandos para a construção de hierarquias vão ser introduzidos nos exemplos.
- Quando num `ASSERT` aparece o nome de uma unidade que ainda não existe, ela é criada automaticamente. O mesmo acontece com os atributos, desde que exista a unidade a que pertencem.
- Regras de produção: `(IF <condicao> THEN <accoes>)`
- Os vários tipos de decomposição:

- `DECOMPOSITION.DISJOINT` — $(C1 \cup C2 \cup C3) \subset C$, $(C1 \cap C2 \cap C3) = \{\}$.

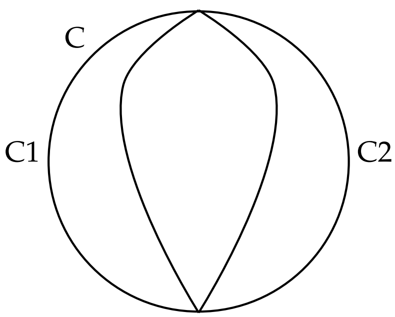
  

  Exemplo: $C = Mamiferos, C1 = Humanos, C2 = Morcegos, C3 = Baleias$.

- `DECOMPOSITION.COMPLETE` — $(C1 \cup C2 \cup C3) = C$.

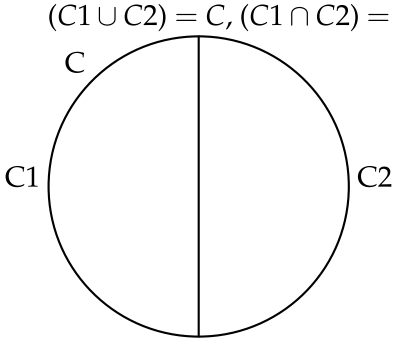
  

  Exemplo: $C = Inteiros, C1 = Impares, C2 = Pares, C3 = Primos$
  (existe um número primo que é par, o 2).

  Exemplo2: $C = Inteiros, C1 = Maioresque5, C2 = Menoresde10$

- `DECOMPOSITION.DISJOINT, DECOMPOSITION.COMPLETE` —
  $(C1 \cup C2) = C$, $(C1 \cap C2) = \{\}$.

  Exemplo: $C = Humanos, C1 = Homens, C2 = Mulheres$
  ou então $C = Inteiros, C1 = Impares, C2 = Pares$.

Representação para os enquadramentos a ser usada nas aulas e no exame[1]:

[1]Figura feita pela Carla Penedo.

KEE

Representação Gráfica

<Nome da unidade>

Superclasses:
Subclasses:
[DECOMPOSITION.DISJOINT]
[DECOMPOSITION.COMPLETE]
Instâncias:

Classe

Membro:

Instância

<Nome do atributo> ([MEMBER | OWN])
Valor:
Classe de valores:
[ONE.OF | SUBCLASS.OF | MEMBER.OF | INTERSECTION | UNION]
Cardinalidade máxima:
Cardinalidade mínima:
Tipo de herança:
[OVERRIDE | OVERRIDE.VALUES | UNION | UNIQUE | METHOD]

Atributo

<Nome do atributo> ([MEMBER | OWN])
Valor: *<Nome da função>*
Classe de valores: METHOD
Tipo de herança: METHOD

Atributo c/ Lig. Proc.

...

Num determinado enquadramento, é sempre necessário escrever, para além do seu nome, as suas superclasses, subclasses e instâncias directas ou então as classes de que é membro directo. Isto serve para indicar se o enquadramento corresponde a uma classe ou a uma instância e qual a sua relação com os outros enquadramentos. De notar que devem aparecer apenas as directas, porque as outras podem ser inferidas pelo KEE.

Quando há herança de atributos, em princípio devemos escrever só o que mudou. Se for tudo igual à superclasse, escrevemos apenas o nome e o tipo do atributo. Nas instâncias, é necessário lembrar que todos os atributos passam a `OWN`.

Quando se quer representar que uma determinada linha do enquadramento não tem qualquer valor, deve aparecer a palavra `NIL`.

Quando se quer representar que se desconhece o valor de uma determinada linha, pode aparecer apenas — (o correcto seria `UNKNOWN`).

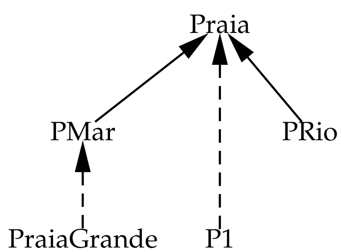
As funções e regras de produção usadas devem estar todas definidas em LISP no fim da representação dos vários enquadramentos.

Exercício 10.1
Represente em KEE a seguinte informação:

- Existem praias de mar e praias de rio.
- As praias podem ou não ser concessionadas.
- As praias de mar têm ondas, mas as de rio não têm.
- A praia mais bonita é a PraiaGrande, que é uma praia de mar não concessionada.
- A P1 ou é uma praia de mar ou é uma praia de rio (mas não ambas simultaneamente).
- A quantidade de lixo existente nas praias pode ser calculada como um kilo por cada cem utentes.

Resposta:

Hierarquia:



Em que:
$a \longrightarrow b$ significa que todos os *a*s são *b*s
$A \dashrightarrow b$ significa que este A é um *b*

Seguem-se os enquadramentos:

Praia

Superclasses: —
Subclasses: PMar, PRio[2]
DECOMPOSITION.DISJOINT
Instâncias: P1[3]

Concessionada?	MEMBER
Valor: —	
Classe Val: BOOLEAN	
Card Max: 1	
Card Min: 1	
Tipo Herança: OVERRIDE	
TemOndas?	MEMBER
Valor: —	
Classe Val: BOOLEAN	
Card Max: 1	
Card Min: 1	
Tipo Herança: OVERRIDE	
MaisBonita	OWN[4]
Valor: PGrande	
Classe Val: (MEMBER.OF Praia)	
Card Max: 1	
Card Min: 1	
NumUtentes	MEMBER
Valor: —	
Classe Val: INTEGER	
Card Max: 1	
Card Min: 1	
Tipo Herança: OVERRIDE	
QuantidadeLixo	MEMBER
Valor: proc-calc-lixo	
Classe Val: METHOD	

