Exercícios de
Lógica para Programação

Maria dos Remédios Cravo

Departamento de Engenharia Informática
Instituto Superior Técnico
Universidade de Lisboa

Conteúdo

Capítulo 1

Conceitos básicos

1.1 Validade de argumentos

1.1.1. Para cada argumento, prove se é válido ou inválido, usando exclusivamente a definição de argumento válido.

(a) O Pedro é jogador de xadrez
O Pedro é inteligente
∴ Todos os jogadores de xadrez são inteligentes

(b) Os cães são animais
Os gatos são animais
O Bobi é um gato ou um cão
∴ O Bobi é um animal

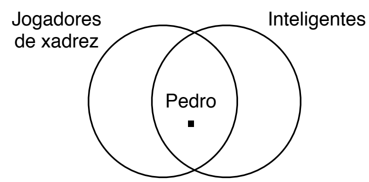
(c) Os cães são plantas
O Bobi não é uma planta
∴ O Bobi não é um cão

(d) O Bobi é um cão
O Bobi não é um cão
∴ O Tareco é um gato

(e) Sempre que faz sol, a Maria joga ténis
∴ Sempre que não faz sol, a Maria não joga ténis

Resposta:

(a) Consideremos o seguinte diagrama:



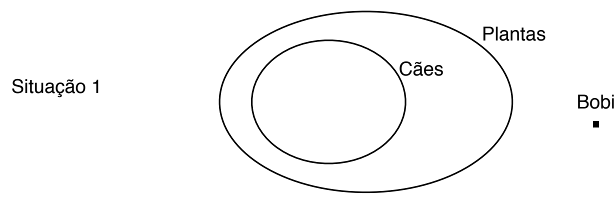
Este diagrama mostra que é possível ter todas as premissas verdadeiras e a conclusão falsa, logo o argumento não é válido.

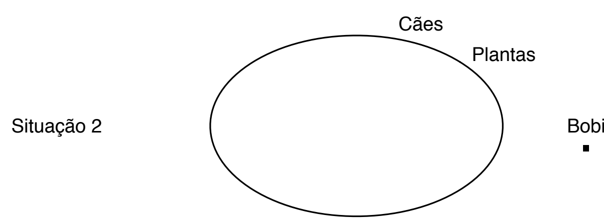
Em alternativa, podemos considerar o seguinte contra-argumento:

2 é um real
2 é um inteiro
$\therefore$ Todos os reais são inteiros

(b) Se o Bobi for um cão então o Bobi é um animal, porque os cães são animais. Por outro lado, se o Bobi não for um cão, então é um gato e nesse caso também é um animal, porque os gatos são animais. Como é impossível ter as premissas verdadeiras e a conclusão falsa, o argumento é válido.

(c) Para as premissas serem verdadeiras, uma das duas situações abaixo tem de se verificar:

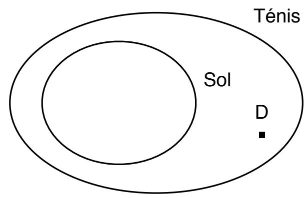




Em qualquer das situações a conclusão tem de ser verdadeira.

(d) É impossível ter ambas as premissas verdadeiras, logo o argumento é válido.

(e) O diagrama abaixo mostra que, sendo a premissa verdadeira, é possível existir um dia (representado por `D` no diagrama) em que não faz sol e a Maria joga ténis, o que torna a conclusão falsa. Logo, o argumento é inválido.



1.1.2. Usando exclusivamente o princípio da forma e os resultados do exercício anterior prove a invalidade ou invalidade dos seguintes argumentos.

(a) O Pedro é inteligente
O Pedro não é inteligente
∴ O António é trabalhador-estudante

(b) Os reais são inteiros
Os racionais são inteiros
x é um real ou um racional
∴ x é um inteiro

(c) Sempre que chove, o Bobi fica na casota
∴ Sempre que não chove, o Bobi não fica na casota

(d) Os cães são animais
O Bobi não é um animal
∴ O Bobi não é um cão

(e) O Carlos é jogador de basquetebol
O Carlos é alto
∴ Todos os jogadores de basquetebol são altos

Resposta:

(a) O argumento é válido porque tem a mesma forma que o argumento (d):
O A é B
O A não é B
∴ O C é D

(b) O argumento é válido porque tem a mesma forma que o argumento (b):
Os A's são B's
Os C's são B's
D é um A ou um C
∴ D é um B

(c) O argumento é inválido porque tem a mesma forma que o argumento (e):
Sempre que A, B
∴ Sempre que não A, não B

(d) O argumento é válido porque tem a mesma forma que o argumento (c):
Os A's são B's
O C não é um B
∴ O C não é um A

(e) O argumento é inválido porque tem a mesma forma que o argumento (a):
O A é B
O A é C
∴ Todos os B's são C's

1.1.3. Considere o seguinte argumento:

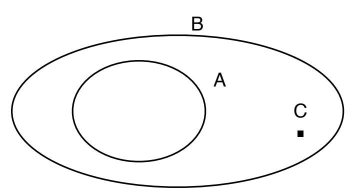
Todos os A's são B's
O C é um B
∴ O C é um A

(a) Prove que o argumento não é válido usando apenas a definição de argumento válido.

(b) Prove que o argumento não é válido usando o princípio da forma.

Resposta:

(a) A situação representada abaixo mostra que é possível ter todas as premissas verdadeiras e a conclusão falsa logo, por definição, o argumento não é válido.



(b) Vamos encontrar um contra-argumento para o argumento dado, isto é, um argumento inválido com a mesma forma. O argumento
Todos os inteiros são reais
2.5 é um real
∴ 2.5 é um inteiro

é inválido porque tem as premissas verdadeiras e a conclusão falsa. Este argumento é da forma do argumento dado, logo este também é inválido, pelo princípio da forma.

1.1.4. Considere o seguinte argumento:

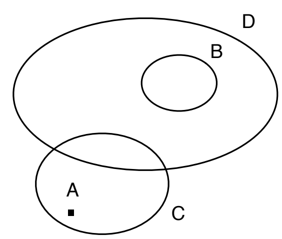
O A é um B ou um C
Os B's são D's
∴ O A é um D

(a) Prove que o argumento não é válido usando apenas a definição de argumento válido.

(b) Prove que o argumento não é válido usando o princípio da forma.

Resposta:

(a) A situação representada abaixo mostra que é possível ter todas as premissas verdadeiras e a conclusão falsa logo, por definição, o argumento não é válido.



(b) Vamos encontrar um contra-argumento para o argumento dado, isto é, um argumento inválido com a mesma forma. O argumento
$\sqrt{2}$ é um inteiro ou um real
Os inteiros são racionais
∴ $\sqrt{2}$ é um racional

é inválido porque tem as premissas verdadeiras e a conclusão falsa. Este argumento é da forma do argumento dado, logo este também é inválido, pelo princípio da forma.

1.1.5. Considere o seguinte argumento:

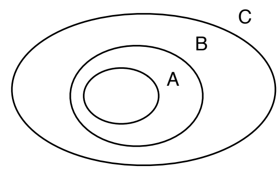
Todos os A's são B's
Todos os B's são C's
∴ Todos os C's são A's

(a) Prove que o argumento não é válido usando apenas a definição de argumento válido.

(b) Prove que o argumento não é válido usando o princípio da forma.

Resposta:

(a) A situação representada abaixo mostra que é possível ter todas as premissas verdadeiras e a conclusão falsa logo, por definição, o argumento não é válido.



(b) Vamos encontrar um contra-argumento para o argumento dado, isto é, um argumento inválido com a mesma forma. O argumento

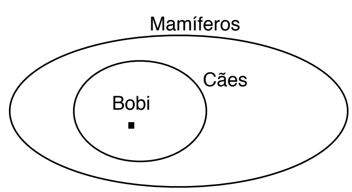
Todos os cães são mamíferos
Todos os mamíferos são animais
$\therefore$ Todos os animais são cães

é inválido porque tem as premissas verdadeiras e a conclusão falsa. Este argumento é da forma do argumento dado, logo este também é inválido, pelo princípio da forma.

1.1.6. Considere o seguinte argumento:

Todos os cães são mamíferos
O Bobi é um mamífero
$\therefore$ O Bobi é um cão

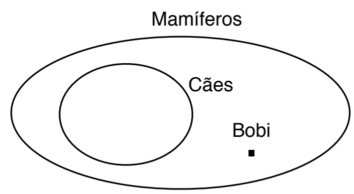
Considere agora o seguinte diagrama:



Será que o diagrama prova que o argumento é válido? Justifique a sua resposta.

Resposta:

Não; o diagrama apenas prova que é possível ter as premissas e a conclusão verdadeiras, o que não basta para o argumento ser válido. Com efeito, podemos imaginar a seguinte situação

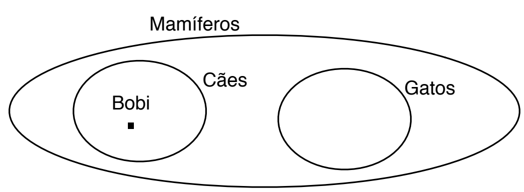


em que as premissas são verdadeiras e a conclusão é falsa. Isto prova que o argumento é inválido.

1.1.7. Considere o seguinte argumento:

Todos os cães são mamíferos
O Bobi é um cão ou um gato
∴ O Bobi é um mamífero

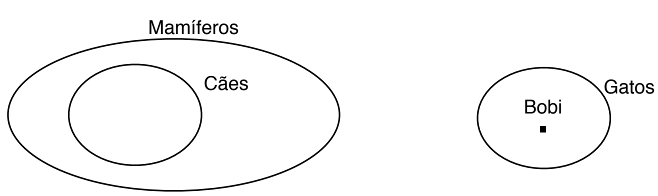
Considere agora o seguinte diagrama:



Será que o diagrama prova que o argumento é válido? Justifique a sua resposta.

Resposta:

Não; o diagrama apenas prova que é possível ter as premissas e a conclusão verdadeiras, o que não basta para o argumento ser válido. Com efeito, podemos imaginar a seguinte situação



em que as premissas são verdadeiras e a conclusão é falsa. Isto prova que o argumento é inválido.

1.1.8. Considere o seguinte argumento:

Todos os cães são mamíferos
Todos os gatos são mamíferos
O Bobi é um mamífero
∴ O Bobi é um cão ou um gato

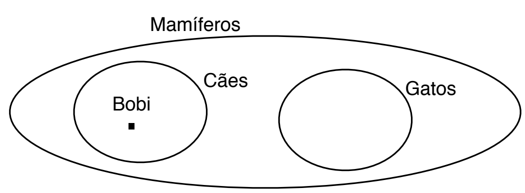
(a) Mostre através de um diagrama que é possível ter todas as premissas verdadeiras e a conclusão verdadeira.

(b) Mostre através de um diagrama que é possível ter todas as premissas verdadeiras e a conclusão falsa.

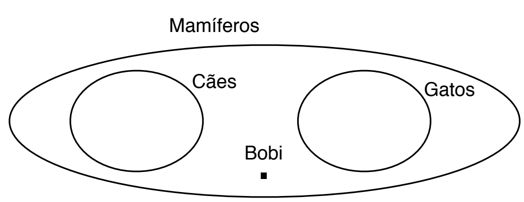
(c) Qual das alíneas anterior lhe permite tirar uma conclusão sobre a validade do argumento? Que conclusão é essa?

Resposta:

(a)



(b)



(c) O argumento é inválido, pois na alínea b) mostrou-se que é possível ter todas as premissas verdadeiras e a conclusão falsa.

1.1.9. Considere que α, β e γ são proposições. Recorda-se que valor lógico da proposição "Se α então β" é falso apenas se o antecedente for verdadeiro e o consequente for falso. Determine a validade dos seguintes argumentos, justificando a sua resposta:

(a) α
Não β
$\therefore$ Se β então γ

(b) α
Não β
$\therefore$ Se γ então α

Resposta:

Assumindo, para cada um dos argumentos, que as premissas são verdadeiras:

(a) "Se β então γ" tem de ser verdadeira, porque β tem de ser falsa. Logo, o argumento é válido.

(b) "Se γ então α" tem de ser verdadeira, porque α tem de ser verdadeira. Logo, o argumento é válido.

1.1.10. Considere que α, β, γ e δ são proposições. Sabendo que o argumento

α
β
$\therefore$ γ e δ

é válido, diga o que pode concluir sobre a validade dos seguintes argumentos (válido, não válido ou não se pode concluir nada).

(a) α
β
$\therefore$ γ ou δ

(b) α
β
$\therefore$ Se γ então δ

(c) α
β
$\therefore$ Não γ

Resposta:

Do enunciado sabemos que se α e β forem verdadeiras, então "γ e δ" também tem de ser verdadeira, ou seja, γ tem de ser verdadeira e δ também. Assumindo, para cada um dos argumentos, que as premissas são verdadeiras:

(a) "γ ou δ" tem de ser verdadeira, porque ambas γ e δ o são. Logo, o argumento é válido.

(b) "Se γ então δ" tem de ser verdadeira, porque δ é verdadeira. Logo, o argumento é válido.

(c) “Não γ" tem de ser falsa, porque γ é verdadeira. Logo, o argumento não é válido.

1.1.11. Considere que α, β, γ, δ e ϵ são proposições. Sabendo que o argumento

α
β
$\therefore$ γ ou δ

é válido, diga o que pode concluir sobre a validade dos seguintes argumentos (válido, não válido ou não se pode concluir nada).

(a) α
β
$\therefore$ γ e δ

(b) α
β
Se γ então ϵ
Se δ então ϵ
$\therefore$ ϵ

Resposta:

Do enunciado sabemos que se α e β forem verdadeiras, então “γ ou δ" também tem de ser verdadeira, ou seja, uma das proposições γ e δ tem de ser verdadeira, ou ambas. Assumindo, para cada um dos argumentos, que as premissas são verdadeiras:

(a) “γ e δ" poderá ter de ser verdadeira, se ambas γ e δ tiverem de ser verdadeiras. Mas se apenas uma de γ e δ tiver de ser verdadeira, “γ e δ" poderá ser falsa. Logo, não podemos concluir nada sobre a validade do argumento.

(b) Sabemos que uma das proposições γ e δ tem de ser verdadeira. Se γ tiver de ser verdadeira, então ϵ também de o ser porque estamos a assumir que “Se γ então ϵ" é verdadeira. Se γ não tiver de ser verdadeira, então δ tem de o ser; então ϵ também de o ser porque estamos a assumir que “Se δ então ϵ" é verdadeira. Logo, o argumento é válido.

1.1.12. Considere que α, β, γ e δ são proposições. Sabendo que o argumento

α
β
$\therefore$ Não γ

é válido, diga o que pode concluir sobre a validade dos seguintes argumentos (válido, não válido ou não se pode concluir nada).

(a) α
β
$\therefore$ Se γ, então δ

(b) α
β
γ
$\therefore$ δ

Resposta:

Do enunciado sabemos que se α e β forem verdadeiras, então "Não γ" também tem de ser verdadeira, ou seja, γ tem de ser falsa. Assumindo, para cada um dos argumentos, que as premissas são verdadeiras:

(a) "Se γ, então δ" tem de ser verdadeira, porque γ tem de ser falsa. Logo, o argumento é válido.

(b) É impossível ter todas as premissas verdadeiras. Logo, o argumento é válido.

1.1.13. Considere o seguinte argumento:

Se α, então β
β
$\therefore\alpha$

(a) Prove que o argumento não é válido usando apenas a definição de argumento válido. Sugestão: atribua valores lógicos às proposições α e β.

(b) Prove que o argumento não é válido usando o princípio da forma.

Resposta:

(a) Se α for falsa e β for verdadeira, então ambas as premissas são verdadeiras e a conclusão é falsa. Logo, o argumento não é válido.

(b) Consideremos o seguinte argumento:
Se π é um inteiro, então π é um real
π é um real
$\therefore$ π é um inteiro

Este argumento não é válido (tem todas as premissas verdadeiras e a conclusão falsa) e tem a mesma forma que argumento inicial. Então, pelo princípio da forma, o argumento inicial não é válido.

1.2 Componentes de uma lógica

1.2.1. Suponha que o argumento (Δ, α) é válido numa determinada lógica. Complete a frase "Se a lógica for __________, então $\Delta \vdash \alpha$.".

Resposta:

Se a lógica for completa, então $\Delta \vdash \alpha$.

1.2.2. (a) Para cada uma das proposições abaixo, diga se é verdadeira ou falsa:

i. Numa lógica não completa, nenhum argumento válido é demonstrável.

ii. Se (Δ, α) é um contra-argumento para o argumento (Δ', α'), então os dois argumentos têm a mesma forma.

(b) Complete cada uma das seguintes proposições:

i. Numa lógica correta, todos os argumentos demonstráveis são __________.

ii. Uma lógica cujo sistema dedutivo permita derivar todas as fórmulas bem formadas é __________.

Resposta:

(a) i. Falsa
ii. Verdadeira

i. válidos
ii. completa

1.2.3. (a) Para cada uma das proposições abaixo, diga se é verdadeira ou falsa:

i. Numa lógica não correta, nenhum argumento demonstrável é válido.

ii. Se (Δ, α) é um contra-argumento para o argumento (Δ', α'), então ambos os argumentos são válidos.

(b) Complete cada uma das seguintes proposições:

i. Numa lógica completa, todos os argumentos válidos são __________.

ii. Uma lógica cujo sistema dedutivo não permita derivar nenhuma fórmula bem formada é __________.

Resposta:

(a) i. Falsa
    ii. Falsa

    i. demonstráveis
    ii. correta

1.2.4. Para cada afirmação, diga se é verdadeira ou falsa (V/F).

(a) O princípio da irrelevância do valor lógico afirma que os valores lógicos das proposições que constituem um argumento não dependem da validade do argumento.
(b) O princípio da irrelevância do valor lógico afirma que o valor lógico de $\alpha \wedge \beta$ não depende dos valores lógicos de α e β.
(c) Numa lógica completa é possível demonstrar todos os argumentos válidos.
(d) O princípio da forma afirma que se dois argumentos têm a mesma forma, então as conclusões dos dois argumentos têm o mesmo valor lógico.

Resposta:

(a) F
(b) F
(c) V
(d) F

1.2.5. Suponha que a linguagem de uma lógica contém o símbolo lógico $\oplus$, correspondente à disjunção exclusiva. Suponha ainda que o sistema dedutivo da lógica em consideração contém a seguinte regra de inferência:

A partir de α e $\alpha \oplus \beta$ podemos inferir β.

Prove que a lógica não é correta.

Resposta:

Para provar que a lógica não é correta vamos provar que existe um argumento demonstrável que não é válido. Consideremos o argumento $(\{A, A \oplus B\}, B)$[1]. O argumento é demonstrável pois existe uma prova da conclusão a partir das premissas:

[1]Supondo que A e B são *fbfs* da lógica.

A
$A \oplus B$
B

Trata-se de uma prova pois as *fbfs* nas duas primeiras linhas correspondem a premissas, e a *fbf* da última linha corresponde à aplicação de uma regra de inferência a *fbfs* anteriores da prova.

No entanto, o argumento não é válido pois existe uma interpretação I que torna todas as premissas verdadeiras e a conclusão falsa: $I(A) = V$, $I(B) = F$. Note-se que, segundo esta interpretação, a *fbf* $A \oplus B$ é verdadeira, pois exactamente uma das proposições A e B é verdadeira.

Capítulo 2

Lógica proposicional (I)

2.1 Sistema dedutivo

2.1.1. Demonstre os argumentos abaixo usando as regras de inferência do sistema dedutivo da Lógica Proposicional que estudou. ([1], Cap.2, exercício 5).

(a) $(\{P \to (P \to Q), P\}, Q)$

(b) $(\{P \to Q, Q \to R\}, P \to R\}$

(c) $(\{P\}, Q \to (P \land Q))$

(d) $(\{P \to (Q \lor R), Q \to S, R \to S\}, P \to S)$

Resposta:

(a)

1	$P \to (P \to Q)$	Prem
2	P	Prem
3	$P \to Q$	E$\to$, (1, 2)
4	Q	E$\to$, (2, 3)

(b)

1	$P \to Q$	Prem
2	$Q \to R$	Prem
3	$\quad \mid \underline{P}$	Hip
4	$\quad \mid P \to Q$	Rei, 1
5	$\quad \mid Q$	E→, (3, 4)
6	$\quad \mid Q \to R$	Rei, 2
7	$\quad \mid R$	E→, (5, 6)
8	$P \to R$	I→, (3, 7)

(c)

1	P	Prem
2	$\quad \mid \underline{Q}$	Hip
3	$\quad \mid P$	Rei, 1
4	$\quad \mid Q$	Rep, 2
5	$\quad \mid P \wedge Q$	I∧, (3, 4)
6	$Q \to (P \wedge Q)$	I→, (2, 5)

(d)

1	$P \to (Q \vee R)$	Prem
2	$Q \to S$	Prem
3	$R \to S$	Prem
4	$\quad \mid \underline{P}$	Hip
5	$\quad \mid P \to (Q \vee R)$	Rei, 1
6	$\quad \mid Q \vee R$	E→, (4, 5)
7	$\quad \mid \quad \mid \underline{Q}$	Hip
8	$\quad \mid \quad \mid Q \to S$	Rei, 2
9	$\quad \mid \quad \mid S$	E→, (7, 8)
10	$\quad \mid \quad \mid \underline{R}$	Hip
11	$\quad \mid \quad \mid R \to S$	Rei, 3
12	$\quad \mid \quad \mid S$	E→, (10, 11)
13	$\quad \mid S$	E∨, (6, (7, 9), (10, 12))
14	$P \to S$	I→, (4, 13)

2.1.2. Demonstre as afirmações abaixo usando as regras de inferência do sistema dedutivo da Lógica Proposicional que estudou.

(a) $\{P \wedge (Q \vee R)\} \vdash (P \wedge Q) \vee (P \wedge R)$

(b) $\{(P \wedge Q) \vee (P \wedge R)\} \vdash P \wedge (Q \vee R)$

(c) $\{P \vee (Q \wedge R)\} \vdash (P \vee Q) \wedge (P \vee R)$

(d) $\{(P \wedge Q) \vee (P \wedge R)\} \vdash P$

(e) $((P \to Q) \wedge (P \to \neg Q)) \to \neg P$ é um teorema.

(f) $\{(P \vee Q) \to R\} \vdash (P \to R) \wedge (Q \to R)$

(g) $\{\neg P\} \vdash P \to Q$

(h) $\{P, \neg Q\} \vdash \neg(P \wedge Q)$

(i) $\{P \to R, Q \to R\} \vdash (P \vee Q) \to R$

(j) $\{P \to (Q \to R), S \to (Q \to R), P \vee S\} \vdash Q \to R$

(k) $\{\neg P \vee Q\} \vdash P \to Q$

(l) $(P \wedge Q) \to (P \vee R)$ é um teorema.

(m) $\{P \to Q, P \to R\} \vdash P \to (Q \wedge R)$.

(n) $\{P \vee Q, P \to R, Q \to R\} \vdash R$.

(o) $\{P \vee Q, P \to R, Q \to S\} \vdash R \vee S$.

(p) $\{P \to Q\} \vdash \neg(P \wedge \neg Q)$.

(q) $\{(P \wedge Q) \to (R \wedge S)\} \vdash (P \wedge Q) \to (R \vee S)$.

(r) $\{\neg P, \neg P \to P\} \vdash Q$

Resposta:

(a) Teremos de provar $(P \wedge Q) \vee (P \wedge R)$, a partir da premissa $P \wedge (Q \vee R)$:

1	$P \wedge (Q \vee R)$	Prem
2	P	E$\wedge$, 1
3	$Q \vee R$	E$\wedge$, 1
4	$\quad Q$	Hip
5	$\quad P$	Rei, 2
6	$\quad Q$	Rep, 4
7	$\quad P \wedge Q$	I$\wedge$, (5, 6)
8	$\quad (P \wedge Q) \vee (P \wedge R)$	I$\vee$, 7
9	$\quad R$	Hip
10	$\quad P$	Rei, 2
11	$\quad R$	Rep, 9
12	$\quad P \wedge R$	I$\wedge$, (5, 6)
13	$\quad (P \wedge Q) \vee (P \wedge R)$	I$\vee$, 12
14	$(P \wedge Q) \vee (P \wedge R)$	E$\vee$, (3, (4, 8), (9, 13))

(b) Teremos de provar $P \wedge (Q \vee R)$, a partir da premissa $(P \wedge Q) \vee (P \wedge R)$:

1	$(P \wedge Q) \vee (P \wedge R)$	Prem
2	$\quad P \wedge Q$	Hip
3	$\quad P$	E$\wedge$, 2
4	$\quad Q$	E$\wedge$, 2
5	$\quad Q \vee R$	I$\vee$, 4
6	$\quad P \wedge (Q \vee R)$	I$\wedge$, (3, 5)
7	$\quad P \wedge R$	Hip
8	$\quad P$	E$\wedge$, 7
9	$\quad R$	E$\wedge$, 7
10	$\quad Q \vee R$	I$\vee$, 9
11	$\quad P \wedge (Q \vee R)$	I$\vee$, (8, 10)
12	$P \wedge (Q \vee R)$	E$\vee$, (1, (2, 6), (7, 11))

(c) Teremos de provar $(P \vee Q) \wedge (P \vee R)$, a partir da premissa $P \vee (Q \wedge R)$:

1	$P \vee (Q \wedge R)$	Prem
2	$\quad \mid P$	Hip
3	$\quad \mid P \vee Q$	I$\vee$, 2
4	$\quad \mid P \vee R$	I$\vee$, 2
5	$\quad \mid (P \vee Q) \wedge (P \vee R)$	I$\wedge$, (3, 4)
6	$\quad \mid Q \wedge R$	Hip
7	$\quad \mid Q$	E$\wedge$, 6
8	$\quad \mid R$	E$\wedge$, 6
9	$\quad \mid P \vee Q$	I$\vee$, 7
10	$\quad \mid P \vee R$	I$\vee$, 8
11	$\quad \mid (P \vee Q) \wedge (P \vee R)$	I$\wedge$, (9, 10)
12	$(P \vee Q) \wedge (P \vee R)$	E$\vee$, (1, (2, 5), (6, 11))

(d) Teremos de provar P a partir da premissa $(P \wedge Q) \vee (P \wedge R)$.

1	$(P \wedge Q) \vee (P \wedge R)$	Prem
2	$\quad \mid P \wedge Q$	Hip
3	$\quad \mid P$	E$\wedge$, 2
4	$\quad \mid P \wedge R$	Hip
5	$\quad \mid P$	E$\wedge$, 4
6	P	E$\vee$, (1, (2, 3), (4, 5))

(e) Teremos de provar $((P \to Q) \wedge (P \to \neg Q)) \to \neg P$ a partir de um conjunto de premissas vazio.

1	$(P \to Q) \wedge (P \to \neg Q)$	Hip
2	$P \to Q$	E∧, 1
3	$P \to \neg Q$	E∧, 1
4	P	Hip
5	$P \to Q$	Rei, 2
6	Q	E→, (4, 5)
7	$P \to \neg Q$	Rei, 3
8	$\neg Q$	E→, (4, 7)
9	$\neg P$	I¬, (4, (6, 8)
10	$((P \to Q) \wedge (P \to \neg Q)) \to \neg P$	I→, (1, 9)

(f) Teremos de provar $(P \to R) \wedge (Q \to R)$ a partir da premissa $(P \vee Q) \to R$.

1	$(P \vee Q) \to R$	Prem
2	P	Hip
3	$P \vee Q$	I∨, 2
4	$(P \vee Q) \to R$	Rei, 1
5	R	E→, (3, 4)
6	$P \to R$	I→, (2, 5)
7	Q	Hip
8	$P \vee Q$	I∨, 7
9	$(P \vee Q) \to R$	Rei, 1
10	R	E→, (3, 4)
11	$Q \to R$	I→, (7, 10)
12	$(P \to R) \wedge (Q \to R)$	I∧, (6, 11)

(g) Teremos de provar $P \to Q$ a partir da premissa $\neg P$.

1	$\neg P$	Prem
2	$\quad \mid P$	Hip
3	$\quad \mid \quad \mid \neg Q$	Hip
4	$\quad \mid \quad \mid P$	Rei, 2
5	$\quad \mid \quad \mid \neg P$	Rei, 1
6	$\quad \mid \neg\neg Q$	I$\neg$, (3, (4, 5))
7	$\quad \mid Q$	E$\neg$, 6
8	$P \to Q$	I$\to$, (3, 7)

(h) Teremos de provar $\neg(P \wedge Q)$ a partir das premissas $\{P, \neg Q\}$.

1	P	Prem
2	$\neg Q$	Prem
3	$\quad \mid P \wedge Q$	Hip
4	$\quad \mid Q$	E$\wedge$, 3
5	$\quad \mid \neg Q$	Rei, 2
6	$\neg(P \wedge Q)$	I$\neg$, (3, (4, 5))

(i) Teremos de provar $(P \vee Q) \to R$ a partir das premissas $\{P \to R, Q \to R\}$.

1	$P \to R$	Prem
2	$Q \to R$	Prem
3	$\quad \mid P \vee Q$	Hip
4	$\quad \mid \quad \mid P$	Hip
5	$\quad \mid \quad \mid P \to R$	Rei, 1
6	$\quad \mid \quad \mid R$	E$\to$, (4, 5)
7	$\quad \mid \quad \mid Q$	Hip
8	$\quad \mid \quad \mid Q \to R$	Rei, 2
9	$\quad \mid \quad \mid R$	E$\to$, (7, 8)
10	$\quad \mid R$	E$\vee$, (3, (4, 6), (7, 9))
11	$(P \vee Q) \to R$	I$\to$, (3, 10)

(j) Teremos de provar $Q \to R$ a partir das premissas $\{P \to (Q \to R), S \to (Q \to R)\}$.

1	$P \to (Q \to R)$	Prem
2	$S \to (Q \to R)$	Prem
3	$P \lor S$	Prem
4	$\quad \mid \underline{P}$	Hip
5	$\quad \mid P \to (Q \to R)$	Rei, 1
6	$\quad \mid Q \to R$	E$\to$, (4, 5)
7	$\quad \mid \underline{S}$	Hip
8	$\quad \mid S \to (Q \to R)$	Rei, 2
9	$\quad \mid Q \to R$	E$\to$, (7, 8)
10	$Q \to R$	E$\lor$, (3, (4, 6), (7, 9))

(k) Teremos de provar $P \to Q$ a partir da premissa $\neg P \lor Q$.

1	$\neg P \lor Q$	Prem
2	$\quad \mid \underline{\neg P}$	Hip
3	$\quad \mid \quad \mid \underline{P}$	Hip
4	$\quad \mid \quad \mid \quad \mid \underline{\neg Q}$	Hip
5	$\quad \mid \quad \mid \quad \mid P$	Rei, 3
6	$\quad \mid \quad \mid \quad \mid \neg P$	Rei, 2
7	$\quad \mid \quad \mid \neg\neg Q$	I$\neg$, (4, (5, 6))
8	$\quad \mid \quad \mid Q$	E$\neg$, 7
9	$\quad \mid P \to Q$	I$\to$, (3, 8)
10	$\quad \mid \underline{Q}$	Hip
11	$\quad \mid \quad \mid \underline{P}$	Hip
12	$\quad \mid \quad \mid Q$	Rei, 10
13	$\quad \mid P \to Q$	I$\to$, (11, 12)
14	$P \to Q$	I$\lor$, 1, (2, 9), (10, 13)

(l) Teremos de provar $(P \land Q) \to (P \lor R)$, a partir de um conjunto de premissas vazio:

1	$P \wedge Q$	Hip
2	P	E$\wedge$, 1
3	$P \vee R$	I$\vee$, 2
4	$(P \wedge Q) \to (P \vee R)$	I$\to$, (1, 3)

(m) Teremos de provar $P \to (Q \wedge R)$, a partir das premissas $\{P \to Q, P \to R\}$:

1	$P \to Q$	Prem
2	$P \to R$	Prem
3	P	Hip
4	$P \to Q$	Rei, 1
5	Q	E$\to$, 3, 4
6	$P \to R$	Rei, 2
7	R	E$\to$, 3, 6
8	$Q \wedge R$	I$\wedge$, 5, 7
9	$P \to (Q \wedge R)$	I$\to$, (3, 8)

(n) Teremos de provar R, a partir das premissas $\{P \vee Q, P \to R, Q \to R\}$:

1	$P \vee Q$	Prem
2	$P \to R$	Prem
3	$Q \to R$	Prem
4	P	Hip
5	$P \to R$	Rei, 2
6	R	E$\to$, 4, 5
7	Q	Hip
8	$Q \to R$	Rei, 3
9	R	E$\to$, 7, 8
10	R	E$\vee$, (1, (4, 6), (7, 9))

(o) Teremos de provar $R \vee S$, a partir das premissas

$\{P \vee Q, P \to R, Q \to S\}$:

1	$P \vee Q$	Prem
2	$P \to R$	Prem
3	$Q \to S$	Prem
4	P	Hip
5	$P \to R$	Rei, 2
6	R	E$\to$, 4, 5
7	$R \vee S$	I$\vee$, 6
8	Q	Hip
9	$Q \to S$	Rei, 3
10	S	E$\to$, 8, 9
11	$R \vee S$	I$\vee$, 10
12	$R \vee S$	E$\vee$, (1, (4, 7), (8, 11))

(p) Teremos de provar $\neg(P \wedge \neg Q)$, a partir da premissa $P \to Q$:

1	$P \to Q$	Prem
2	$P \wedge \neg Q$	Hip
3	P	I$\wedge$, 2
4	$P \to Q$	Rei, 1
5	Q	E$\to$, 3, 4
6	$\neg Q$	E$\wedge$, 2
7	$\neg(P \wedge \neg Q)$	I$\neg$, (2, (5, 6))

(q)

1	$(P \wedge Q) \to (R \wedge S)$	Prem
2	$P \wedge Q$	Hip
3	$(P \wedge Q) \to (R \wedge S)$	Rei, 2
4	$R \wedge S$	E$\to$, (2, 3)
5	R	E$\wedge$, 4
6	$R \vee S$	I$\vee$, 5
7	$(P \wedge Q) \to (R \vee S)$	I$\to$, (2, 6)

(r) Teremos de provar Q a partir das premissas $\{\neg P, \neg P \to P\}$:

1	$\neg P$	Prem
2	$\neg P \to P$	Prem
3	P	E$\to$, (1, 2)
4	$\quad\vert\ \neg Q$	Hip
5	$\quad\vert\ P$	Rei, 3
6	$\quad\vert\ \neg P$	Rei, 1
7	$\neg\neg Q$	I$\neg$, (4, (5, 6))
8	Q	E$\neg$, 7

2.1.3. Demonstre as afirmações abaixo usando as regras de inferência do sistema dedutivo da Lógica Proposicional que estudou. Em cada alínea, pode usar resultados das alíneas anteriores. Pode também usar as seguintes regras de inferência, derivadas das primeiras leis de De Morgan:

- A partir de $\neg(\alpha \lor \beta)$, podemos inferir $\neg\alpha \land \neg\beta$:

n	$\neg(\alpha \lor \beta)$	
$\vdots$	$\vdots$	
m	$\neg\alpha \land \neg\beta$	De Morgan (v1), n

- A partir de $\neg(\alpha \land \beta)$, podemos inferir $\neg\alpha \lor \neg\beta$:

n	$\neg(\alpha \land \beta)$	
$\vdots$	$\vdots$	
m	$\neg\alpha \lor \neg\beta$	De Morgan (v2), n

(a) $((P \to Q) \land (P \to \neg Q)) \to \neg P$ é um teorema.

(b) $((P \to Q) \land (\neg P \to Q)) \to Q$ é um teorema.

(c) $(\{(P \lor Q) \land (P \lor R)\}, P \lor (Q \land R))$ é um argumento demonstrável.
Sugestão: use os teoremas dos exemplos 2.2.15 e 2.2.22, das págs 42 e 50 do livro, respetivamente.

(d) $((P \to R) \lor (Q \to R)) \to ((P \land Q) \to R)$ é um teorema.

(e) $(\neg P \lor \neg\neg R) \to (P \to R)$ é um teorema.
Sugestão: Derive a seguinte regra de inferência: a partir de α e $\neg\alpha \lor \beta$, podemos inferir β:

n	α	
$\vdots$	$\vdots$	
m	$\neg\alpha \lor \beta$	
$\vdots$	$\vdots$	
k	β	E$\lor'$, (n, m)

e em seguida use esta regra para demonstrar o teorema $(\neg P \vee \neg\neg R) \rightarrow (P \rightarrow R)$.

(f) $\neg(P \rightarrow R) \rightarrow (P \wedge \neg R)$ é um teorema. Sugestão: use o teorema demonstrado no exercício 2.1.3e.

(g) $((P \wedge Q) \rightarrow R) \rightarrow ((P \rightarrow R) \vee (Q \rightarrow R))$ é um teorema. Sugestão: use o teorema demonstrado no exercício 2.1.3f.

(h) $(\{P \rightarrow Q\}, \neg P \vee Q)$ é um argumento demonstrável.

(i) $\neg P \vee (Q \rightarrow P)$ é um teorema.