PMar

Superclasses: Praia
Subclasses: —
Instâncias: PGrande

TemOndas?	MEMBER
Valor: T	
Tipo Herança: UNIQUE	

PRio

Superclasses: Praia
Subclasses: —
Instâncias: —

TemOndas?	MEMBER
Valor: NIL	
Tipo Herança: UNIQUE	

PGrande

Membro: PMar

Concessionada?	OWN
Valor: NIL	

P1

Membro: Praia

Definição do procedimento `proc-calc-lixo`, do tipo if-needed.

```
(defun proc-calc-lixo (unit)
   (/ (get-value unit NumUtentes) 100))
```

Para dizermos que a `P1` ou é uma praia de mar ou é uma praia de rio (mas não ambas simultaneamente), precisamos de usar TellAndAsk, pois não o conseguimos representar graficamente:

```
(ASSERT (OR (AND (IN.CLASS P1 PMar)
                 (NOT (IN.CLASS P1 PRio)))
            (AND (NOT (IN.CLASS P1 PMar))
                 (IN.CLASS P1 PRio))))
```

[2]Consideramos que pode haver outros tipos de praias, por exemplo as praias em lagos ou em barragens, por isso a decomposição não é completa.

[3]Apenas escrevemos as superclasses, subclasses e instâncias directas, as outras são determinadas pelo KEE.

[4]Os atributos próprios não têm tipo de herança porque não são propagados.

Exercício 10.2
Considere a seguinte informação acerca de mamíferos. Com base nela e no conhecimento (de senso comum) que tem acerca do mundo, represente-a em KEE.

Existem aproximadamente 4500 espécies de mamíferos no mundo, e todos eles respiram oxigénio do ar. Os mamíferos também são caracterizados por cuidarem das suas crias enquanto bebés e por as alimentarem de leite materno.

As fêmeas dos mamíferos desenvolvem as crias dentro do ventre e quando chegam ao fim da gestação podem ter entre 1 e 27 crias.

Uma das excepções a esta regra são os monotremas (que incluem os ornitorrincos e as equidnas), que põem ovos e os incubam, para as crias se desenvolverem. Os machos dos ornitorrincos são também caracterizados, entre outras coisas, por terem espigões venenosos nas patas traseiras.

Os humanos, em contrapartida, podem ser caracterizados pelo índice de massa corporal: peso em kilos a dividir pela altura em metros ao quadrado. Este valor é interpretado da seguinte forma:

IMC inferior 18,5	Peso abaixo do normal
IMC de 18,5 a 25	Peso Normal
IMC de 25 a 29,9	Excesso de Peso
IMC superior 30	Obesidade

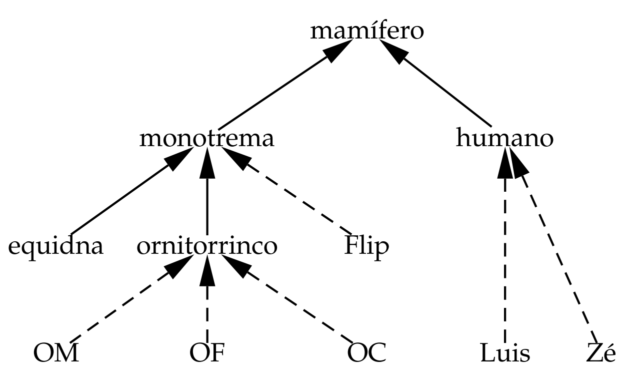
O Luís é um humano que pesa 90Kg e mede 1,90m. O Zé é uma cria do Luis.

A OF é um ornitorrinco fêmea, o OM é um ornitorrinco macho e a OC é uma cria de OF.

Há um mamífero (chamemos-lhe Flip) que ou é um ornitorrinco ou é uma equidna.

Resposta:

Em primeiro lugar, devemos decidir que hierarquia é que vamos usar.



Em que:

$a \longrightarrow b$	significa que todos os *a*s são *b*s
$A \dashrightarrow b$	significa que este A é um *b*

Nesta hierarquia, representámos o Flip como sendo um monotrema pois não sabemos se ele é ornitorrinco ou equidna, mas temos a certeza que é monotrema.

Temos que decidir como vamos representar machos/fêmeas, bebés, crias, etc. Temos várias alternativas:

1. Dividir cada classe em machos e fêmeas e ter herança múltipla — esta alternativa implica triplicar o número de classes, pois podemos ter mamíferos de que sabemos a espécie mas não o sexo.
2. Ter herança múltipla apenas para as instâncias — esta alternativa não implica ter muitas mais classes, mas não vamos conseguir dizer que os ornitorrincos machos têm espigões venenosos, pois o conceito de ornitorrinco macho não existe.
3. Representar o sexo como um atributo dos mamíferos e usar regras e procedimentos para inferir nova informação — esta alternativa é a mais correcta neste caso.

Nota: se estivéssemos a falar apenas dos humanos, que podem ser homens ou mulheres, fazia sentido ter as três classes, aqui não faz porque temos muitas classes já à partida. Mais uma vez, a melhor representação depende do resto do conhecimento e da utilização que quisermos fazer dele...

Depois, podemos começar a escrever os vários enquadramentos, em que:

- "—" numa faceta de um atributo significa que o valor dessa faceta é desconhecido.
- "NIL" numa faceta de um atributo significa que essa faceta não tem valor, ou seja, o seu valor é NIL.