Resposta:

(a) Para $((P \rightarrow Q) \wedge (P \rightarrow \neg Q)) \rightarrow \neg P$ ser um teorema tem de ser derivável a partir de um conjunto de premissas vazio:

1	$(P \rightarrow Q) \wedge (P \rightarrow \neg Q)$	Hip
2	$P \rightarrow Q$	E$\wedge$, 1
3	$P \rightarrow \neg Q$	E$\wedge$, 1
4	P	Hip
5	$P \rightarrow Q$	Rei, 2
6	Q	E$\rightarrow$, (4, 5)
7	$P \rightarrow \neg Q$	Rei, 3
8	$\neg Q$	E$\rightarrow$, (4, 7)
9	$\neg P$	I$\neg$, (4, (6, 8)
10	$((P \rightarrow Q) \wedge (P \rightarrow \neg Q)) \rightarrow \neg P$	I$\rightarrow$, (1, 9)

(b) Para $((P \rightarrow Q) \wedge (\neg P \rightarrow Q)) \rightarrow Q$ ser um teorema tem de ser derivável a partir de um conjunto de premissas vazio:

1	$(P \rightarrow Q) \wedge (\neg P \rightarrow Q)$	Hip
2	$P \rightarrow Q$	E$\wedge$, 1
3	$\neg P \rightarrow Q$	E$\wedge$, 1
4	$\quad \neg Q$	Hip
5	$\quad\quad P$	Hip
6	$\quad\quad P \rightarrow Q$	Rei, 2
7	$\quad\quad Q$	E$\rightarrow$, (5, 6)
8	$\quad\quad \neg Q$	Rei, 4
9	$\quad \neg P$	I$\neg$, (5, (7, 8))
10	$\quad\quad \neg P$	Hip
11	$\quad\quad \neg P \rightarrow Q$	Rei, 3
12	$\quad\quad Q$	E$\rightarrow$, (10, 11)
13	$\quad\quad \neg Q$	Rei, 4
14	$\quad \neg\neg P$	I$\neg$, (10, (12, 13))
15	$\quad P$	E$\neg$, 14
16	$\neg\neg Q$	I$\neg$, (4, (9, 15))
17	Q	E$\neg$, 16
18	$((P \rightarrow Q) \wedge (\neg P \rightarrow Q)) \rightarrow Q$	I$\rightarrow$, (1, 17)

(c) Para o argumento $(\{(P \vee Q) \wedge (P \vee R)\}, P \vee (Q \wedge R)$ ser demonstrável tem de existir uma prova da conclusão a partir da premissa:

1	$(P \vee Q) \wedge (P \vee R)$	Prem
2	$P \vee \neg P$	Teo
3	$\mid\ P$	Hip
4	$\mid\ P \vee (Q \wedge R)$	I$\vee$, 2
5	$\mid\ \neg P$	Hip
6	$\mid\ (P \vee Q) \wedge (P \vee R)$	Rei, 1
7	$\mid\ P \vee R$	E$\wedge$, 6
8	$\mid\ \neg P$	Rep, 5
9	$\mid\ (P \vee R) \wedge \neg P$	I$\wedge$, (7, 8)
10	$\mid\ ((P \vee R) \wedge \neg P) \rightarrow R$	Teo
11	$\mid\ R$	E$\rightarrow$, (9, 10)
12	$\mid\ P \vee Q$	E$\wedge$, 6
13	$\mid\ \neg P$	Rep, 5
14	$\mid\ (P \vee Q) \wedge \neg P$	I$\wedge$, (12, 13)
15	$\mid\ ((P \vee Q) \wedge \neg P) \rightarrow Q$	Teo
16	$\mid\ Q$	E$\rightarrow$, (14, 15)
17	$\mid\ R$	Rep, 11
18	$\mid\ Q \wedge R$	I$\wedge$, (16, 17)
19	$\mid\ P \vee (Q \wedge R)$	I$\vee$, 18
20	$P \vee (Q \wedge R)$	E$\vee$, (2, (3, 4), (5, 19))

(d) Para $((P \rightarrow R) \vee (Q \rightarrow R)) \rightarrow ((P \wedge Q) \rightarrow R)$ ser um teorema tem de

ser derivável a partir de um conjunto de premissas vazio:

1	$\vert\ (P \to R) \lor (Q \to R)$	Hip
2	$\vert\ \vert\ P \land Q$	Hip
3	$\vert\ \vert\ P$	E$\land$, 2
4	$\vert\ \vert\ Q$	E$\land$, 2
5	$\vert\ \vert\ (P \to R) \lor (Q \to R)$	Rei, 1
6	$\vert\ \vert\ \vert\ P \to R$	Hip
7	$\vert\ \vert\ \vert\ P$	Rei, 3
8	$\vert\ \vert\ \vert\ P \to R$	Rep, 6
9	$\vert\ \vert\ \vert\ R$	E$\to$, (7, 8)
10	$\vert\ \vert\ \vert\ Q \to R$	Hip
11	$\vert\ \vert\ \vert\ Q$	Rei, 4
12	$\vert\ \vert\ \vert\ Q \to R$	Rep, 10
13	$\vert\ \vert\ \vert\ R$	E$\to$, (11, 12)
14	$\vert\ \vert\ R$	E$\lor$, (5, (6, 9), (10, 13))
15	$\vert\ (P \land Q) \to R$	I$\to$, (2, 14)
16	$((P \to R) \lor (Q \to R)) \to ((P \land Q) \to R)$	I$\to$, (1, 15)

(e) Para o argumento $(\{(P \lor Q) \land (P \lor R)\}, P \lor (Q \land R)$ ser demonstrável tem de existir uma prova da conclusão a partir da premissa:

1	$(P \vee Q) \wedge (P \vee R)$	Prem
2	$P \vee \neg P$	Teo
3	$\quad \mid \underline{P}$	Hip
4	$\quad \mid P \vee (Q \wedge R)$	I$\vee$, 2
5	$\quad \mid \underline{\neg P}$	Hip
6	$\quad \mid (P \vee Q) \wedge (P \vee R)$	Rei, 1
7	$\quad \mid P \vee R$	E$\wedge$, 6
8	$\quad \mid \neg P$	Rep, 5
9	$\quad \mid (P \vee R) \wedge \neg P$	I$\wedge$, (7, 8)
10	$\quad \mid ((P \vee R) \wedge \neg P) \to R$	Teo
11	$\quad \mid R$	E$\to$, (9, 10)
12	$\quad \mid P \vee Q$	E$\wedge$, 6
13	$\quad \mid \neg P$	Rep, 5
14	$\quad \mid (P \vee Q) \wedge \neg P$	I$\wedge$, (12, 13)
15	$\quad \mid ((P \vee Q) \wedge \neg P) \to Q$	Teo
16	$\quad \mid Q$	E$\to$, (14, 15)
17	$\quad \mid R$	Rep, 11
18	$\quad \mid Q \wedge R$	I$\wedge$, (16, 17)
19	$\quad \mid P \vee (Q \wedge R)$	I$\vee$, 18
20	$P \vee (Q \wedge R)$	E$\vee$, (2, (3, 4), (5, 19))

(f) Para $((P \to R) \vee (Q \to R)) \to ((P \wedge Q) \to R)$ ser um teorema tem de

ser derivável a partir de um conjunto de premissas vazio:

1	$(P \to R) \vee (Q \to R)$	Hip
2	$P \wedge Q$	Hip
3	P	E$\wedge$, 2
4	Q	E$\wedge$, 2
5	$(P \to R) \vee (Q \to R)$	Rei, 1
6	$P \to R$	Hip
7	P	Rei, 3
8	$P \to R$	Rep, 6
9	R	E$\to$, (7, 8)
10	$Q \to R$	Hip
11	Q	Rei, 4
12	$Q \to R$	Rep, 10
13	R	E$\to$, (11, 12)
14	R	E$\vee$, (5, (6, 9), (10, 13))
15	$(P \wedge Q) \to R$	I$\to$, (2, 14)
16	$((P \to R) \vee (Q \to R)) \to ((P \wedge Q) \to R)$	I$\to$, (1, 15)

(g) Derivamos primeiro a seguinte regra de inferência: a partir de α e $\neg\alpha \vee \beta$, podemos inferir β:

n	α	
$\vdots$	$\vdots$	
m	$\neg\alpha \vee \beta$	
$\vdots$	$\vdots$	
k	β	E$\vee'$, $(n,\ m)$

1	α	
2	$\neg\alpha \vee \beta$	
3	$\mid\ \neg\alpha$	Hip
4	$\mid\ \mid\ \neg\beta$	Hip
5	$\mid\ \mid\ \alpha$	Rei, 1
6	$\mid\ \mid\ \neg\alpha$	Rei, 3
7	$\mid\ \neg\neg\beta$	I$\neg$, (4, (5, 6))
8	$\mid\ \beta$	E$\neg$, 7
9	$\mid\ \beta$	Hip
10	$\mid\ \beta$	Rep, 9
11	β	E$\vee$, (2, (3, 8), (9, 10))

Para $(\neg P \vee \neg\neg R) \rightarrow (P \rightarrow R)$ ser um teorema tem de ser derivável a partir de um conjunto de premissas vazio:

1	$\mid\ \neg P \vee \neg\neg R$	Hip
2	$\mid\ \mid\ P$	Hip
3	$\mid\ \mid\ \neg P \vee \neg\neg R$	Rei, 1
4	$\mid\ \mid\ \neg\neg R$	E$\vee'$, (2, 3)
5	$\mid\ \mid\ R$	E$\neg$, 4
6	$\mid\ P \rightarrow R$	I$\rightarrow$, (2, 5)
7	$(\neg P \vee \neg\neg R) \rightarrow (P \rightarrow R)$	I$\rightarrow$, (1, 6)

(h) Para $\neg(P \rightarrow R) \rightarrow (P \wedge \neg R)$ ser um teorema tem de ser derivável a partir de um conjunto de premissas vazio:

1	$\quad\mid \neg(P \to R)$	Hip
2	$\quad\mid\quad\mid \neg(P \wedge \neg R)$	Hip
3	$\quad\mid\quad\mid \neg P \vee \neg\neg R$	De Morgan (v2), 2
4	$\quad\mid\quad\mid (\neg P \vee \neg\neg R) \to (P \to R)$	Teo
5	$\quad\mid\quad\mid P \to R$	E$\to$, (3, 4)
6	$\quad\mid\quad\mid \neg(P \to R)$	Rei, 1
7	$\quad\mid \neg\neg(P \wedge \neg R)$	I$\neg$, (2, (5, 6))
8	$\quad\mid P \wedge \neg R$	E$\neg$, 7
9	$\neg(P \to R) \to (P \wedge \neg R)$	I$\to$, (1, 8)

O teorema usado na prova acima foi demonstrado na alínea 2.1.3e.

(i) Para $((P \wedge Q) \to R) \to ((P \to R) \vee (Q \to R))$ ser um teorema tem de

ser derivável a partir de um conjunto de premissas vazio:

1	$(P \wedge Q) \to R$	Hip
2	$\quad \neg((P \to R) \vee (Q \to R))$	Hip
3	$\quad \neg(P \to R) \wedge \neg(Q \to R)$	De Morgan (v1), 2
4	$\quad \neg(P \to R)$	E$\wedge$, 3
5	$\quad \neg(P \to R) \to (P \wedge \neg R)$	Teo
6	$\quad P \wedge \neg R$	E$\to$, (4, 5)
7	$\quad \neg(Q \to R)$	E$\wedge$, 3
8	$\quad \neg(Q \to R) \to (Q \wedge \neg R)$	Teo
9	$\quad Q \wedge \neg R$	E$\to$, (7, 8)
10	$\quad P$	E$\wedge$, 6
11	$\quad Q$	E$\wedge$, 9
12	$\quad P \wedge Q$	I$\wedge$, (10, 11)
13	$\quad (P \wedge Q) \to R$	Rei, 1
14	$\quad R$	E$\to$, (12, 13)
15	$\quad \neg R$	E$\wedge$, 9
16	$(P \to R) \vee (Q \to R)$	I$\neg$, (2, (14, 15))
17	$((P \wedge Q) \to R) \to ((P \to R) \vee (Q \to R))$	I$\to$, (1, 16)

O teorema usado na prova acima foi demonstrado no exercício 2.1.3f.

(j) Para o argumento $(\{P \to Q\}, \neg P \vee Q)$ ser demonstrável tem de existir uma prova da conclusão a partir da premissa:

1	$P \rightarrow Q$	Prem
2	$\quad \neg(\neg P \vee Q)$	Hip
3	$\quad \neg\neg P \wedge \neg Q$	De Morgan (v1), 2
4	$\quad \neg\neg P$	E$\wedge$, 3
5	$\quad P$	E$\neg$, 4
6	$\quad P \rightarrow Q$	Rei, 1
7	$\quad Q$	E$\rightarrow$, (5, 6)
8	$\quad \neg Q$	E$\wedge$, 3
9	$\neg\neg(\neg P \vee Q)$	I$\neg$, (2, (7, 8))
10	$\neg P \vee Q$	E$\neg$, 9

(k) Para $\neg P \vee (Q \rightarrow P)$ ser um teorema tem de ser derivável a partir de um conjunto de premissas vazio:

1	$\quad \neg P \vee (Q \rightarrow P)$	Hip
2	$\quad \neg\neg P \wedge \neg Q$	De Morgan (v1), 1
3	$\quad \neg\neg P$	E$\wedge$, 2
4	$\quad P$	E$\neg$, 3
5	$\quad P \rightarrow Q$	Rei, 1
6	$\quad Q$	E$\rightarrow$, (4, 5)
7	$\quad \neg Q$	E$\wedge$, 2
8	$\neg\neg(\neg P \vee Q)$	I$\neg$, (1, (6, 7))
9	$\neg P \vee Q$	E$\neg$, 8

2.1.4. Sabendo que a *fbf* $((P \vee Q) \wedge \neg P) \rightarrow Q$ é um teorema, demonstre o argumento $(\{(P \vee Q) \wedge \neg P\}, Q)$, usando apenas as propriedades do sistema dedutivo da lógica proposicional.

Resposta:

Se $((P \vee Q) \wedge \neg P) \rightarrow Q$ é um teorema, então $\{\} \vdash ((P \vee Q) \wedge \neg P) \rightarrow Q$.
Consideremos o teorema
"Para qualquer conjunto de *fbfs* Δ e quaisquer *fbfs* α e β, se $\Delta \vdash (\alpha \rightarrow \beta)$, então $(\Delta \cup \{\alpha\}) \vdash \beta$."

Então, sendo $\Delta = \{\}$, $\alpha = (P \vee Q) \wedge \neg P$ e $\beta = Q$, temos que $\{(P \vee Q) \wedge \neg P\} \vdash Q$, ou seja, o argumento $(\{(P \vee Q) \wedge \neg P\}, Q)$ é demonstrável.

2.1.5. Complete as seguintes frases, com uma das palavras *transitividade*, *dedução* ou *monotonicidade*.

(a) Sabendo que $\{P, P \rightarrow Q\} \vdash Q$, podemos garantir que $\{P, P \rightarrow Q, R\} \vdash Q$ pelo teorema da ______________.

(b) Sabendo que $\{P, P \rightarrow Q, Q \rightarrow R\} \vdash Q$, $\{P, P \rightarrow Q, Q \rightarrow R\} \vdash R$ e $\{Q, R\} \vdash R \rightarrow Q$ podemos garantir que $\{P, P \rightarrow Q, Q \rightarrow R\} \vdash R \rightarrow Q$ pelo teorema da ______________.

Resposta:

(a) monotonicidade.

(b) transitividade.

2.2 Sistema semântico

2.2.1. Sendo α_1, α_2 e α_3 *fbfs* da Lógica Proposicional, escolha a única alternativa que torna incorreta a seguinte afirmação:
"$\Delta = \{\alpha_1, \alpha_2, \alpha_3\}$ não é satisfazível se e só se
A. Nenhuma das *fbfs* de Δ é satisfazível.
B. A *fbf* $\alpha_1 \wedge \alpha_2 \wedge \alpha_3$ não é satisfazível.
C. $\{\alpha_1, \alpha_2\} \models \neg\alpha_3$."

Resposta:

A

2.2.2. Classifique as seguintes afirmações em sempre verdadeira (**V**), sempre falsa (**F**), ou possivelmente verdadeira (**P**):

(a) A *fbf* α é tautológica e falsificável.

(b) A *fbf* α é satisfazível e falsificável.

(c) A *fbf* α é satisfazível e contraditória.

(d) A *fbf* α é satisfazível ou falsificável.

(e) A *fbf* α é tautológica ou contraditória.

Resposta:

(a) A *fbf* α é tautológica e falsificável.
Resp: **F**

(b) A *fbf* α é satisfazível e falsificável.
Resp: **P**

(c) A *fbf* α é satisfazível e contraditória.
Resp: **F**

(d) A *fbf* α é satisfazível ou falsificável.
Resp: **V**

(e) A *fbf* α é tautológica ou contraditória.
Resp: **P**

2.2.3. Complete a frase seguinte:
"Numa lógica _______ (completa/correta), se $\Delta \not\models \alpha$, então $\Delta \nvdash \alpha$".

Resposta:

"Numa lógica correta, se $\Delta \not\models \alpha$, então $\Delta \nvdash \alpha$".

2.2.4. Prove que $\alpha = P \to R$ não é uma consequência semântica do conjunto $\Delta = \{(P \to R) \vee (Q \to R)\}$.

Resposta:

A interpretação
$I(P) = V$, $I(Q) = F$, $I(R) = F$
é um modelo de Δ, e não satisfaz α. Logo α não é uma consequência semântica de Δ.

2.2.5. Determine os modelos dos seguintes conjuntos de *fbfs* . Para cada conjunto, indique todas as *fbf* atómicas que sejam consequências semânticas do conjunto, se tais *fbf* existirem.

(a) $\{\neg P \to Q, \neg Q\}$.

(b) $\{(P \wedge Q) \to R, P, \neg R\}$.

(c) $\{P \to R, Q \to R, P \vee Q\}$.

Resposta:

(a) $\{\neg P \to Q, \neg Q\}$.

Interpretação	P	Q	$\neg P$	$\neg P \to Q$	$\neg Q$
I_1	V	V	F	V	F
I_2	V	F	F	V	V
I_3	F	V	V	V	F
I_4	F	F	V	F	V

Modelos: I_2.
Fbfs atómicas: P.

(b) $\{(P \wedge Q) \to R, P, \neg R\}$.

Interpretação	P	Q	R	$(P \land Q) \to R$	$\neg R$
I_1	V	V	V	V	F
I_2	V	V	F	F	V
I_3	V	F	V	V	F
I_4	V	F	F	V	V
I_5	F	V	V	V	F
I_6	F	V	F	V	V
I_7	F	F	V	V	F
I_8	F	F	F	V	V

Modelos: I_4.
Fbfs atómicas: P.

(c) $\{P \to R, Q \to R, P \lor Q\}$.

Interpretação	P	Q	R	$P \to R$	$Q \to R$	$P \lor Q$
I_1	V	V	V	V	V	V
I_2	V	V	F	F	F	V
I_3	V	F	V	V	V	V
I_4	V	F	F	F	V	V
I_5	F	V	V	V	V	V
I_6	F	V	F	V	F	V
I_7	F	F	V	V	V	F
I_8	F	F	F	V	V	F

Modelos: I_1, I_3, I_5.
Fbfs atómicas: R.

2.2.6. Prove que a *fbf* $((P \land Q) \to R) \to ((P \to R) \lor (Q \to R))$ é uma tautologia.

Resposta:

Teremos de provar que a fbf dada é verdadeira segundo todas as interpretações:

P	Q	R	$(P \land Q) \to R$	$P \to R$	$Q \to R$	$((P \land Q) \to R) \to$ $((P \to R) \lor (Q \to R))$
V	V	V	V	V	V	V
V	V	F	F	F	F	V
V	F	V	V	V	V	V
V	F	F	V	F	V	V
F	V	V	V	V	V	V
F	V	F	V	V	F	V
F	F	V	V	V	V	V
F	F	F	V	V	V	V

2.2.7. Prove que a *fbf* $(P \lor Q) \land (P \to R) \land (Q \to R) \land \neg R$ é uma contradição.

Resposta:

Teremos de provar que a fbf dada é falsa segundo todas as interpretações:

P	Q	R	$(P \vee Q)$	$P \rightarrow R$	$Q \rightarrow R$	$\neg R$	$(P \vee Q) \wedge (P \rightarrow R)$ $\wedge (Q \rightarrow R) \wedge \neg R$
V	V	V	V	V	V	F	F
V	V	F	V	F	F	V	F
V	F	V	V	V	V	F	F
V	F	F	V	F	V	V	F
F	V	V	V	V	V	F	F
F	V	F	V	V	F	V	F
F	F	V	F	V	V	F	F
F	F	F	F	V	V	V	F

2.2.8. Prove que $(P \vee Q) \rightarrow R$ é uma consequência semântica do conjunto $\Delta = \{P \rightarrow R, Q \rightarrow R\}$, preenchendo apenas as posições necessárias da tabela abaixo. Justifique a sua resposta.

Interpretação	P	Q	R	$P \rightarrow R$	$Q \rightarrow R$	$P \vee Q$	$(P \vee Q) \rightarrow R$
I_1	V	V	V	V	V		
I_2	V	V	F	F			
I_3	V	F	V	V	V		
I_4	V	F	F	F			
I_5	F	V	V	V	V		
I_6	F	V	F	V	F		
I_7	F	F	V	V	V		
I_8	F	F	F	V	V		

Resposta:

Para provar que $(P \vee Q) \rightarrow R$ é consequência semântica do conjunto Δ, basta provar que $(P \vee Q) \rightarrow R$ é verdadeira em todos os modelos de Δ, ou seja, nas interpretações I_1, I_3, I_5, I_7, I_8.

Interpretação	P	Q	R	$P \rightarrow R$	$Q \rightarrow R$	$P \vee Q$	$(P \vee Q) \rightarrow R$
I_1	V	V	V	V	V	V	V
I_2	V	V	F	F			
I_3	V	F	V	V	V	V	V
I_4	V	F	F	F			
I_5	F	V	V	V	V	V	V
I_6	F	V	F	V	F		
I_7	F	F	V	V	V	F	V
I_8	F	F	F	V	V	F	V

2.2.9. Prove a seguinte afirmação:

"Se Δ for um conjunto de *fbfs* não satisfazível,
então dada qualquer *fbf* α, tem-se que $\Delta \models \alpha$."

Resposta:

Se Δ for um conjunto de *fbfs* não satisfazível, então, por definição, não tem nenhum modelo. Logo, não existe nenhum modelo de Δ que não satisfaça α.

2.2.10. Sabendo que o conjunto de *fbfs*

$$\Delta = \{P \to Q, Q \to R, P, \neg R\}$$

não é satisfazível, e que qualquer seu subconjunto próprio é satisfazível, classifique as seguintes afirmações em verdadeiras ou falsas, sem construir tabelas de verdade. Justifique as suas respostas.

(a) $\{P \to Q, Q \to R, P\} \models R$.

(b) $\{P \to Q, P, \neg R\} \models \neg(Q \to R)$.

(c) $\{P \to Q, Q \to R, \neg R\} \models P$.

(d) $\{P \to Q, Q \to R, P, \neg R\} \models P \wedge \neg P$.

Resposta:

(a) $\{P \to Q, Q \to R, P\} \models R$.
Verdadeira, pelo teorema da refutação.

(b) $\{P \to Q, P, \neg R\} \models \neg(Q \to R)$.
Verdadeira, pelo teorema da refutação.

(c) $\{P \to Q, Q \to R, \neg R\} \models P$.
Falsa; $\{P \to Q, Q \to R, \neg R\}$ é satisfazível, pelo enunciado. Logo tem pelo menos um modelo; este modelo não satisfaz P, porque Δ não é satisfazível.

(d) $\{P \to Q, Q \to R, P, \neg R\} \models P \wedge \neg P$.
Verdadeira, pela afirmação provada no exercício 2.2.9.

2.2.11. Usando a semântica da Lógica Proposicional, classifique as *fbfs* abaixo em satisfazíveis, falsificáveis, tautologias ou contradições, preenchendo a tabela abaixo com "S"(sim) e "N"(não). Justifique as suas respostas.

fbf	satisf.	falsif.	tautol.	contrad.
$(P \vee Q) \to (P \vee R)$				
$(P \wedge \neg P) \to Q$				
$(P \to Q) \wedge (P \to \neg Q) \wedge P$				

Resposta:

fbf	satisf.	falsif.	tautol.	contrad.
$(P \vee Q) \rightarrow (P \vee R)$	S	S	N	N
$(P \wedge \neg P) \rightarrow Q$	S	N	S	N
$(P \rightarrow Q) \wedge (P \rightarrow \neg Q) \wedge P$	N	S	N	S

Justificação:

- $(P \vee Q) \rightarrow (P \vee R)$:
  Existe pelo menos uma interpretação que torna a *fbf* verdadeira, e pelo menos uma interpretação que torna a *fbf* falsa:
  I_1, tal que $I_1(P) = I_1(Q) = I_1(R) = V$, torna a *fbf* verdadeira.
  I_2, tal que $I_2(P) = I_2(R) = F,\ I_2(Q) = V$, torna a *fbf* falsa.
- $(P \wedge \neg P) \rightarrow Q$:
  Qualquer que seja a interpretação I, $I(P \wedge \neg P) = F$, logo $I((P \wedge \neg P) \rightarrow Q) = V$.
- $(P \rightarrow Q) \wedge (P \rightarrow \neg Q) \wedge P$:
  Suponhamos que existe uma interpretação I, tal que $I((P \rightarrow Q) \wedge (P \rightarrow \neg Q) \wedge P) = V$; então $I(P \rightarrow Q) = V$, $I(P \rightarrow \neg Q) = V$ e $I(P) = V$. Logo, $I(Q) = V$ e $I(\neg Q) = V$, o que é impossível. Podemos concluir então que não existe nenhuma interpretação que torne a *fbf* verdadeira.

2.2.12. Considere a seguinte tabela (com o mesmo significado que a tabela do exercício 2.2.11). Diga, justificando, quais as linhas que não é possível existirem.

fbf	satisf.	falsif.	tautol.	contrad.
α	S	S	N	N
β	S	N	S	N
γ	S	S	S	N
δ	N	S	N	S
ϵ	N	N	N	S

Resposta:

A linha 3 não pode existir porque, entre outras razões, não é possível uma *fbf* ser falsificável e ser uma tautologia.
A linha 5 não pode existir porque, entre outras razões, não é possível uma *fbf* não ser satisfazível e não ser falsificável.

2.2.13. Considere a seguinte tabela (com o mesmo significado que a tabela do exercício 2.2.11). Preencha todas as posições que for possível.

fbf	satisf.	falsif.	tautol.	contrad.
α	S	S		
β	S	N		
γ			S	
δ	N		N	
ϵ	S			
ϕ			N	
σ	N			

Resposta:

fbf	satisf.	falsif.	tautol.	contrad.
α	S	S	N	N
β	S	N	S	N
γ	S	N	S	N
δ	N	S	N	S
ϵ	S			
ϕ		S	N	
σ	N	S	N	S

2.2.14. Indique os passos a seguir para, dados um conjunto de *fbfs* Δ e uma *fbf* α, provar que $\Delta \nvdash \alpha$, justificando a sua resposta.

Resposta:

$\Delta \nvdash \alpha$ significa que não existe uma prova de α a partir de Δ. Como não é possível mostrar a não existência de uma prova, vamos usar o facto da Lógica Proposicional ser correta. Assim, sabemos que se $\Delta \vdash \alpha$, então $\Delta \models \alpha$. Consequentemente, se $\Delta \not\models \alpha$, então $\Delta \nvdash \alpha$. $\Delta \not\models \alpha$ significa que existe um modelo de Δ que não satisfaz α. Assim, para provar que $\Delta \nvdash \alpha$, basta encontrar uma interpretação que satisfaça todas as *fbfs* de Δ e não satisfaça α.

Capítulo 3

Lógica proposicional (II)

3.1 Forma clausal e resolução

3.1.1. Passe as seguintes *fbfs* para a forma clausal:

(a) $(P \to R) \wedge (Q \to R) \wedge \neg((P \vee Q) \to R)$

(b) $\neg((P \to R) \wedge (Q \to R))$

(c) $P \to (Q \wedge ((Q \vee S) \to R))$

Resposta:

(a) $(P \to R) \wedge (Q \to R) \wedge \neg((P \vee Q) \to R)$:

- *Eliminação do símbolo* $\to$

$$(\neg P \vee R) \wedge (\neg Q \vee R) \wedge \neg(\neg(P \vee Q) \vee R)$$

- *Redução do domínio do símbolo* $\neg$

$$(\neg P \vee R) \wedge (\neg Q \vee R) \wedge (\neg\neg(P \vee Q) \wedge \neg R)$$

$$(\neg P \vee R) \wedge (\neg Q \vee R) \wedge (P \vee Q) \wedge \neg R$$

- *Obtenção da forma conjuntiva normal*
  Não se aplica.
- *Eliminação do símbolo* $\wedge$

$$\{\neg P \vee R, \neg Q \vee R, P \vee Q, \neg R\}$$

- *Eliminação do símbolo* $\vee$

$$\{\{\neg P, R\}, \{\neg Q, R\}, \{P, Q\}, \{\neg R\}\}$$

(b) $\neg((P \to R) \wedge (Q \to R))$:

- *Eliminação do símbolo* $\to$

$$\neg((\neg P \vee R) \wedge (\neg Q \vee R))$$

- *Redução do domínio do símbolo* $\neg$

$$\neg(\neg P \vee R) \vee \neg(\neg Q \vee R))$$

$$(P \wedge \neg R) \vee (Q \wedge \neg R)$$

- *Obtenção da forma conjuntiva normal*

$$((P \wedge \neg R) \vee Q) \wedge ((P \wedge \neg R) \vee \neg R)$$

$$(P \vee Q) \wedge (\neg R \vee Q) \wedge (P \vee \neg R) \wedge (\neg R \vee \neg R)$$

- *Eliminação do símbolo* $\wedge$

$$\{(P \vee Q), (\neg R \vee Q), (P \vee \neg R), (\neg R \vee \neg R)\}$$

- *Eliminação do símbolo* $\vee$

$$\{\{P, Q\}, \{\neg R, Q\}, \{P, \neg R\}, \{\neg R\}\}$$

(c) $P \to (Q \wedge ((Q \vee S) \to R))$:

- *Eliminação do símbolo* $\to$

$$\neg P \vee (Q \wedge (\neg(Q \vee S) \vee R))$$

- *Redução do domínio do símbolo* $\neg$

$$\neg P \vee (Q \wedge ((\neg Q \wedge \neg S) \vee R))$$

- *Obtenção da forma conjuntiva normal*

$$(\neg P \vee Q) \wedge (\neg P \vee ((\neg Q \wedge \neg S) \vee R))$$

$$(\neg P \vee Q) \wedge (\neg P \vee ((\neg Q \vee R) \wedge (\neg S \vee R)))$$

$$(\neg P \vee Q) \wedge (\neg P \vee \neg Q \vee R) \wedge (\neg P \vee \neg S \vee R)$$

- *Eliminação do símbolo* $\wedge$

$$\{\neg P \vee Q, \neg P \vee \neg Q \vee R, \neg P \vee \neg S \vee R\}$$

- *Eliminação do símbolo* $\vee$

$$\{\{\neg P, Q\}, \{\neg P, \neg Q, R\}, \{\neg P, \neg S, R\}\}$$

3.1.2. Considere as afirmações do exercício 2.1.2:

(a) $(P \wedge Q) \rightarrow (P \vee R)$ é um teorema.

(b) $\{P \rightarrow Q, P \rightarrow R\} \vdash P \rightarrow (Q \wedge R)$.

(c) $\{P \vee Q, P \rightarrow R, Q \rightarrow R\} \vdash R$.

(d) $\{P \vee Q, P \rightarrow R, Q \rightarrow S\} \vdash R \vee S$.

(e) $\{P \rightarrow Q\} \vdash \neg(P \wedge \neg Q)$.

Prove estas afirmações usando resolução.

Resposta:

(a) Para demonstrar um teorema usando resolução, teremos de fazer uma prova por refutação a partir da sua negação. Neste caso, a partir de $\neg((P \wedge Q) \rightarrow (P \vee R))$.

- *Passagem à forma clausal:*
  $\neg((P \wedge Q) \rightarrow (P \vee R))$
  $\neg(\neg(P \wedge Q) \vee (P \vee R))$
  $(P \wedge Q) \wedge \neg(P \vee R))$
  $(P \wedge Q) \wedge (\neg P \wedge \neg R))$
  $\{\{P\}, \{Q\}, \{\neg P\}, \{\neg R\}\}$
- *Prova por refutação:*

1	$\{P\}$	Prem
2	$\{\neg P\}$	Prem
3	$\{\}$	Res, (1, 2)

(b) Prova de $\{P \rightarrow Q, P \rightarrow R\} \vdash P \rightarrow (Q \wedge R)$:

- *Passagem à forma clausal:*
  - *Premissas:*
    $\{P \rightarrow Q, P \rightarrow R\}$
    $\{\{\neg P, Q\}, \{\neg P, R\}\}$
  - *Conclusão:*
    $P \rightarrow (Q \wedge R)$
    $\neg P \vee (Q \wedge R)$
    $(\neg P \vee Q) \wedge (\neg P \vee R)$
    $\{\{\neg P, Q\}, \{\neg P, R\}\}$
- *Prova:*
  Prova trivial, pois a forma clausal das premissas e da conclusão é a mesma.

(c) Prova de $\{P \vee Q, P \rightarrow R, Q \rightarrow R\} \vdash R$:

- *Passagem à forma clausal:*

- *Premissas:*
  $\{P \vee Q, P \to R, Q \to R\}$
  $\{\{P, Q\}, \{\neg P, R\}, \{\neg Q, R\}\}$
- *Conclusão:*
  R
  $\{\{R\}\}$

• *Prova:*

1	$\{P, Q\}$	Prem
2	$\{\neg P, R\}$	Prem
3	$\{\neg Q, R\}$	Prem
4	$\{Q, R\}$	Res, (1, 2)
5	$\{R\}$	Res, (3, 4)

(d) Prova de $\{P \vee Q, P \to R, Q \to S\} \vdash R \vee S$:

• *Passagem à forma clausal:*

- *Premissas:*
  $\{P \vee Q, P \to R, Q \to S\}$
  $\{\{P, Q\}, \{\neg P, R\}, \{\neg Q, S\}\}$
- *Conclusão:*
  $R \vee S$
  $\{\{R, S\}\}$

• *Prova:*

1	$\{P, Q\}$	Prem
2	$\{\neg P, R\}$	Prem
3	$\{\neg Q, S\}$	Prem
4	$\{Q, R\}$	Res, (1, 2)
5	$\{R, S\}$	Res, (3, 4)

(e) Prova de $\{P \to Q\} \vdash \neg(P \wedge \neg Q)$:

• *Passagem à forma clausal:*

- *Premissas:*
  $\{P \to Q\}$
  $\{\{\neg P, Q\}\}$
- *Conclusão:*
  $\neg(P \wedge \neg Q)$
  $\neg P \vee Q)$
  $\{\{\neg P, Q\}\}$

• *Prova:*
Prova trivial, pois a forma clausal da premissa e da conclusão é a mesma.

3.1.3. Usando resolução prove as seguintes afirmações. Use provas por refutação *apenas* quando não for possível fazer a prova de outra forma.

(a) $\{\neg P\} \vdash P \to Q$.

(b) $\{P, \neg Q\} \vdash \neg(P \wedge Q)$.

(c) $\{P \to R, Q \to R, \neg P \to Q\} \vdash R$.

(d) $\{P, P \to Q, P \to R\} \vdash Q \wedge R$.

(e) $\{\neg P \to Q, P \to \neg Q, P \to R, R \to Q\} \vdash \neg P$.

(f) $\{P \to Q, P \to \neg Q, P\} \vdash R$.

(g) $\{P \to (R \wedge S)\} \vdash P \to (R \vee S)$.

Resposta:

(a) $\{\neg P\} \vdash P \to Q$
Uma vez que as premissas não contêm o símbolo Q, só será possível fazer uma prova por refutação. Para tal, vamos adicionar às premissas a negação da conclusão,

$$\{\neg P, \neg(P \to Q)\}$$

- *Passagem à forma clausal:*
  - *Premissa:*
    $\{\{\neg P\}\}$
  - *Negação da conclusão:*
    $\neg(P \to Q)$
    $\neg(\neg P \vee Q)$
    $P \wedge \neg Q)$
    $\{\{P\}, \{\neg Q\}\}$
- *Prova por refutação:*

1	$\{\neg P\}$	Prem
2	$\{P\}$	Prem
3	$\{\}$	Res, (1, 2)

(b) $\{P, \neg Q\} \vdash \neg(P \wedge Q)$

- *Passagem à forma clausal:*
  - *Premissas:*
    $\{\{P\}, \{\neg Q\}\}$
    Uma vez que não é possível aplicar a resolução às premissas, só será possível fazer uma prova por refutação.

  - *Negação da conclusão:*
    $\neg(\neg(P \wedge Q))$
    $P \wedge Q$
    $P \wedge \neg Q)$
    $\{\{P\}, \{Q\}\}$

- *Prova por refutação:*

1	$\{\neg Q\}$	Prem
2	$\{Q\}$	Prem
3	$\{\}$	Res, (1, 2)

(c) $\{P \to R, Q \to R, \neg P \to Q\} \vdash R$

- *Passagem à forma clausal:*
  $\{\{\neg P, R\}, \{\neg Q, R\}, \{P, Q\}\} \vdash \{\{R\}\}$
- *Prova:*

1	$\{\neg P, R\}$	Prem
2	$\{\neg Q, R\}$	Prem
3	$\{P, Q\}$	Prem
4	$\{Q, R\}$	Res, (1, 3)
5	$\{R\}$	Res, (2, 4)

(d) $\{P, P \to Q, P \to R\} \vdash Q \wedge R$

- *Passagem à forma clausal:*
  $\{\{P\} \{\neg P, Q\}, \{\neg P, R\}\} \vdash \{\{Q\}, \{R\}\}$
- *Prova:*
  Uma vez que a conclusão é constituída por duas cláusulas, teremos de provar cada uma delas:

1	$\{P\}$	Prem
2	$\{\neg P, Q\}$	Prem
3	$\{\neg P, R\}$	Prem
4	$\{Q\}$	Res, (1, 2)
5	$\{R\}$	Res, (1, 3)

(e) $\{\neg P \to Q, P \to \neg Q, P \to R, R \to Q\} \vdash \neg P$

- *Passagem à forma clausal:*
  $\{\{P, Q\}, \{\neg P, \neg Q\}, \{\neg P, R\}, \{\neg R, Q\}\} \vdash \{\{\neg P\}\}$
- *Prova:*

1	$\{P, Q\}$	Prem
2	$\{\neg P, \neg Q\}$	Prem
3	$\{\neg P, R\}$	Prem
4	$\{\neg R, Q\}$	Prem
5	$\{Q, R\}$	Res, (1, 3)
6	$\{Q\}$	Res, (4, 5)
7	$\{\neg P\}$	Res, (2, 6)

(f) $\{P \to Q, P \to \neg Q, P\} \vdash R$
Uma vez que as premissas não contêm o símbolo R, só será possível fazer uma prova por refutação. Para tal, vamos adicionar às premissas a negação da conclusão,

$$\{P \to Q, P \to \neg Q, P, \neg R\}$$

e tentar obter a cláusula vazia a partir do conjunto de cláusulas correspondente.