Mamifero

Superclasses: —
Subclasses: Monotrema, Humano
DECOMPOSITION.DISJOINT
Instâncias: —

NumEspecies	OWN
Valor: Aprox4500 Classe Val: — Card Max: 1 Card Min: 1	
Respira	MEMBER
Valor: OxigenioAr Classe Val: (ONE.OF OxigenioAr OxigenioAgua) Card Max: 1 Card Min: 1 Tipo Herança: UNIQUE[5]	
Crias	MEMBER
Valor: — Classe Val: (MEMBER.OF Mamifero)[6] Card Max: — Card Min: 0 Tipo Herança: OVERRIDE.VALUES	
CuidaCriasBebes	MEMBER
Valor: T Classe Val: BOOLEAN[7] Card Max: 1 Card Min: 1 Tipo Herança: UNIQUE	
AlimentoCriasBebes	MEMBER
Valor: LeiteMaterno Classe Val: —[8] Card Max: — Card Min: 1 Tipo Herança: UNION[9]	
TipoReproducao	MEMBER
Valor: Viviparo Classe Val: (ONE.OF Viviparo Oviparo Ovoviviparo) Card Max: 1 Card Min: 1 Tipo Herança: OVERRIDE	
Sexo	MEMBER
Valor: — Classe Val: (ONE.OF Masculino Feminino) Card Max: 1 Card Min: 1 Tipo Herança: OVERRIDE	
Peso	MEMBER
Valor: — Classe Val: REAL Card Max: 1 Card Min: 1 Tipo Herança: OVERRIDE	
Altura	MEMBER
Valor: — Classe Val: REAL Card Max: 1 Card Min: 1 Tipo Herança: OVERRIDE	

Para representarmos o número de crias das gestações da fêmeas, precisamos de usar regras, pois é um atributo que depende do valor de outro atributo (o sexo):

```
(IF (AND (IN.CLASS ?x Mamifero)
         (OWN.VALUE Sexo ?x Feminino))
 THEN (ASSERT (OWN.VALUE NumMinCriasGestacao ?x 1))
      (ASSERT (OWN.VALUE.IN.CLASS NumMinCriasGestacao ?x INTEGER))
      (ASSERT (OWN.MAX.CARD NumMinCriasGestacao ?x 1))
      (ASSERT (OWN.MIN.CARD NumMinCriasGestacao ?x 1))
      (ASSERT (OWN.VALUE NumMaxCriasGestacao ?x 27))
      (ASSERT (OWN.VALUE.IN.CLASS NumMaxCriasGestacao ?x INTEGER))
      (ASSERT (OWN.MAX.CARD NumMaxCriasGestacao ?x 1))
      (ASSERT (OWN.MIN.CARD NumMaxCriasGestacao ?x 1)))
```

Convém notar que estes novos atributos também existem para as monotremas. Isto não é muito grave se considerarmos cada "ninhada" como uma espécie de gestação. Caso queiramos, também podemos acrescentar no antecedente da regra `(NOT (IN.CLASS ?x Monotrema))`

Monotrema	
Superclasses: Mamifero Subclasses: Ornitorrinco, Equidna[10] DECOMPOSITION.DISJOINT DECOMPOSITION.COMPLETE Instâncias: Flip	
Crias Classe Val: (MEMBER.OF Monotrema)	MEMBER
TipoReproducao Valor: Oviparo Tipo Herança: UNIQUE	MEMBER

Os monotremas herdam todos os atributos dos mamíferos, e não é preciso escrevê-los porque são inferidos pelo KEE. Só devemos escrever o que muda.

Seria complicado representar que "as fêmeas dos monotremas põem ovos e incubam os seus ovos para as crias se desenvolverem", mas podemos dizer que as fêmeas põem ovos:

```
(IF (AND (IN.CLASS ?x Monotrema)
         (OWN.VALUE Sexo ?x Feminino))
 THEN (ASSERT (OWN.VALUE PoeOvos ?x T))
      (ASSERT (OWN.VALUE.IN.CLASS PoeOvos ?x BOOLEAN))
      (ASSERT (OWN.MAX.CARD PoeOvos ?x 1))
      (ASSERT (OWN.MIN.CARD PoeOvos ?x 1)))
```

[5] Usamos este tipo de herança porque não queremos que o valor deste atributo possa ser alterado.

[6] O valor deste atributo só dever ser preenchido nas instâncias, pois não há classes de mamíferos com crias, as suas instâncias é que podem ter crias.

[7] Esta não é a representação ideal, mas temos dificuldade em dizer que cada mamífero cuida das **suas** crias.

[8] O que é alimento para uma classe pode não ser alimento para outra: os elefantes comem erva, mas os bebés humanos não.

[9] Para podermos acrescentar outros alimentos, mas termos sempre o leite materno.

[10] A decomposição é completa porque só se conhecem estas duas classes de monotremas.

Ornitorrinco

Superclasses: Monotrema
Subclasses: —
Instâncias: OM, OF, OC

Crias	MEMBER
Classe Val: (MEMBER.OF Ornitorrinco)	

Equidna

Superclasses: Monotrema
Subclasses: —
Instâncias: —

Crias	MEMBER
Classe Val: (MEMBER.OF Equidna)	

Para representar os espigões dos ornitorrincos machos precisamos de uma regra:

```
(IF (AND (IN.CLASS ?x Monotrema)
         (OWN.VALUE Sexo ?x Masculino))
 THEN (ASSERT (OWN.VALUE TemEspigaoVenenoso ?x T))
      (ASSERT (OWN.VALUE.IN.CLASS TemEspigaoVenenoso ?x BOOLEAN))
      (ASSERT (OWN.MAX.CARD TemEspigaoVenenoso ?x 1))
      (ASSERT (OWN.MIN.CARD TemEspigaoVenenoso ?x 1)))
```

De seguida, temos as instâncias dos monotremas:

OM

Membro: Ornitorrinco

Sexo	OWN
Valor: Masculino	

OF

Membro: Ornitorrinco

Crias	OWN
Valor: OC	
Sexo	OWN
Valor: Feminino	

OC

Membro: Ornitorrinco

Para o Flip, podemos dizer que é um monotrema:

Flip

Membro: Monotrema

Neste caso, como a decomposição dos monotremas é disjunta e completa, ele vai ter que ser ornitorrinco ou equidna e não poderá ser as duas coisas simultaneamente. Caso esta decomposição não se verificasse, era necessário acrescentar a regra:

```
(ASSERT (OR (AND (IN.CLASS Flip Ornitorrinco)
                 (NOT (IN.CLASS Flip Equidna)))
            (AND (NOT (IN.CLASS Flip Ornitorrinco))
                 (IN.CLASS Flip Equidna))))
```