- *Passagem à forma clausal:*
$\{\{\neg P, Q\}, \{\neg P, \neg Q\}, \{P\}, \{\neg R\}\}$
- *Prova por refutação:*

1	$\{\neg P, Q\}$	Prem
2	$\{\neg P, \neg Q\}$	Prem
3	$\{P\}$	Prem
4	$\{\neg R\}$	Prem
5	$\{\neg P\}$	Res, (1, 2)
6	$\{\}$	Res, (3, 5)

(g) Faremos uma prova por refutação:

- *Passagem à forma clausal:*
  - *Premissa:*
  $P \to (R \wedge S)$
  $\neg P \vee (R \wedge S)$
  $(\neg P \vee R) \wedge (\neg P \vee S)$
  $\{\{\neg P, R\}, \{\neg P, S\}\}$
  - *Negação da conclusão:*
  $\neg(P \to (R \vee S))$
  $\neg(\neg P \vee R \vee S)$
  $(P \wedge \neg R \wedge \neg S))$
  $\{\{P\}, \{\neg R\}, \{\neg S\}\}$
- *Prova por refutação:*

1	$\{\neg P, R\}$	Prem
2	$\{P\}$	Prem
3	$\{\neg R\}$	Prem
4	$\{R\}$	Res, (1, 2)
5	$\{\}$	Res, (3, 4)

3.1.4. Usando resolução e uma prova por refutação, prove que

$$\{P \wedge (Q \vee R)\} \vdash (P \wedge Q) \vee (P \wedge R)$$

Resposta:

- *Passagem à forma clausal:*
  - *Premissas:*
    $\{\{P\},\{Q,R\}\}$
  - *Negação da conclusão:*
    $\neg((P \wedge Q) \vee (P \wedge R))$
    $\neg(P \wedge Q) \wedge \neg(P \wedge R)$
    $(\neg P \vee \neg Q) \wedge (\neg P \vee \neg R)$
    $\{\{\neg P, \neg Q\}, \{\neg P, \neg R\}\}$
- *Prova por refutação:*

1	$\{P\}$	Prem
2	$\{Q,R\}$	Prem
3	$\{\neg P, \neg Q\}$	Prem
4	$\{\neg P, \neg R\}\}$	Prem
5	$\{\neg Q\}$	Res, (1, 3)
6	$\{\neg R\}$	Res, (1, 4)
7	$\{R\}$	Res, (2, 5)
8	$\{\}$	Res, (6, 7)

3.2 Estratégias em resolução

3.2.1. Demonstre os seguintes argumentos, usando resolução por saturação de níveis.

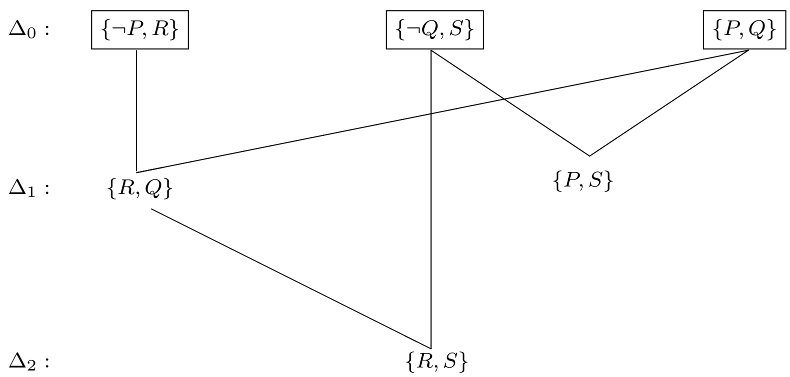
(a) $(\{P \rightarrow R, Q \rightarrow S, P \vee Q\}, R \vee S)$

(b) $(\{P \rightarrow Q, Q \rightarrow R, P \vee Q\}, R)$

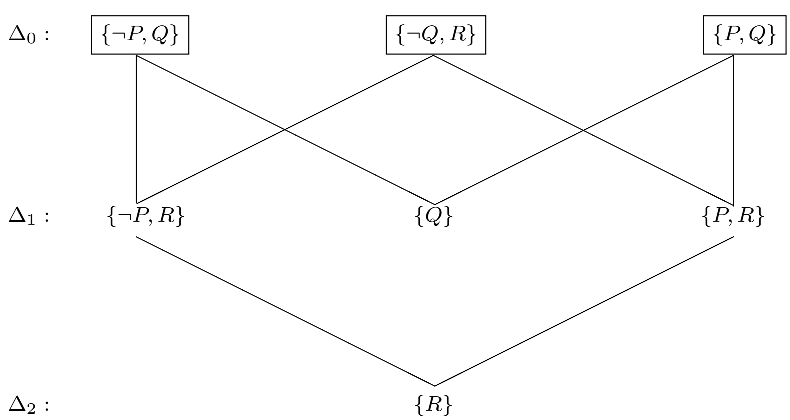
(c) $\{P \rightarrow (Q \rightarrow R), S \rightarrow (Q \rightarrow R), P \vee S\} \vdash Q \rightarrow R$

Resposta:

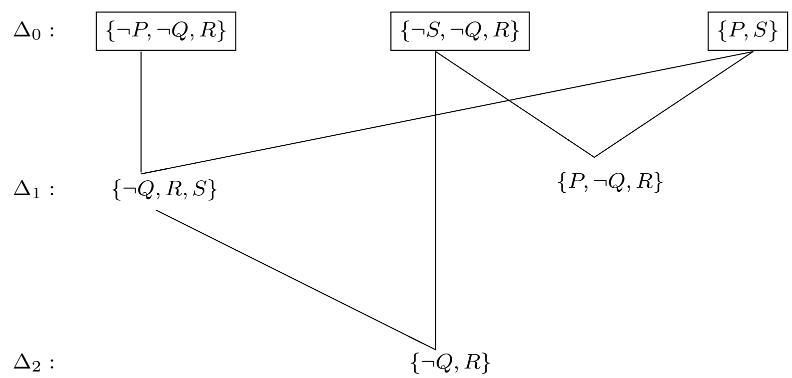
(a)



(b)



(c)
- *Passagem à forma clausal:*
  - *Premissas:*
    $\{\{\neg P, \neg Q, R\}, \{\neg S, \neg Q, R\}, \{P, S\}\}$
  - *Conclusão:*
    $\{\{\neg Q, R\}\}$
- *Prova por saturação de níveis:*



3.2.2. Considere a demonstração do argumento

$$(\{\{P, Q\}, \{\neg P, \neg Q, S\}, \{Q, R\}, \{\neg P, Q\}, \{P, S, R\}, \{\neg Q\}, \{P, Q, R\}\}, T)$$

usando resolução e uma prova por refutação. Após a adição da negação da conclusão, o conjunto de premissas seria:

$$\{\{P, Q\}, \{\neg P, \neg Q, S\}, \{Q, R\}, \{\neg P, Q\}, \{P, S, R\}, \{\neg Q\}, \{P, Q, R\}, \{\neg T\}\}$$

Depois de aplicadas as estratégias de eliminação de cláusulas qual seria o conjunto de premissas? Justifique a sua resposta.

Resposta:

Depois de aplicadas as estratégias de eliminação de cláusulas o conjunto de premissas passaria a ser:

$$\{\{P, Q\}, \{\neg P, Q\}, \{\neg Q\}\}$$

A cláusula $\{\neg P, \neg Q, S\}$ foi eliminada por ser ser subordinada pela cláusula $\{\neg Q\}$. A cláusula $\{P, Q, R\}$ foi eliminada por ser ser subordinada pela cláusula $\{P, Q\}$. As restantes cláusulas foram eliminadas por conterem um literal puro (S, R ou $\neg T$).

3.2.3. Considere a seguinte prova por refutação:

1	$\{P, Q\}$	Prem
2	$\{\neg P, \neg Q, S\}$	Prem
3	$\{Q, R\}$	Prem
4	$\{\neg P, Q\}$	Prem
5	$\{P, S, R\}$	Prem
6	$\{\neg Q\}$	Prem
7	$\{Q, \neg Q, S\}$	Res, (1, 2)
8	$\{Q, S, R\}$	Res, (4, 5)
9	$\{P\}$	Res, (1, 6)
10	$\{\neg P\}$	Res, (4, 6)
11	$\{\}$	Res, (9, 10)

Indique as linhas da prova que seriam eliminadas pelas estratégias de eliminação de cláusulas, justificando a sua resposta.

Resposta:

Seriam eliminadas as linhas 7 e 8:

A linha 7 seria eliminada porque a cláusula $\{Q, \neg Q, S\}$ é um teorema.

A linha 8 seria eliminada porque a cláusula $\{Q, S, R\}$ é subordinada pela cláusula $\{Q, R\}$ (linha 3).

3.2.4. Para cada uma das provas abaixo, diga se foi usada a estratégia de resolução linear. Em caso afirmativo, diga qual a cláusula inicial e quais as cláusulas centrais. Em caso negativo, justifique a sua resposta.

(a)

1	$\{P, Q\}$	Prem
2	$\{\neg P, \neg Q\}$	Prem
3	$\{\neg P, R\}$	Prem
4	$\{\neg R, Q\}$	Prem
5	$\{Q, R\}$	Res, (1, 3)
6	$\{Q\}$	Res, (4, 5)
7	$\{\neg P\}$	Res, (2, 6)

(b)

1	$\{Q, S\}$	Prem
2	$\{\neg Q, P\}$	Prem
3	$\{Q, R\}$	Prem
4	$\{\neg P, \neg Q\}$	Prem
5	$\{P, S\}$	Res, (1, 2)
6	$\{\neg P, R\}$	Res, (3, 4)
7	$\{S, R\}$	Res, (5, 6)

Resposta:

(a) Foi usada resolução linear. A cláusula inicial é $\{P, Q\}$ ou $\{\neg P, R\}$. As cláusulas centrais são $\{Q, R\}$, $\{Q\}$ e $\{\neg P\}$.

(b) Não foi usada resolução linear, pois na linha 6 não foi usada a cláusula da linha 5.

3.2.5. Para cada um dos seguintes argumentos, apresente uma prova por refutação, usando resolução linear e, quando possível, resolução unitária.

(a) $(\{\neg Q \to S, Q \to P, \neg(P \wedge Q)\}, S)$.

(b) $(\{P \to S, \neg(\neg P \wedge \neg S), \neg(Q \wedge \neg R), Q \vee R\}, S \wedge R)$.

Resposta:

(a) *Premissas e negação da conclusão, na forma clausal:*

$$\{\{Q, S\}, \{\neg Q, P\}, \{\neg P, \neg Q\}, \{\neg S\}\}$$

Prova, usando resolução linear:

1	$\{Q, S\}$	Prem
2	$\{\neg Q, P\}$	Prem
3	$\{\neg P, \neg Q\}$	Prem
4	$\{\neg S\}$	Prem
5	$\{Q\}$	Res, (1, 4)
6	$\{P\}$	Res, (2, 5)
7	$\{\neg Q\}$	Res, (3, 6)
8	$\{\}$	Res, (5, 7)

(b) *Premissas e negação da conclusão, na forma clausal:*

$$\{\{\neg P, S\}, \{P, S\}, \{\neg Q, R\}, \{Q, R\}, \{\neg S, \neg R\}\}$$

Não é possível usar resolução unitária pois o conjunto não contém nenhuma clásula unitária.
Prova, usando resolução linear:

1	$\{\neg P, S\}$	Prem
2	$\{P, S\}$	Prem
3	$\{\neg Q, R\}$	Prem
4	$\{Q, R\}$	Prem
5	$\{\neg S, \neg R\}$	Prem
6	$\{S\}$	Res, (1, 2)
7	$\{\neg R\}$	Res, (5, 6)
8	$\{\neg Q\}$	Res, (3, 7)
9	$\{R\}$	Res, (4, 8)
10	$\{\}$	Res, (7, 9)

3.2.6. Usando resolução unitária e uma prova por refutação, prove que:

(a) $((P \to Q) \wedge (P \to \neg Q)) \to \neg P$ é um teorema.

(b) $\{(P \vee Q) \to R\} \vdash (P \to R) \wedge (Q \to R)$.

(c) $\{(P \to R) \wedge (Q \to R)\} \vdash (P \vee Q) \to R$.

(d) $\{(P \wedge Q) \vee (P \wedge R)\} \vdash P \wedge (Q \vee R)$.

(e) $\{\neg(P \wedge Q)\} \vdash \neg P \vee \neg Q$.

(f) $\{(P \vee Q) \wedge (P \vee R)\} \vdash P \vee (Q \wedge R)$.

Resposta:

(a) • *Passagem à forma clausal:*

- *Premissas:*
  $\{\}$
- *Negação da conclusão:*
  $\neg(((P \to Q) \wedge (P \to \neg Q)) \to \neg P)$
  $\neg(((\neg P \vee Q) \wedge (\neg P \vee \neg Q)) \to \neg P)$
  $\neg(\neg((\neg P \vee Q) \wedge (\neg P \vee \neg Q)) \vee \neg P)$
  $((\neg P \vee Q) \wedge (\neg P \vee \neg Q)) \wedge P$
  $\{\{\neg P, Q\}, \{\neg P, \neg Q\}, \{P\}\}$

• *Prova por refutação:*

1	$\{\neg P, Q\}$	Prem
2	$\{\neg P, \neg Q\}$	Prem
3	$\{P\}$	Prem
4	$\{Q\}$	Res, (1, 3)
5	$\{\neg Q\}$	Res, (2, 3)
6	$\{\}$	Res, (4, 5)

(b)

- *Passagem à forma clausal:*
  - *Premissa:*
    $(P \vee Q) \to R$
    $\neg(P \vee Q) \vee R$
    $(\neg P \wedge \neg Q) \vee R$
    $(\neg P \vee R) \wedge (\neg Q \vee R)$
    $\{\{\neg P, R\}, \{\neg Q, R\}\}$
  - *Negação da conclusão:*
    $\neg((P \to R) \wedge (Q \to R))$
    $\neg(\neg P \vee R) \vee \neg(\neg Q \vee R))$
    $(P \wedge \neg R) \vee (Q \wedge \neg R))$
    $((P \wedge \neg R) \vee Q) \wedge ((P \wedge \neg R) \vee \neg R)))$
    $(P \vee Q) \wedge (\neg R \vee Q) \wedge (P \vee \neg R) \wedge (\neg R \vee \neg R)$
    $\{\{P, Q\}, \{\neg R, Q\}, \{P, \neg R\}, \{\neg R\}\}$
- *Prova por refutação:*

1	$\{\neg P, R\}$	Prem
2	$\{\neg Q, R\}$	Prem
3	$\{P, Q\}$	Prem
4	$\{\neg R\}$	Prem
5	$\{\neg P\}$	Res, (1, 4)
6	$\{\neg Q\}$	Res, (2, 4)
7	$\{Q\}$	Res, (3, 5)
8	$\{\}$	Res, (6, 7)

(c)

- *Passagem à forma clausal:*
  - *Premissa:*
    $(P \to R) \wedge (Q \to R)$
    $(\neg P \vee R) \wedge (\neg Q \vee R)$
    $\{\{\neg P, R\}, \{\neg Q, R\}\}$
  - *Negação da conclusão:*
    $\neg((P \vee Q) \to R)$
    $\neg(\neg(P \vee Q) \vee R)$
    $(P \vee Q) \wedge \neg R$
    $\{\{P, Q\}, \{\neg R\}\}$
- *Prova por refutação:*

1	$\{\neg P, R\}$	Prem
2	$\{\neg Q, R\}$	Prem
3	$\{P, Q\}$	Prem
4	$\{\neg R\}$	Prem
5	$\{\neg P\}$	Res, (1, 4)
6	$\{\neg Q\}$	Res, (2, 4)
7	$\{Q\}$	Res, (3, 5)
8	$\{\}$	Res, (6, 7)

(d) • *Passagem à forma clausal:*

- *Premissa:*
  $(P \wedge Q) \vee (P \wedge R)$
  $((P \wedge Q) \vee P) \wedge ((P \wedge Q) \vee R)$
  $(P \vee P) \wedge (Q \vee P) \wedge (P \vee R) \wedge (Q \vee R)$
  $\{\{P\}, \{Q, P\}, \{P, R\}, \{Q, R\}\}$
- *Negação da conclusão:*
  $\neg(P \wedge (Q \vee R))$
  $\neg P \vee \neg(Q \vee R)$
  $\neg P \vee (\neg Q \wedge \neg R)$
  $(\neg P \vee \neg Q) \wedge (\neg P \vee \neg R)$
  $\{\{\neg P, \neg Q\}, \{\neg P, \neg R\}\}$

• *Prova por refutação:*

1	$\{P\}$	Prem
2	$\{Q, R\}$	Prem
3	$\{\neg P, \neg Q\}$	Prem
4	$\{\neg P, \neg R\}$	Prem
5	$\{\neg Q\}$	Res, (1, 3)
6	$\{\neg R\}$	Res, (1, 4)
7	$\{R\}$	Res, (2, 5)
8	$\{\}$	Res, (6, 7)

(e) • *Passagem à forma clausal:*

- *Premissa:*
  $\neg(P \wedge Q)$
  $\neg P \vee \neg Q$
  $\{\{\neg P, \neg Q\}\}$
- *Negação da conclusão:*
  $\neg(\neg P \vee \neg Q)$
  $P \wedge Q$
  $\{\{P\}, \{Q\}\}$

- *Prova por refutação:*

1	$\{\neg P, \neg Q\}$	Prem
2	$\{P\}$	Prem
3	$\{Q\}$	Prem
4	$\{\neg Q\}$	Res, (1, 2)
5	$\{\}$	Res, (3, 4)

(f)
- *Passagem à forma clausal:*
  - *Premissa:*
    $\{\{P, Q\}, \{P, R\}\}$
  - *Negação da conclusão:*
    $\neg(P \vee (Q \wedge R))$
    $\neg P \wedge \neg(Q \wedge R))$
    $\neg P \wedge (\neg Q \vee \neg R)$
    $\{\{\neg P\}, \{\neg Q, \neg R\}\}$
- *Prova por refutação:*

1	$\{P, Q\}$	Prem
2	$\{P, R\}$	Prem
3	$\{\neg P\}$	Prem
4	$\{\neg Q, \neg R\}$	Prem
5	$\{Q\}$	Res, (1, 3)
6	$\{R\}$	Res, (2, 3)
7	$\{\neg R\}$	Res, (4, 5)
8	$\{\}$	Res, (6, 7)

3.3 BDDs e OBDDs

3.3.1. Classifique as seguintes afirmações em verdadeiras (**V**), ou falsas (**F**):

(a) Num OBDD todas as folhas estão à profundidade máxima.

(b) Num OBDD todos os nós à profundidade máxima são folhas.

(c) Num OBDD todos os nós à mesma profundidade têm o mesmo rótulo.

(d) Numa árvore de decisão todos os nós à mesma profundidade têm o mesmo rótulo.

Resposta:

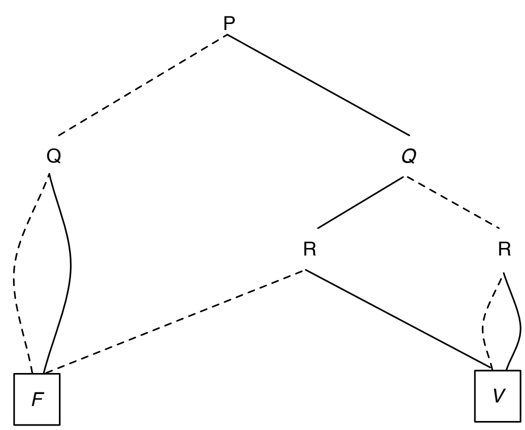
(a) Num OBDD todas as folhas estão à profundidade máxima.
Resp: **F**

(b) Num OBDD todos os nós à profundidade máxima são folhas.
Resp: **V**

(c) Num OBDD todos os nós à mesma profundidade têm o mesmo rótulo.
Resp: **F**

(d) Numa árvore de decisão todos os nós à mesma profundidade têm o mesmo rótulo.
Resp: **V**

3.3.2. Considere o seguinte BDD:



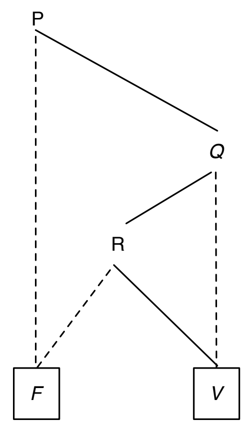
(a) Obtenha o BDD reduzido correspondente, por aplicação das transformações aplicáveis em BDDs:

- R1 *Remoção de folhas duplicadas.*
- R2 *Remoção de testes redundantes.*
- R3 *Remoção de nós redundantes.*

(b) Quais as interpretações que satisfazem a *fbf* representada pelo BDD?