Para os humanos, teremos o seguinte:

Humano	
Superclasses: Mamifero Subclasses: — Instâncias: Luis, Ze	
Crias Classe Val: (MEMBER.OF Humano)	MEMBER
IMC Valor: proc-imc (**ifneeded**) Classe Val: METHOD	MEMBER
TipoPeso Valor: proc-tipo-peso (**ifneeded**) Classe Val: METHOD	MEMBER

```
(defun proc-imc (unit)
   (/ (get-value unit Peso)
      (expt (get-value unit Altura) 2)))

(defun proc-tipo-peso (unit)
   (let ((imc (get-value unit IMC)))
      (cond ((not (numberp imc)) TipoPesoDesconhecido)
            ((< imc 18.5) PesoInferiorNormal)
            ((< imc 25) PesoNormal)
            ((< imc 29.9) PesoExcessivo)
            (t Obesidade))))
```

Nota: embora o peso e a altura só sejam necessários para os humanos, colocámos estes atributos nos mamíferos, pois é a classe onde fazem sentido.

De seguida, temos as instâncias dos humanos:

Luis	
Membro: Humano	
Crias Valor: Ze	OWN
Sexo Valor: Masculino	OWN
Peso Valor: 90	OWN
Altura Valor: 1.9	OWN

Ze	
Membro: Humano	
Sexo Valor: Masculino	OWN

Seria interessante também representar as relações familiares, por exemplo que o Luís é pai do Zé, mas isso está fora do âmbito deste exercício.

Exercício 10.3
Considere a representação em KEE da informação do Exercício 10.2.
Explique a resposta que seria dada pelo KEE a cada uma das seguintes perguntas:

1. Quantas espécies de mamíferos existem?
2. E quantas espécies de monotremas existem?
3. Qual é o tipo de reprodução dos mamíferos?
4. Quem é que é mamífero?
5. Quais são as subclasses de mamífero?
6. O que é que o Zé respira?
7. Quem é que respira oxigénio do ar?
8. Qual é o índice de massa corporal do Luís?
9. Qual é o tipo de peso do Luís?
10. O Luís cuida do Zé?
11. E do Flip?
12. O Luís é pai do Zé?
13. Qual é a forma de reprodução do Flip?
14. E qual é o seu peso?
15. Quem é que põe ovos?

Resposta:

Nota: A resposta do KEE à pergunta do valor de um atributo quando o enquadramento não tem esse atributo é `NIL`.

1. Quantas espécies de mamíferos existem?
   `(QUERY (OWN.VALUE NumEspecies Mamifero ?x))`
   Resposta: A constante `Aprox4500`.

2. E quantas espécies de monotremas existem?
   `(QUERY (OWN.VALUE NumEspecies Monotrema ?x))`
   Resposta: A classe Monotrema não tem o atributo NumEspecies, porque este é um atributo próprio da classe Mamifero e por isso não é propagado. O KEE responde que não tem, ou seja, `NIL`.
   Eventualmente, faria sentido ter um atributo próprio em cada classe que indicasse o número de espécies que ela representa. Nesta caso, para os monotremas teria o valor 2.

3. Qual é o tipo de reprodução dos mamíferos?
   `(QUERY (MEMBER.VALUE TipoReproducao Mamifero ?x))`
   Resposta: O valor por omissão: `Viviparo`.

4. Quem é que é mamífero?
   `(QUERY (IN.CLASS ?x Mamifero))`
   Resposta: Todas as instâncias: `OM, OF, OC, Flip, Luis, Ze`.

5. Quais são as subclasses de mamífero?
   `(QUERY (SUBCLASS.OF ?x Mamifero))`
   Resposta: Todas as subclasses: `Monotrema, Ornitorrinco, Equidna, Humano`.

6. O que é que o Zé respira?
   `(QUERY (OWN.VALUE Respira Ze ?x))`
   Resposta: O valor herdado da classe Mamifero: `OxigenioAr`.

7. Quem é que respira oxigénio do ar?
   `(QUERY (OWN.VALUE Respira ?x OxigenioAr))`
   ou `(QUERY (MEMBER.VALUE Respira ?x OxigenioAr))`
   Resposta: Dependendo da formulação da pergunta: `OM, OF, OC, Flip, Luis, Ze` ou `Mamifero, Monotrema, Ornitorrinco, Equidna, Humano`.

8. Qual é o índice de massa corporal do Luís?
   `(QUERY (OWN.VALUE IMC Luis ?x))`
   Resposta: Depois de invocar o procedimento `proc-imc`: `24.93`.

9. Qual é o tipo de peso do Luís?
   `(QUERY (OWN.VALUE TipoPeso Luis ?x))`
   Resposta: Depois de invocar o procedimento `proc-tipo-peso`: `PesoNormal`.

10. O Luís cuida do Zé?
    Esta pergunta não se pode fazer directamente, mas podemos compôr uma pergunta:

    ```
    (QUERY (AND (OWN.VALUE CuidaCriasBebes Luis T)
                (OWN.VALUE Crias Luis ?x)
                (member Ze ?x)))
    ```

    Resposta: `T`, uma vez que o Luis é mamífero e o Zé é uma das suas crias.

11. E do Flip?

    ```
    (QUERY (AND (OWN.VALUE CuidaCriasBebes Luis T)
                (OWN.VALUE Crias Luis ?x)
                (member Flip ?x)))
    ```

    Resposta: `NIL`, uma vez que o Flip não é uma das crias do Luis.

12. O Luís é pai do Zé?
    Não temos o atributo Pai, mas sabemos que o Luís é pai do Zé se o Zé for uma das suas crias e o sexo do Luís for masculino.

    ```
    (QUERY (AND (OWN.VALUE Sexo Luis Masculino)
                (OWN.VALUE Crias Luis ?x)
                (member Ze ?x)))
    ```

Resposta: `T`

13. Qual é a forma de reprodução do Flip?
    `(QUERY (OWN.VALUE TipoReproducao Flip ?x))`
    Resposta: Valor herdado de Monotrema, que é a classe mais próxima e porque o tipo de herança é OVERRIDE: `Oviparo`.
14. E qual é o seu peso?
    `(QUERY (OWN.VALUE Peso Flip ?x))`
    Resposta: Este valor ainda não foi introduzido, por isso é desconhecido: `UNKNOWN` ou `--`.
15. Quem é que põe ovos?
    `(QUERY (OWN.VALUE PoeOvos ?x T))`
    Resposta: Vai usar a regra de produção que foi escrita e responder com as monotremas fêmeas que conhecer: `OF`.

Exercício 10.4
Discuta possíveis representações em KEE para os seguintes tipos de conhecimento:

1. Relações familiares: pai, mãe, irmãos, avós, tios, primos, etc.
2. Números e operações aritméticas: pares, ímpares, primos, somas, multiplicações, elementos neutros, etc.
3. Relações e suas propriedades: transitividade, reflexividade, simetria, equivalência, etc.
4. Conhecimento ou crenças de pessoas: o que é que alguém sabe, pensa que sabe, acredita, etc.

Resposta:

1. Relações familiares
   Vamos querer falar acerca de pessoas. Por isso, `Pessoa` vai ser uma classe.
2. Números e operações aritméticas
3. Relações e suas propriedades
4. Conhecimento ou crenças de pessoas

11 Lógicas Descritivas — KL-ONE e Sintaxe Abstracta

Sumário:

- Representação usando KL-ONE.
- Representação usando a sintaxe abstracta das lógicas descritivas.

Resumo:

Nas lógicas descritivas representam-se as características definidoras dos conceitos. Por isso, não há valores por omissão nem cancelamento do que é herdado, porque se representam apenas as condições necessárias para pertencer a esse conceito.

Nas lógicas decritivas distingue-se entre:

- **Conhecimento intensional** — usado para representar conhecimento acerca dum domínio, através da definição dos conceitos nele existentes. Isto é feito na TBox (ou Terminological Component).
- **Conhecimento extensional** — usado para especificar as propriedades dos objectos existentes no domínio. Isto é feito na ABox (ou Assertional Component).

TBox

Tipos de conceitos

- **Conceitos genéricos** representam classes e podem ser:
  - **Primitivos** — definidos apenas através de condições necessárias, isto é, a definição que temos é incompleta. São representados em KL-ONE com *. Ex: "Coisa"; "Cão" é um "Mamífero" que ladra e ...
  - **Definidos** — definidos através de condições necessárias e suficientes. Ex: "Mulher" é um "Humano" do sexo feminino.
- **Conceitos individuais** representam instâncias, isto é, classes com um único elemento. O facto de serem descritos na TBox não obriga a que existam no mundo (ou seja, na ABox).

Definição de conceitos

- **Especificações taxonómicas** — relacionam um conceito com outros mais gerais ou mais específicos do que ele (superconceito/subconceito).
- **Especificações de papéis** — descrevem as relações entre as instâncias do conceito e as de outros conceitos com ele associados. Correspondem a uma generalização da noção de atributo e estão organizados numa hierarquia. Podem ser especificados através de restrições ou diferenciações.
- **Condições estruturais** — definem relações entre papéis e restringem os valores que um papel pode ter. Podem ser:

  – **Gerais** ou relações estruturais, quando especificam o modo como os papéis e os conceitos se interrelacionam.
  – **Particulares** ou “role value maps”, quando correspondem a ligações entre valores de papéis.

Conceito bem formado satisfaz pelo menos uma das seguintes restrições:

- Tem mais do que um superconceito, sendo a conjunção deles
- Tem pelo menos uma restrição em relação ao seu superconceito
- É primitivo

ABox

Na ABox especificam-se as propriedades dos objectos existentes no domínio.

Passos para representar conhecimento em lógicas descritivas

1. Enumerar os termos, sem distinguir entre indivíduos, conceitos e papéis.
2. Distinguir entre conceitos, papéis e indivíduos — os conceitos têm existência independente, enquanto os papéis dependem de outros objectos para existirem; os indivíduos são os que existem no domínio.
3. Desenvolver a taxonomia de conceitos — criar uma classificação para todos os conceitos, e imaginando as suas possíveis extensões.
4. Determinar as propriedades e partes dos conceitos — para cada conceito, determinar as suas propriedades, que são os papéis: propriedades instrínsecas e extrínsecas, partes e relações com outros conceitos.
5. Determinar as restrições de número e de valor — determinar a cardinalidade e a classe de cada papel de cada conceito.
6. Determinar as restrições de valor que não foram representadas — no caso de ser necessário, representar novos conceitos na taxonomia, para respeitar as restrições de valor do passo anterior.
7. Distinguir propriedades essenciais de acessórias
8. Distinguir conceitos definidos e primitivos

Sintaxe Abstracta

Só há uma definição para cada conceito e papel.

As definições não têm ciclos.

Unique Name Assertion — nomes diferentes denotam indivíduos diferentes.

Sintaxe Abstracta das Lógicas Descritivas

Descrição de Conceitos

$\top$	conceito universal
$\bot$	conceito inconsistente
C	conceito C
$\neg\mathsf{C}$	complemento de C
$\mathsf{C}_1 \sqcap \mathsf{C}_2$	intercecção de conceitos
$\mathsf{C}_1 \sqcup \mathsf{C}_2$	reunião de conceitos
$\forall\mathsf{R.C}$	restrição de valor
$\exists\mathsf{R}.\top$	quantificação existencial limitada
$\exists\mathsf{R.C}$	quantificação existencial completa
$\geq n\mathsf{R}$	restrição de cardinalidade
$\leq n\mathsf{R}$	restrição de cardinalidade
$\{\mathsf{i}_1, \ldots, \mathsf{i}_\mathsf{n}\}$	conjunto de conceitos individuais