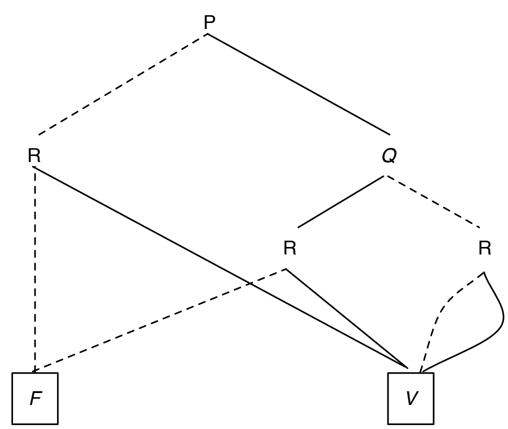
Resposta:

(a) Aplicação de R2:



(b) $I_1(P) = V$, $I_1(Q) = V$, $I_1(R) = V$.
$I_2(P) = V$, $I_2(Q) = F$, $I_2(R) = V$.
$I_3(P) = V$, $I_3(Q) = F$, $I_3(R) = F$.

3.3.3. Considere o seguinte BDD:

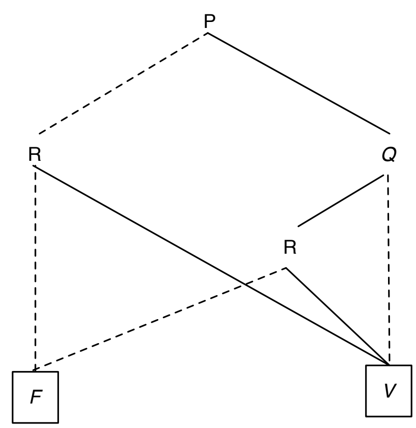


(a) Obtenha o BDD reduzido correspondente, por aplicação das transformações aplicáveis em BDDs:
- R1 *Remoção de folhas duplicadas.*
- R2 *Remoção de testes redundantes.*
- R3 *Remoção de nós redundantes.*

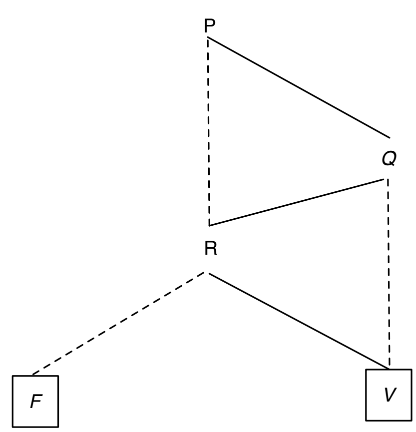
(b) Quais as interpretações que satisfazem a *fbf* representada pelo BDD?

Resposta:

(a) Aplicação de R2:

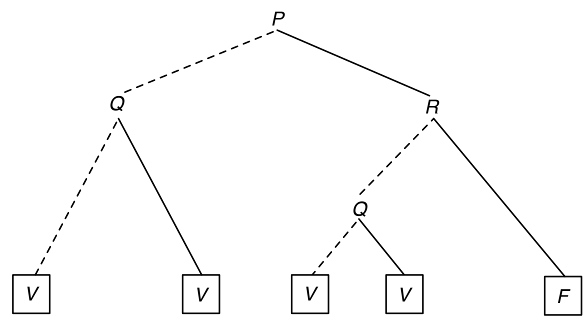


Aplicação de R3:



(b) $I_1(P) = V$, $I_1(Q) = V$, $I_1(R) = V$.
$I_2(P) = V$, $I_2(Q) = F$, $I_2(R) = V$.
$I_3(P) = V$, $I_3(Q) = F$, $I_3(R) = F$.
$I_4(P) = F$, $I_4(Q) = V$, $I_4(R) = V$.
$I_5(P) = F$, $I_5(Q) = F$, $I_5(R) = V$.

3.3.4. Considere o seguinte BDD:

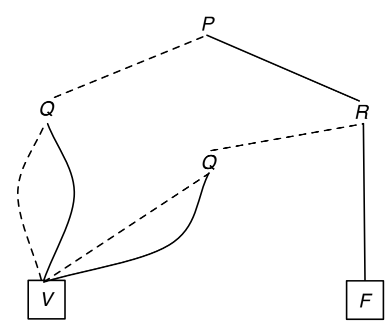


(a) Obtenha o BDD reduzido correspondente, por aplicação das transformações aplicáveis em BDDs:
- R1 *Remoção de folhas duplicadas.*
- R2 *Remoção de testes redundantes.*
- R3 *Remoção de nós redundantes.*

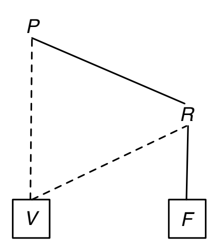
(b) Quais as interpretações que satisfazem a *fbf* representada pelo BDD?

Resposta:

(a) Aplicação de R1:

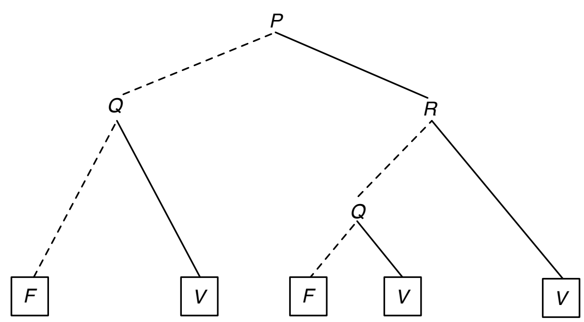


Aplicação de R2:



(b) Todas as interpretações que não satisfazem P ou não satisfazem R.

3.3.5. Considere o seguinte BDD:

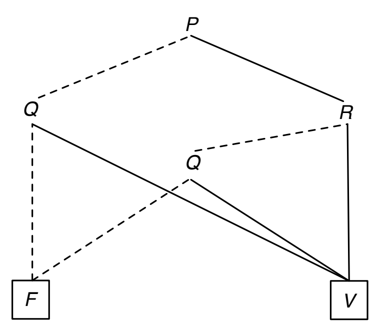


Obtenha o BDD reduzido correspondente, por aplicação das transformações aplicáveis em BDDs:

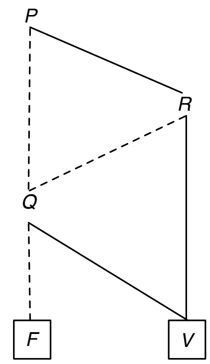
- R1 *Remoção de folhas duplicadas.*
- R2 *Remoção de testes redundantes.*
- R3 *Remoção de nós redundantes.*

Resposta:

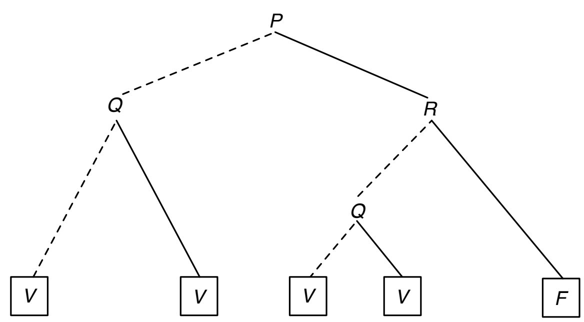
Aplicação de R1:



Aplicação de R3:



3.3.6. Aplique os algoritmos *reduz* e *compacta* ao seguinte OBDD:



Resposta:

Atribuição de identificadores:

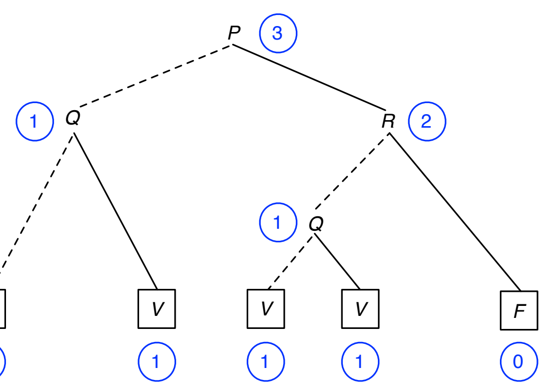
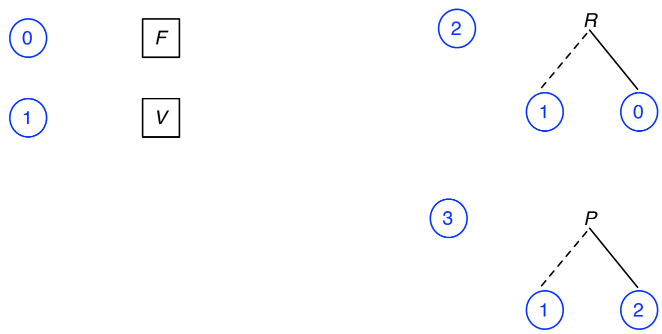
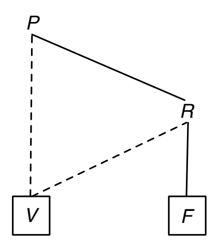


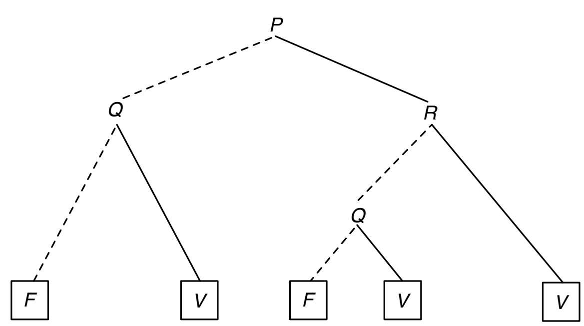
Tabela associativa:



Compactação:



3.3.7. Aplique os algoritmos *reduz* e *compacta* ao seguinte OBDD:



Resposta:

Atribuição de identificadores:

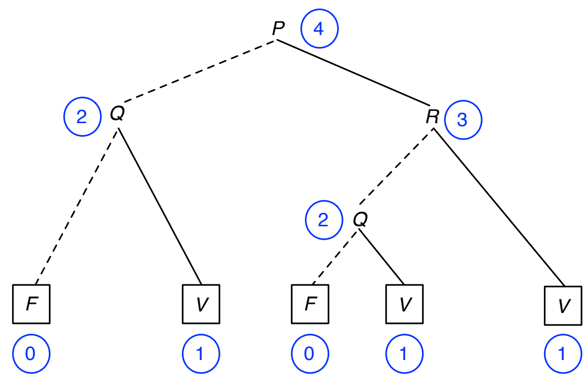
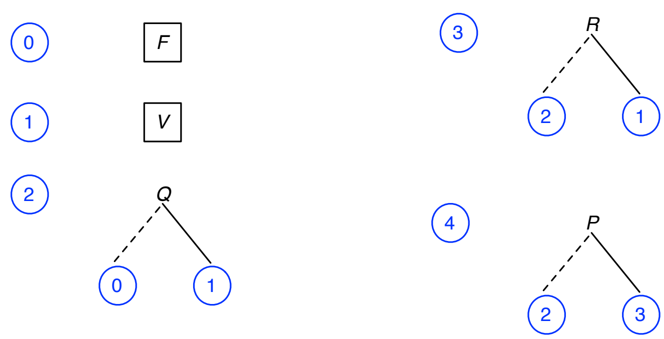
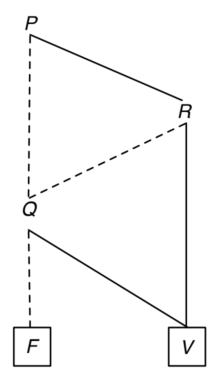


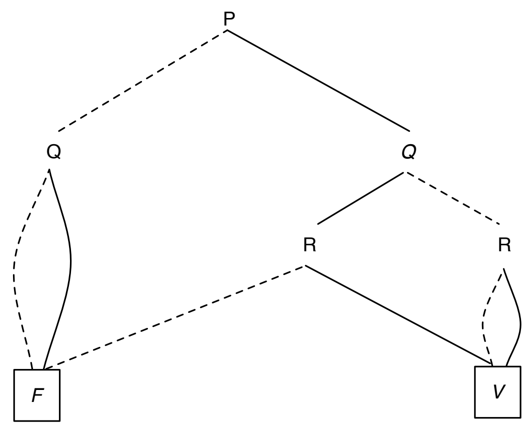
Tabela associativa:



Compactação:



3.3.8. Aplique os algoritmos *reduz* e *compacta* ao seguinte OBDD:



Resposta:

Atribuição de identificadores:

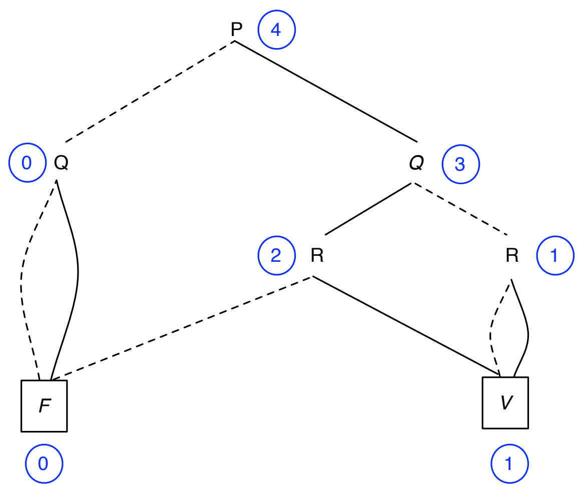
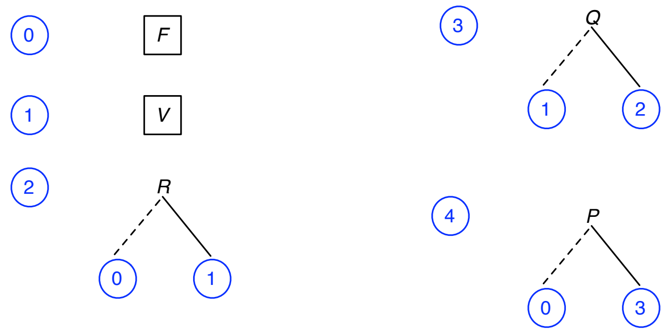
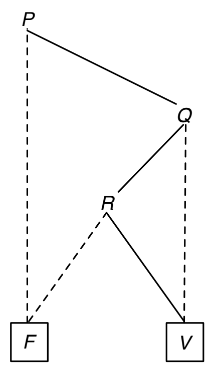


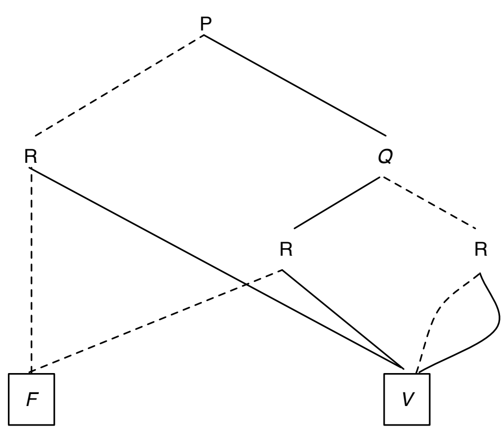
Tabela associativa:



Compactação:



3.3.9. Aplique os algoritmos *reduz* e *compacta* ao seguinte OBDD:



Resposta:

Atribuição de identificadores:

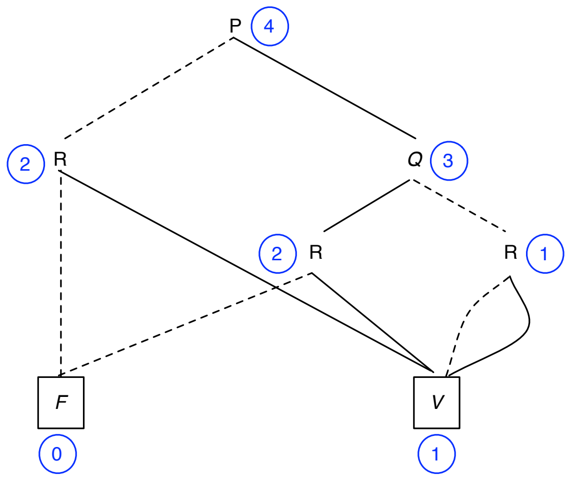
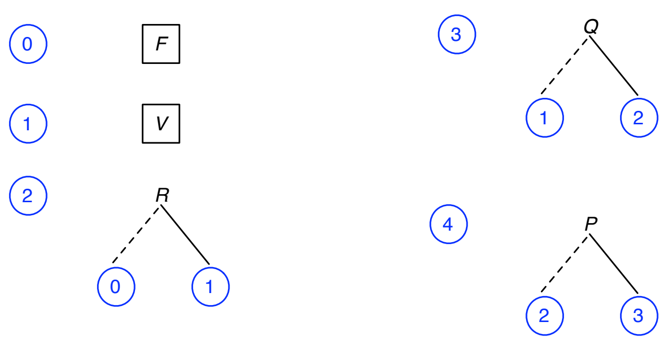
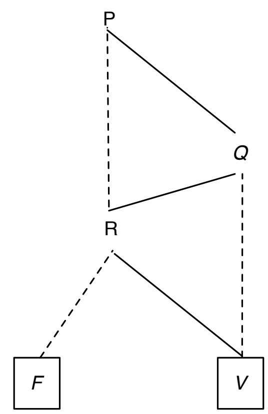


Tabela associativa:



Compactação:



3.3.10. Considere as *fbfs* $(P \vee Q) \to R$ e P e a ordem $P \prec Q \prec R$.

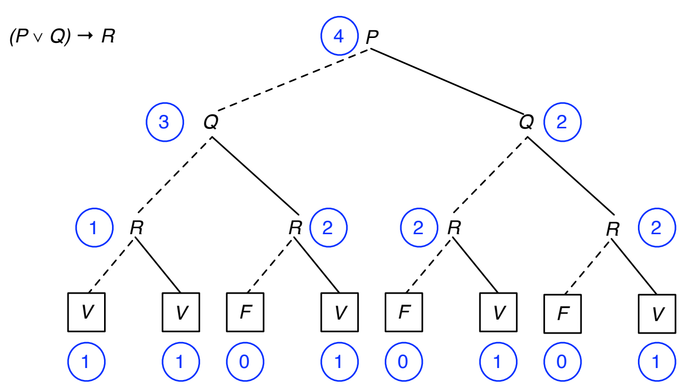
(a) Obtenha os seus OBDDs reduzidos, por aplicação dos algoritmos *reduz* e *compacta* às respetivas árvores binárias de decisão.

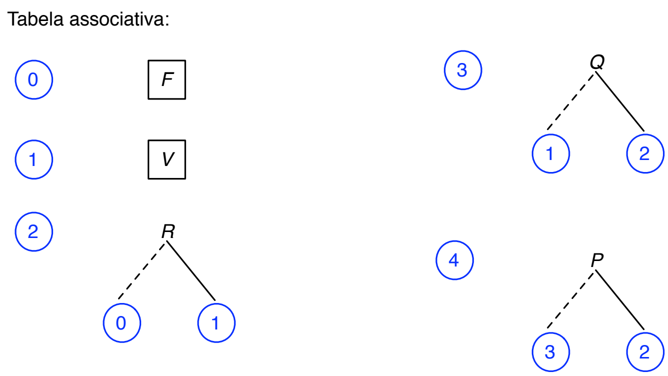
(b) Usando o algoritmo *aplica*, obtenha o OBDD reduzido da *fbf* $((P \vee Q) \to R) \wedge P$.