Descrição de Papéis

$\cup$	papel universal
R	papel R
$\neg\mathsf{R}$	complemento de R
$\mathsf{R}_1 \sqcap \mathsf{R}_2$	intercecção de papéis
$\mathsf{R}_1 \sqcup \mathsf{R}_2$	reunião de papéis
$\mathsf{R}_1 \odot \mathsf{R}_2$	composição de papéis
R^+	fecho transitivo
R^-	relação inversa

TBox

$\mathsf{C}_1 \equiv \mathsf{C}_2$	igualdade de conceitos
$\mathsf{R}_1 \equiv \mathsf{R}_2$	igualdade de papéis
$\mathsf{C}_1 \sqsubseteq \mathsf{C}_2$	inclusão de conceitos
$\mathsf{R}_1 \sqsubseteq \mathsf{R}_2$	inclusão de papéis

ABox

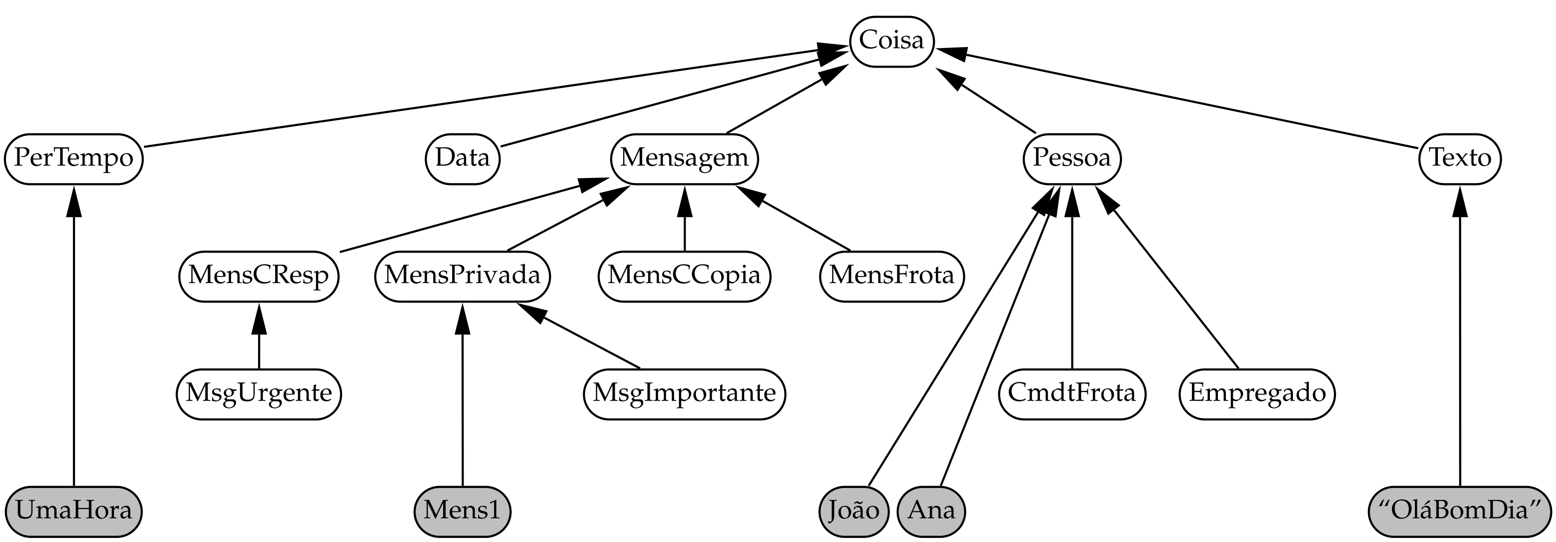
$\mathsf{C(a)}$	assercão de conceito
$\mathsf{R(b,c)}$	asserção de papel: o filler c preenche o papel R do conceito b

Exercício 11.1
(Adaptado do artigo "An overview of the KL-ONE knowledge representation system" de R. J. Brachman e J. G. Schmolze)
Represente em KL-ONE a seguinte informação:

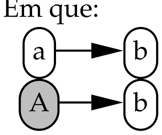
1. Uma mensagem é, entre outras coisas, uma coisa, com pelo menos um emissor (que é uma pessoa), pelo menos um receptor (que é uma pessoa), um corpo (que é um texto), uma data de emissão e uma data de recepção (que são datas).
2. Uma mensagem de frota é uma mensagem cujo(s) emissor(es) é (são) comandante(s) de frota.
3. Uma mensagem privada é uma mensagem com um único receptor.
4. O João enviou à Ana uma mensagem privada com o texto "Olá, bom dia.".
5. Uma mensagem com cópia é uma mensagem que tem, entre os possíveis receptores, pelo menos um que é aquele a quem a mensagem se destina (que é o ParaReceptor) e tem pelo menos um receptor para o qual é enviada uma cópia da mensagem (o CópiaReceptor).
6. Uma mensagem importante é uma mensagem privada cujo receptor é um empregado e cujo emissor é o chefe do receptor.
7. Uma mensagem com resposta é uma mensagem com uma data de resposta, que é uma data.
8. Uma mensagem urgente é uma mensagem com resposta que é respondida menos de uma hora depois de ser recebida. (Ou seja, uma mensagem urgente é uma mensagem com resposta cuja data de recepção e data de resposta satisfazem uma relação MenorQue, cujo menor é a data de recepção, cujo maior é a data de resposta e cuja diferença é menor que uma hora, que é um período de tempo.)

Resposta:

Podemos começar por pensar na hierarquia de classes e instâncias que vamos ter.