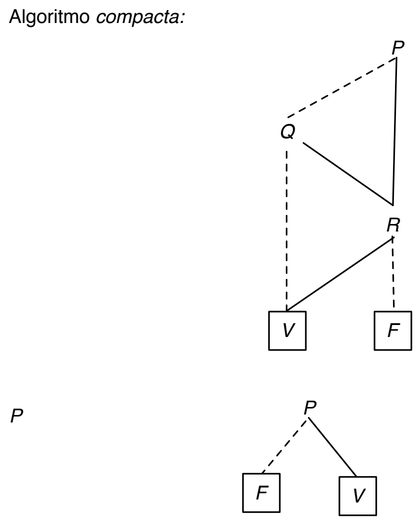
(c) O resultado obtido na alínea anterior permite concluir que $\{((P \vee Q) \to R) \wedge P\} \models R$? Justifique a sua resposta.

Resposta:

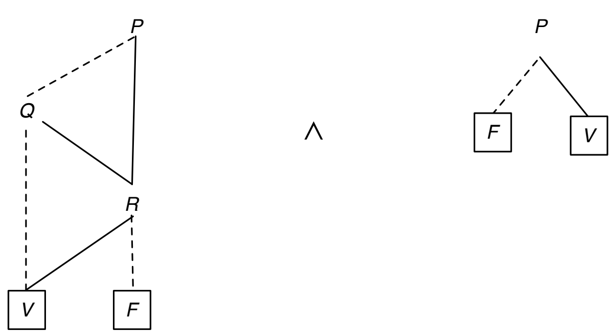
(a) OBDDs reduzidos de $(P \vee Q) \to R$ e P:

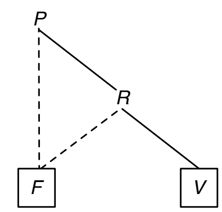






(b) OBDD reduzido de $((P \vee Q) \to R) \wedge P$:





(c) O resultado obtido na alínea anterior permite concluir que $\{((P \vee Q) \to R) \wedge P\} \models R$. Com efeito, o OBDD anterior permite concluir que a *fbf* $((P \vee Q) \to R) \wedge P$ tem 2 modelos:

$M_1(P) = V$, $M_1(Q) = V$ e $M_1(R) = V$ e
$M_2(P) = V$, $M_2(Q) = F$ e $M_2(R) = V$.

Em ambos os modelos R é verdadeira.

3.3.11. Considere a ordem $P \prec Q \prec R$.

(a) Obtenha o OBDD reduzido da *fbf* $P \to R$, por aplicação dos algoritmos *reduz* e *compacta* à árvore binária de decisão desta *fbf* .

(b) Obtenha o OBDD reduzido da *fbf* $Q \to R$, a partir do resultado da alínea anterior.

(c) Usando o algoritmo *aplica*, obtenha o OBDD reduzido da *fbf* $(P \to R) \wedge (Q \to R)$.

(d) Compare o OBDD que obteve na alínea anterior com o OBDD da *fbf* $(P \vee Q) \to R$ obtido na alínea (a) do exercício anterior. O que pode concluir?

(e) Usando o algoritmo *aplica*, obtenha agora o OBDD reduzido da *fbf* $(P \to R) \vee (Q \to R)$.

(f) O resultado obtido na alínea anterior permite concluir que $\{(P \to R) \vee (Q \to R)\} \models (P \wedge Q) \to R$?

Resposta:

(a)

$P \to R$

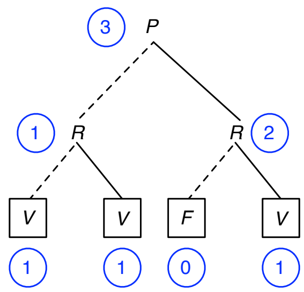
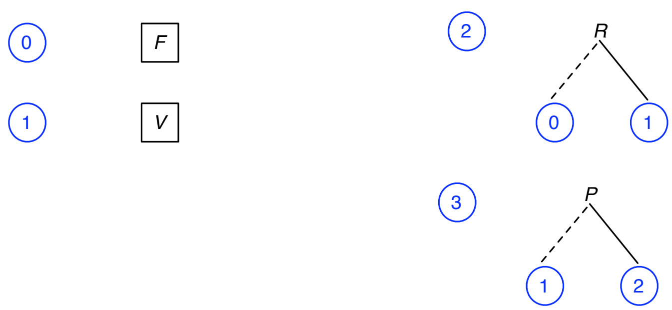
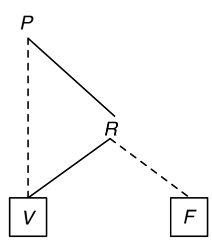


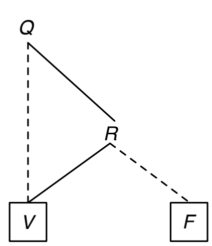
Tabela associativa:



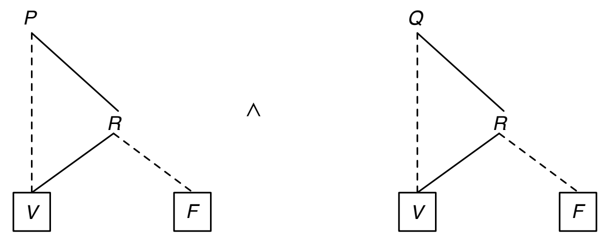
Algoritmo *compacta:*

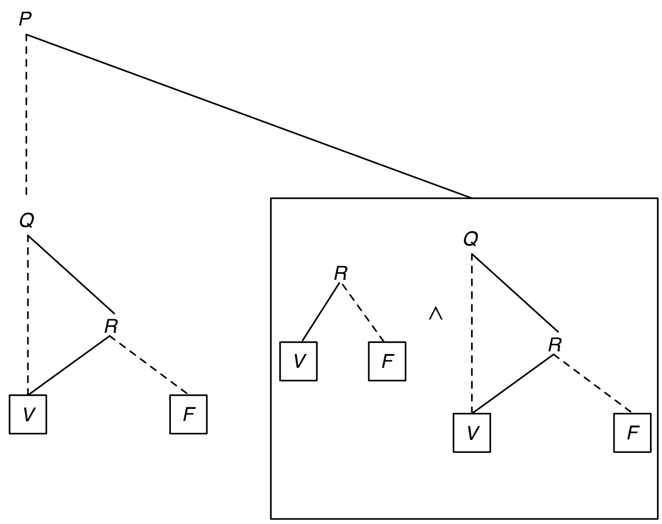


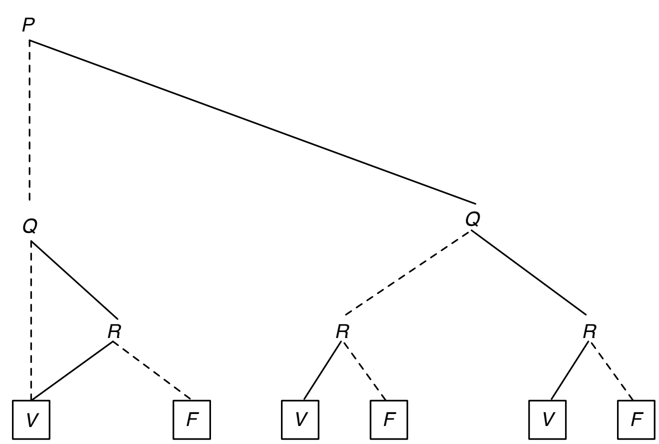
(b)

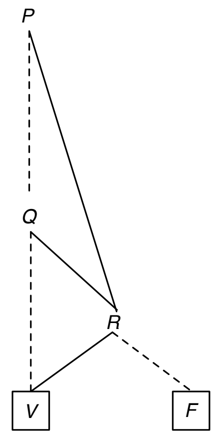


(c)



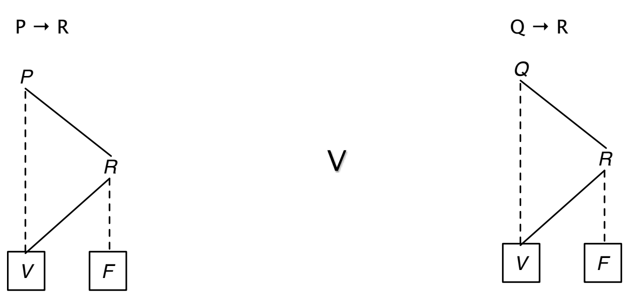


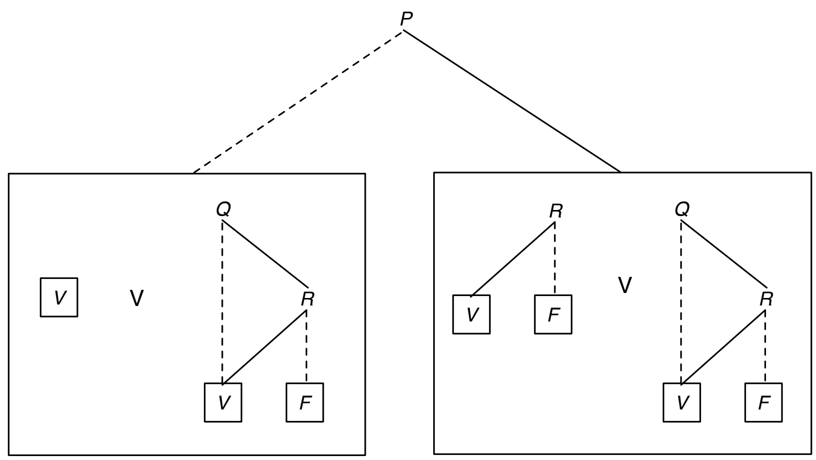


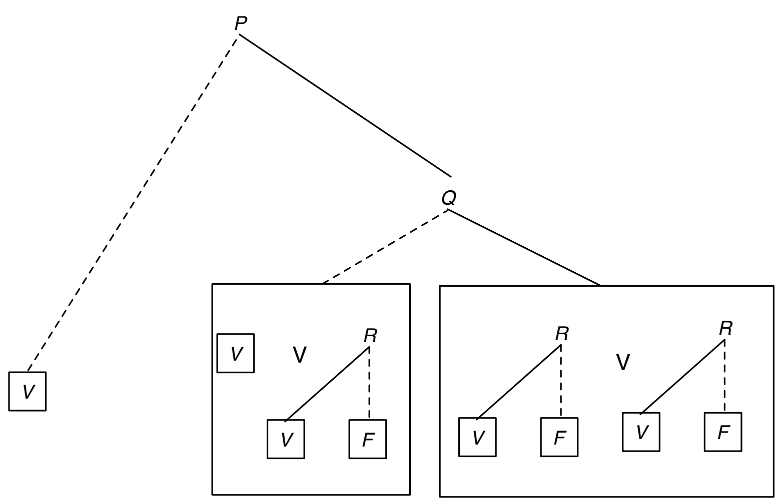


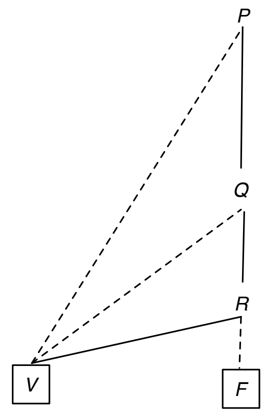
(d) Os OBDDs são estruturalmente semelhantes. Logo, as *fbfs* $(P \vee Q) \to R$ e $(P \to R) \wedge (Q \to R)$ são equivalentes.

(e)




P
Q
V
V
R
V
F
R
V
F
V
Q
R
V
F


P
Q
V
V
V
R
V
F
R
V
F
V
R
V
F



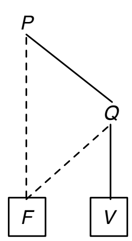
(f) O resultado obtido na alínea anterior permite concluir que $\{(P \to R) \vee (Q \to R)\} \models (P \wedge Q) \to R$. Com efeito, o OBDD anterior permite concluir que a *fbf* $(P \to R) \vee (Q \to R)$ tem os seguintes modelos:

$M_1(P) = F$, $M_1(Q) = V/F$ e $M_1(R) = V/F$,
$M_2(P) = V$, $M_2(Q) = F$ e $M_2(R) = V/F$,
$M_2(P) = V$, $M_2(Q) = V$ e $M_2(R) = V$.

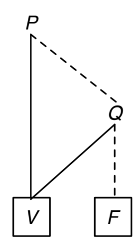
Em qualquer destes modelos a *fbf* $(P \wedge Q) \to R$ é verdadeira.

3.3.12. Considere as *fbfs* $P \wedge Q$ e $P \vee Q$, cujos OBDDs reduzidos são

$P \wedge Q$



$P \vee Q$

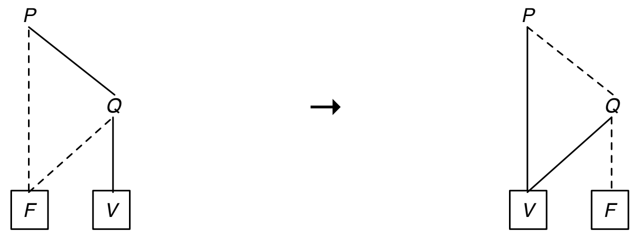


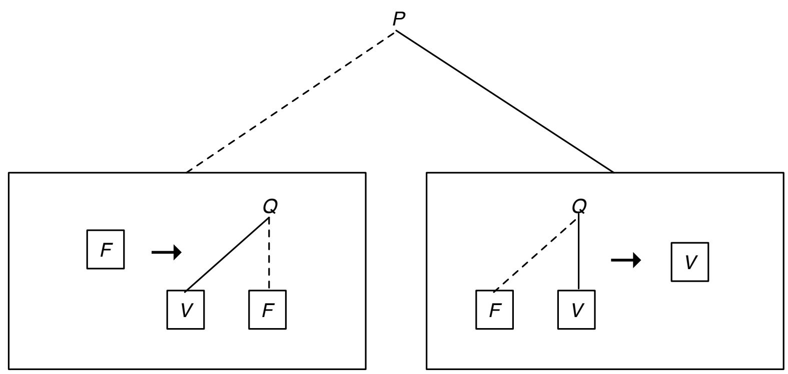
(a) Utilizando o algoritmo *aplica*, determine o OBDD reduzido da *fbf* $(P \wedge Q) \to (P \vee Q)$.

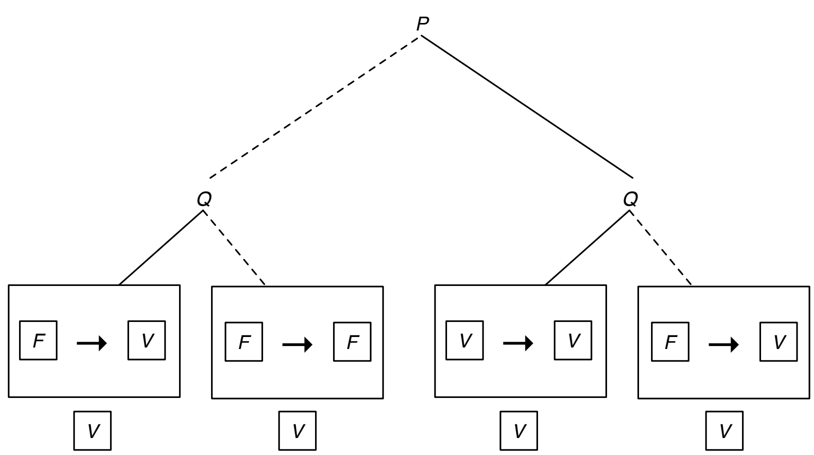
(b) Atendendo ao resultado da alínea anterior, como classifica a *fbf* $(P \wedge Q) \to (P \vee Q)$?

Resposta:

(a)



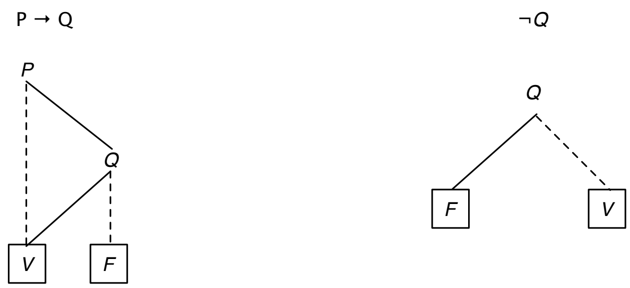




Todas as folhas são $\boxed{V}$, logo o OBDD reduzido consiste na folha $\boxed{V}$.

(b) A *fbf* $(P \wedge Q) \to (P \vee Q)$ é uma tautologia.

3.3.13. Considere as *fbfs* $P \to Q$ e $\neg Q$, cujos OBDDs reduzidos são

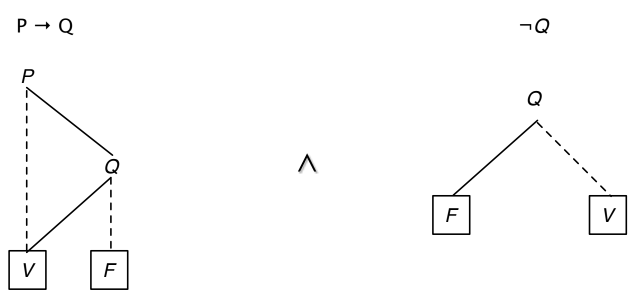


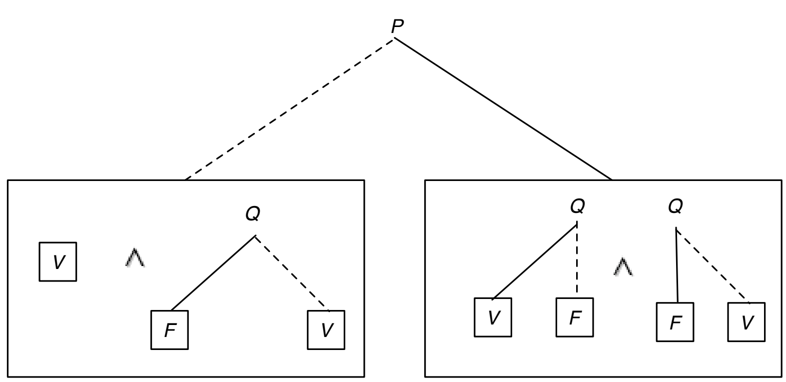
(a) Utilizando o algoritmo *aplica*, determine o OBDD reduzido da *fbf* $(P \to Q) \wedge \neg Q$.

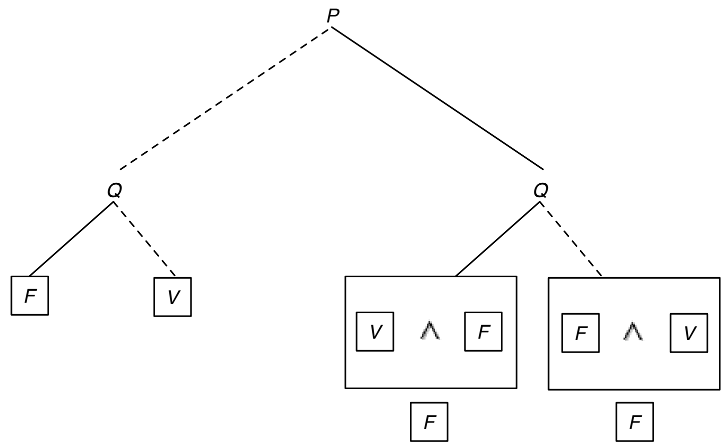
(b) O resultado obtido na alínea anterior permite concluir que $\{P \to Q, \neg Q\} \models \neg P$? Justifique a sua resposta.