Em que:

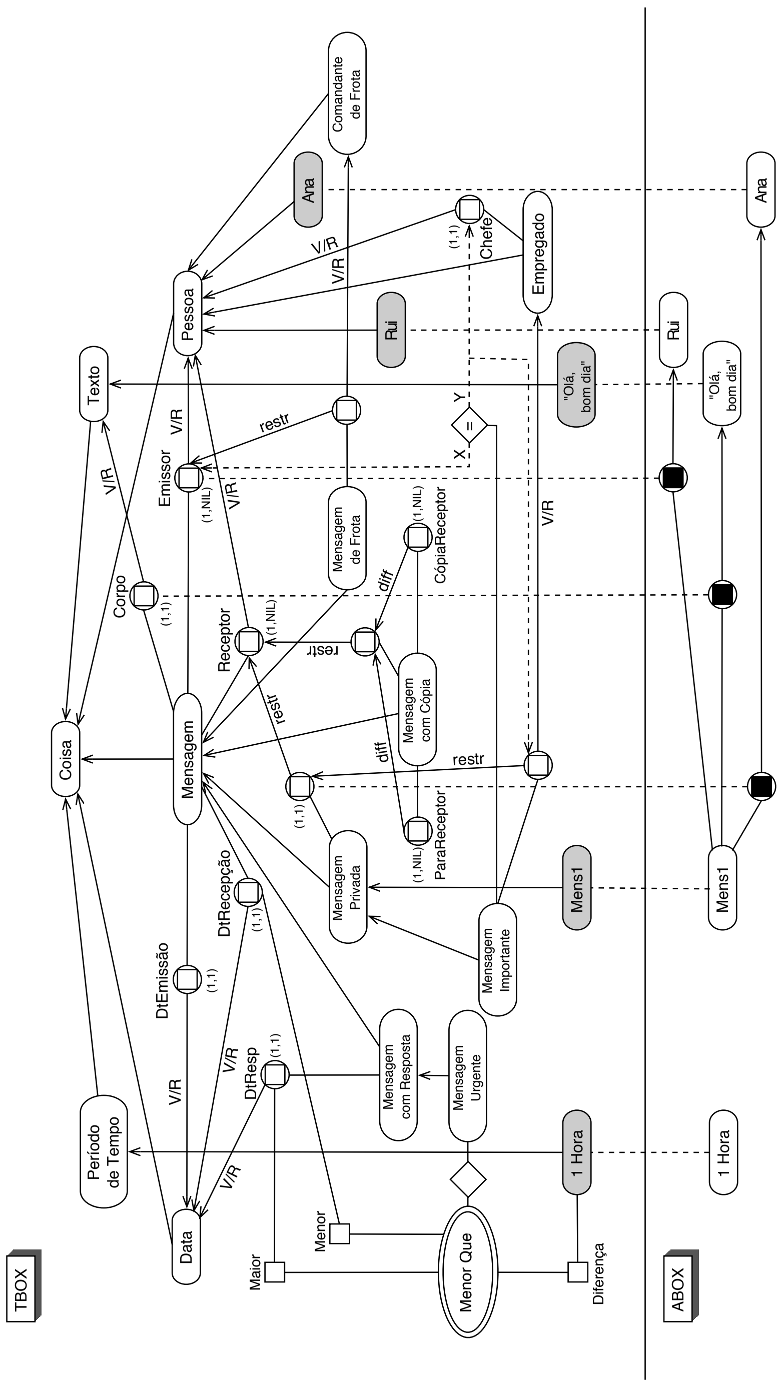


significa que todos os *a*s são *b*s

significa que este A é um *b*

Ver figura na página seguinte. As figuras da resposta a este exercício, bem como os acetatos para usar nas aulas, foram passadas para computador pela Carla Penedo.

Relativamente a estas figuras, para representar as mensagens urgentes é ainda necessário dizer que a relação *menor que* significa que quem satisfizer o seu papel *menor* é menor que quem satisfizer o papel *maior* e que a diferença entre eles é menor que quem satisfizer o papel *diferença*.



Exercício 11.2
Represente, usando a sintaxe abstracta das lógicas descritivas, a seguinte informação:

- As praias podem ser caracterizadas pela sua localização (que é um lugar), pelo seu comprimento e pela sua largura (que são números).
- As praias têm também vários acessos, que são passagens.
- Dos vários acessos existentes numa praia, existe um que é o principal.
- As praias de mar estão localizadas à beira-mar e as praias de rio estão localizadas à beira-rio.
- A P1 é uma praia de mar de comprimento 1000 cujo acesso principal é o A1.

Resposta:

Resposta:

Na TBox temos a descrição dos conceitos:

$$
\begin{aligned}
\mathsf{Praia} \sqsubseteq^{11} \ & \top \quad \sqcap \\
& \forall \mathsf{localizacao.Lugar} \quad \sqcap \\
& \geq 1 \quad \mathsf{localizacao} \quad \sqcap \\
& \leq 1 \quad \mathsf{localizacao} \quad \sqcap \\
& \forall \mathsf{comprimento.Numero} \quad \sqcap \\
& \geq 1 \quad \mathsf{comprimento} \quad \sqcap \\
& \leq 1 \quad \mathsf{comprimento} \quad \sqcap \\
& \forall \mathsf{largura.Numero} \quad \sqcap \\
& \geq 1 \quad \mathsf{largura} \quad \sqcap \\
& \leq 1 \quad \mathsf{largura} \quad \sqcap \\
& \forall \mathsf{acesso.Passagem} \quad \sqcap \\
& \geq 1 \quad \mathsf{acesso} \quad \sqcap \\
& \exists \mathsf{principal.Passagem} \quad \sqcap \\
& \leq 1 \quad \mathsf{principal}
\end{aligned}
$$

$$\mathsf{Lugar} \sqsubseteq \top$$

$$\mathsf{Numero} \sqsubseteq \top$$

$$\mathsf{Passagem} \sqsubseteq \top$$

$$\mathsf{principal} \sqsubseteq \mathsf{acesso}$$

$$
\begin{aligned}
\mathsf{PraiaMar} \equiv {}^{12}\ & \mathsf{Praia} \quad \sqcap \\
& \forall \mathsf{localizacao.BeiraMar}
\end{aligned}
$$

$$\mathsf{BeiraMar} \sqsubseteq \mathsf{Lugar}$$

$$
\begin{aligned}
\mathsf{PraiaRio} \equiv \ & \mathsf{Praia} \quad \sqcap \\
& \forall \mathsf{localizacao.BeiraRio}
\end{aligned}
$$

$$\mathsf{BeiraRio} \sqsubseteq \mathsf{Lugar}$$

Na ABox temos informação acerca das instâncias:

$$\mathsf{PraiaMar(P1)}$$

$$\mathsf{comprimento(P1, 1000)}$$

$$\mathsf{principal(P1, A1)}$$

[11] Porque pode haver objectos que satisfaçam a descrição da direita e não sejam praias (Praia é um conceito primitivo).

[12] Porque todas as praias de mar são praias localizadas à beira mar e todos os objectos que satisfaçam a descrição da direita são obrigatoriamente praias de mar (PraiaMar é um conceito definido).

Com base na informação representada na TBox e na ABox podemos inferir:

$$\mathsf{Praia}(\mathsf{P1})$$
$$\mathsf{Numero}(\mathsf{1000})$$
$$\mathsf{acesso}(\mathsf{P1}, \mathsf{A1})$$
$$\mathsf{Passagem}(\mathsf{A1})$$
$$\mathsf{localizacao}(\mathsf{P1}, \mathsf{BeiraMar})$$

Exercício 11.3

Usando a sintaxe abstracta das lógicas terminológicas, represente a seguinte informação:

- Existem vários tipos de portas lógicas: portas AND, OR e NOT.
- As portas lógicas têm pelo menos uma entrada e exactamente uma saída, que são valores lógicos.
- Os valores lógicos são TRUE e FALSE.
- As portas NOT têm apenas uma entrada e uma saída.
- N1 é uma porta NOT com entrada TRUE e saída FALSE.

Resposta:

Na TBox temos a descrição dos conceitos:

$$\begin{aligned} \mathsf{PortaLogica} \sqsubseteq\ & \top \quad \sqcap \\ & \forall \mathsf{entrada}.\mathsf{ValorLogico} \quad \sqcap \\ & \geq 1 \quad \mathsf{entrada} \quad \sqcap \\ & \forall \mathsf{saida}.\mathsf{ValorLogico} \quad \sqcap \\ & \geq 1 \quad \mathsf{saida} \quad \sqcap \\ & \leq 1 \quad \mathsf{saida} \end{aligned}$$

$$\mathsf{ValorLogico} \equiv \{\mathsf{TRUE}, \mathsf{FALSE}\}$$
$$\mathsf{PortaAnd} \sqsubseteq \mathsf{PortaLogica}$$
$$\mathsf{PortaOr} \sqsubseteq \mathsf{PortaLogica}$$
$$\begin{aligned} \mathsf{PortaNot} \sqsubseteq\ & \mathsf{PortaLogica} \quad \sqcap \\ & \leq 1 \quad \mathsf{entrada} \end{aligned}$$

Na ABox temos informação acerca das instâncias:

$$\mathsf{PortaNot}(\mathsf{N1})$$
$$\mathsf{entrada}(\mathsf{N1}, \mathsf{TRUE})$$
$$\mathsf{saida}(\mathsf{N1}, \mathsf{FALSE})$$

Com base na informação representada na TBox e na ABox podemos inferir:

$$\mathsf{PortaLogica}(\mathsf{N1})$$

Exercício 11.4

Considere a seguinte informação acerca de mamíferos. Com base nela e no conhecimento (de senso comum) que tem acerca do mundo, represente-a usando a sintaxe abstracta das lógicas descritivas.

Existem aproximadamente 4500 espécies de mamíferos no mundo, e todos eles respiram oxigénio do ar. Os mamíferos também são caracterizados por cuidarem das suas crias enquanto bebés e por as alimentarem de leite materno.

As fêmeas dos mamíferos desenvolvem as crias dentro do ventre e quando chegam ao fim da gestação podem ter entre 1 e 27 crias.

Uma das excepções a esta regra são os monotremas (que incluem os ornitorrincos e as equidnas), que põem ovos e os incubam, para as crias se desenvolverem. Os machos dos ornitorrincos são também caracterizados, entre outras coisas, por terem espigões venenosos nas patas traseiras.

Os humanos, em contrapartida, podem ser caracterizados pelo índice de massa corporal: peso em kilos a dividir pela altura em metros ao quadrado. Este valor é interpretado da seguinte forma:

IMC inferior 18,5	Peso abaixo do normal
IMC de 18,5 a 25	Peso Normal
IMC de 25 a 29,9	Excesso de Peso
IMC superior 30	Obesidade

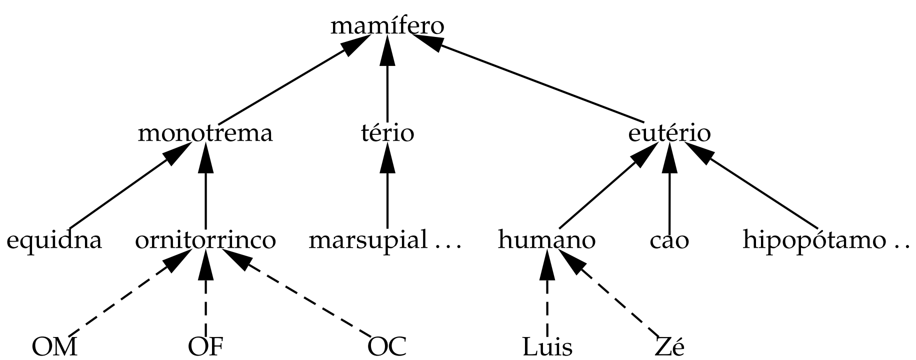
O Luís é um humano que pesa 90Kg e mede 1,90m. O Zé é uma cria do Luis.

A OF é um ornitorrinco fêmea, o OM é um ornitorrinco macho e a OC é uma cria de OF.

Há um mamífero (chamemos-lhe Flip) que ou é um ornitorrinco ou é uma equidna.

Resposta:

Em primeiro lugar, devemos decidir que hierarquia é que vamos usar. Como nestes formalismos não podemos ter excepções, pois apenas se representam as características definidoras dos conceitos, vamos ter que ver como é que podemos ter uma hierarquia em que as propriedades das classes não tenham excepções. No caso dos mamíferos, a hierarquia é a seguinte:



Em que:

a ⟶ b significa que todos os *a*s são *b*s
A - - ➤ b significa que este A é um *b*