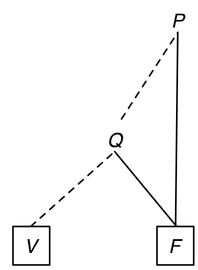
Resposta:

(a)









(b) O resultado obtido na alínea anterior permite concluir que $\{P \to Q, \neg Q\} \models \neg P$. Com efeito, o OBDD anterior permite concluir que a *fbf* $(P \to Q) \wedge \neg Q$ tem 1 modelo:
$M_1(P) = F$ e $M_1(Q) = F$.

Neste modelo, a *fbf* $\neg P$ é verdadeira.

3.3.14. (a) Obtenha os OBDD's reduzidos das *fbfs* P e Q.

(b) Utilizando o algoritmo *aplica*, e o resultado da alínea a), determine o OBDD reduzido da *fbf* $P \vee Q$.

(c) Utilizando o algoritmo *aplica*, e o resultado da alínea a), determine o OBDD reduzido da *fbf* $\neg Q$.

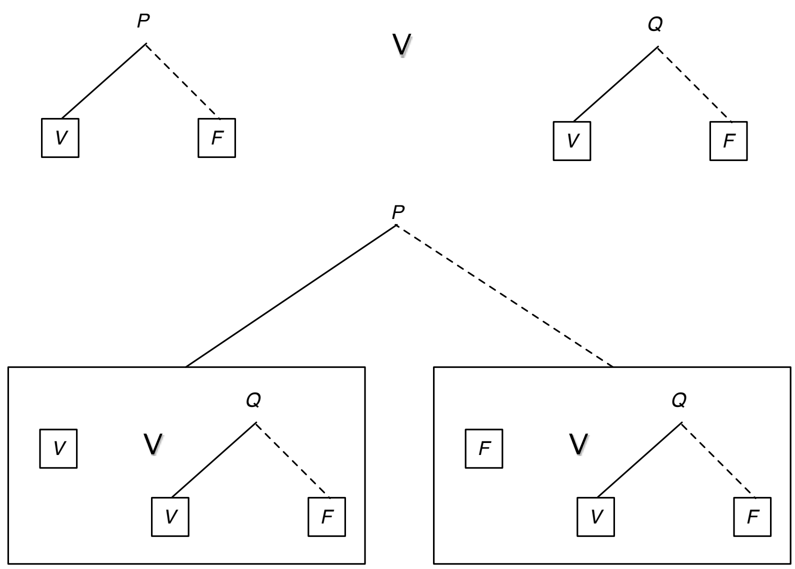
(d) Utilizando o algoritmo *aplica*, e os resultados das alínea b) e c), determine o OBDD reduzido da *fbf* $(P \vee Q) \wedge \neg Q$.

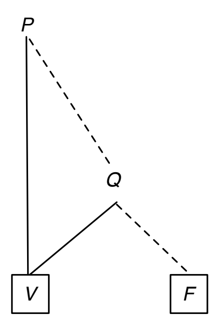
(e) O resultado obtido na alínea anterior permite concluir que $\{P \vee Q, \neg Q\} \models P$? Justifique a sua resposta.

Resposta:

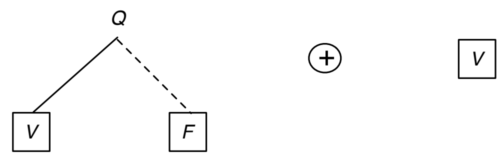
(a)

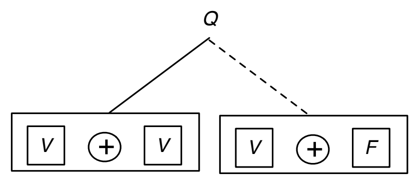
(b)

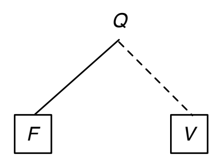




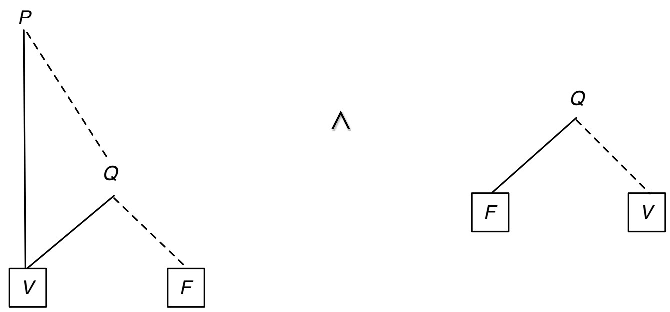
(c)

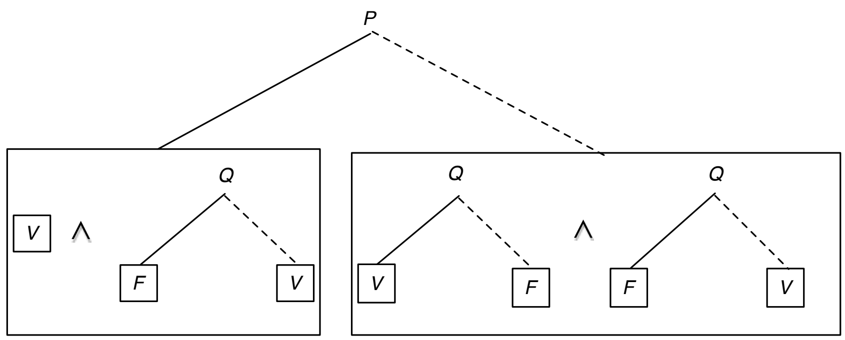


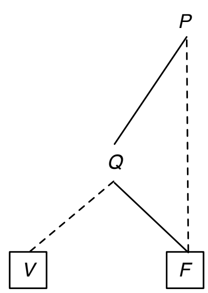




(d)







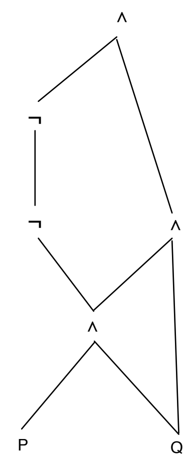
(e) O resultado obtido na alínea anterior permite concluir que $\{P \vee Q, \neg Q\} \models P$. Com efeito, o OBDD anterior permite concluir que a *fbf* $(P \vee Q) \wedge \neg Q$ tem 1 modelo:

$M_1(P) = V$ e $M_1(Q) = F$.

Neste modelo, a *fbf* P é verdadeira.

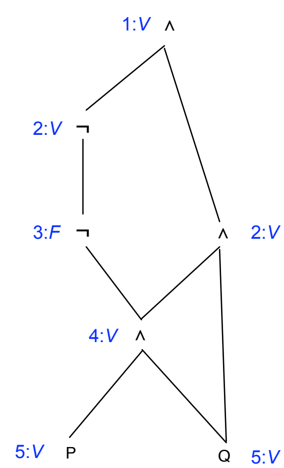
3.4 Algoritmo de propagação de marcas

3.4.1. Considere o seguinte DAG.



Use o algoritmo de propagação de marcas para determinar se a *fbf* correspondente é satisfazível. Em caso afirmativo apresente uma testemunha.

Resposta:



Testemunha:

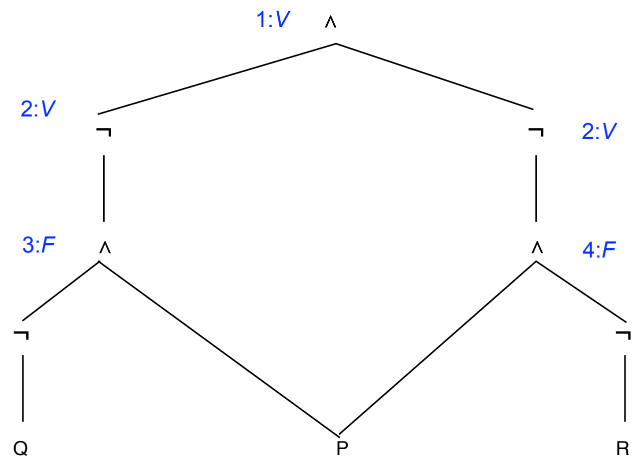
$$I(P) = V,\ I(Q) = V.$$

3.4.2. Use o algoritmo de propagação de marcas para determinar se a *fbf* $(P \to Q) \wedge (P \to R)$ é satisfazível. Em caso afirmativo apresente uma testemunha.

Resposta:

(a) Eliminação do símbolo $\rightarrow$: $\neg(P \wedge \neg Q) \wedge \neg(P \wedge \neg R)$.

(b) Obtenção do DAG da *fbf* $\neg(P \wedge \neg Q) \wedge \neg(P \wedge \neg R)$ e propagação de marcas:



O teste do nó P com a marca V marca os nós Q e R com a marca V. Logo, uma testemunha é

$$I(P) = V,\ I(Q) = V,\ I(R) = V.$$

3.4.3. Usando o algoritmo de propagação de marcas, prove que a *fbf* $((P \vee Q) \wedge \neg Q) \rightarrow P$ é uma tautologia. Sugestão: prove que a negação da *fbf* não é satisfazível.

Resposta:

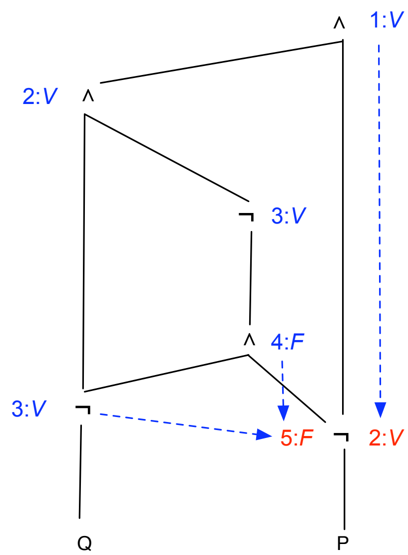
(a) Negação da *fbf*: $\neg(((P \vee Q) \wedge \neg Q) \rightarrow P)$

(b) Eliminação do símbolo $\rightarrow$: $\neg\neg(((P \vee Q) \wedge \neg Q) \wedge \neg P)$

(c) Eliminação da dupla negação: $((P \vee Q) \wedge \neg Q) \wedge \neg P$

(d) Eliminação do símbolo $\vee$: $(\neg(\neg P \wedge \neg Q) \wedge \neg Q) \wedge \neg P$

(e) Obtenção do DAG da *fbf* $(\neg(\neg P \wedge \neg Q) \wedge \neg Q) \wedge \neg P$ e propagação de marcas:

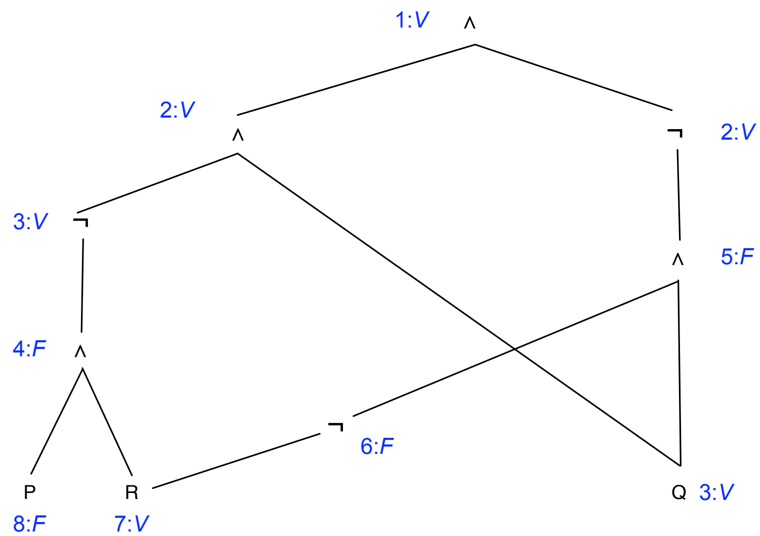


3.4.4. Use o algoritmo de propagação de marcas para determinar se a *fbf* $(\neg P \vee \neg R) \wedge Q \wedge (Q \rightarrow R)$ é satisfazível. Em caso afirmativo apresente uma testemunha.

Resposta:

(a) Eliminação dos símbolos $\rightarrow$ e $\vee$: $\neg(\neg\neg P \wedge \neg\neg R) \wedge Q \wedge \neg(Q \wedge \neg R)$; eliminação da dupla negação: $\neg(P \wedge R) \wedge Q \wedge \neg(Q \wedge \neg R)$

(b) Obtenção do DAG da *fbf* $\neg(P \wedge R) \wedge Q \wedge \neg(Q \wedge \neg R)$ e propagação de marcas:



Testemunha:

$$I(P) = F,\ I(Q) = V,\ I(R) = V.$$

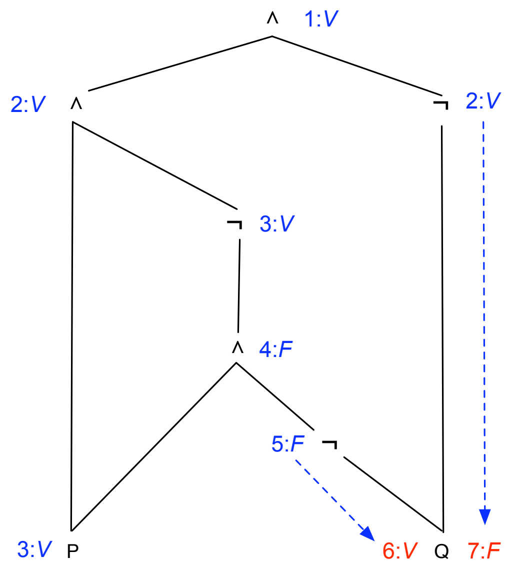
3.4.5. Usando o algoritmo de propagação de marcas, prove que a *fbf* $(P \wedge (P \to Q)) \to Q$ é uma tautologia. Sugestão: prove que a negação da *fbf* não é satisfazível.

Resposta:

(a) Eliminação do símbolo $\to$: $\neg((P \wedge \neg(P \wedge \neg Q)) \wedge \neg Q)$

(b) Negação do resultado: $\neg\neg((P \wedge \neg(P \wedge \neg Q)) \wedge \neg Q)$;
eliminação da dupla negação: $(P \wedge \neg(P \wedge \neg Q)) \wedge \neg Q$

(c) Obtenção do DAG da *fbf* $(P \wedge \neg(P \wedge \neg Q)) \wedge \neg Q$ e propagação de marcas:

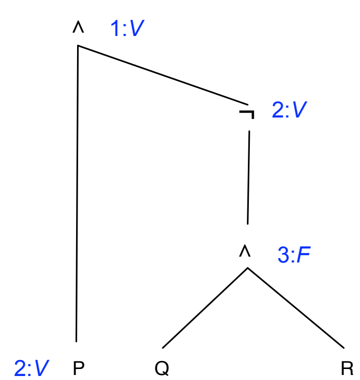


3.4.6. Use o algoritmo de propagação de marcas para determinar se a *fbf* $P \wedge (\neg Q \vee \neg R)$ é satisfazível. Em caso afirmativo apresente uma testemunha.

Resposta:

(a) Eliminação do símbolo $\vee$: $P \wedge \neg(\neg\neg Q \wedge \neg\neg R)$;
eliminação da dupla negação: $P \wedge \neg(Q \wedge R)$

(b) Obtenção do DAG da *fbf* $P \wedge \neg(Q \wedge R)$ e propagação de marcas:



(c) Como os nós Q e R ficaram por marcar, é aplicado o algoritmo de teste de nós. O teste de qualquer destes nós com a marca V, marca o outro nó com F. Assim, uma testemunha é:

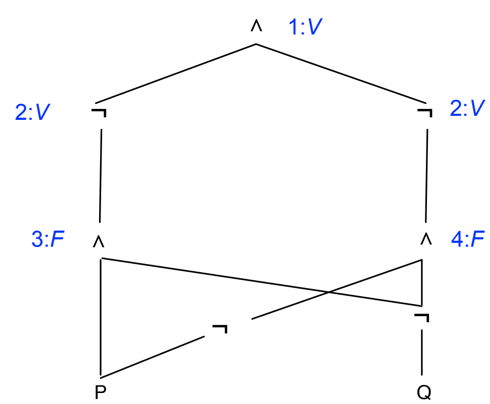
$$I(P) = V,\ I(Q) = V,\ I(R) = F.$$

3.4.7. Use o algoritmo de propagação de marcas para determinar se a *fbf* $(P \to Q) \wedge (\neg P \to Q)$ é satisfazível. Em caso afirmativo apresente uma testemunha.

Resposta:

(a) Eliminação do símbolo $\to$: $\neg(P \wedge \neg Q) \wedge \neg(\neg P \wedge \neg Q)$

(b) Obtenção do DAG da *fbf* $\neg(P \wedge \neg Q) \wedge \neg(\neg P \wedge \neg Q)$ e propagação de marcas:



(c) Como os nós P e Q ficaram por marcar, é aplicado o algoritmo de teste de nós. Apresentam-se as várias alternativas (uma delas seria suficiente):

- O teste do nó P com a marca V, marca o nó Q com V. Obtém-se uma marcação completa e consistente. Assim, uma testemunha é:

$$I(P) = V,\ I(Q) = V.$$

- O teste do nó P com a marca F, marca o nó Q com V. Obtém-se uma marcação completa e consistente. Assim, uma testemunha é:

$$I(P) = F,\ I(Q) = V.$$

- O teste do nó Q com a marca V, deixa o nó P por marcar. Assim, o nó Q seria agora testado com a marca F. Este teste leva a uma contradição. Logo, o nó Q é marcado com a marca permanente V. A propagação desta marca não consegue marcar P. Logo, seria novamente aplicado o algoritmo de teste de nós. O teste de P com qualquer marca leva a uma marcação completa e consistente. Assim, obteríamos uma das duas seguintes testemunhas:

$$I(P) = V,\ I(Q) = V \quad ou \quad I(P) = F,\ I(Q) = V.$$

3.5 Algoritmos baseados em DP

3.5.1. Escolha as respostas corretas.

(a) Um conjunto de cláusulas vazio corresponde a uma

i. tautologia.

ii. contradição.

Resposta: ____

Resposta:

i.

(b) Um conjunto de cláusulas contendo a cláusula vazia corresponde a uma

i. tautologia.

ii. contradição.

Resposta: ____

Resposta:

ii.

3.5.2. Complete a frase seguinte:

No algoritmo DP, a escolha da ordem pela qual são eliminados os símbolos de proposição pode influenciar ________________________ ____.

Resposta:

No algoritmo DP, a escolha da ordem pela qual são eliminados os símbolos de proposição pode influenciar <u>*a quantidade de processamento necessária*</u>*.*

3.5.3. Use o algoritmo DP para provar que $\{P, P \to Q\} \models Q$. Sugestão: use o teorema da refutação.

Resposta:

Usando o teorema de refutação, provaremos que o conjunto $\{P, P \to Q, \neg Q\}$ não é satisfazível.

(a) *Passagem à forma clausal:*
$\{\{P\}, \{\neg P, Q\}, \{\neg Q\}\}$.

(b) *Aplicação do algoritmo DP ao conjunto de cláusulas obtido:*
Por eliminação de P e Q, por esta ordem, obtemos, sucessivamente, os conjuntos:

$$\exists P(\Delta) = \{\{Q\}, \{\neg Q\}\}$$
$$\exists Q(\exists P(\Delta)) = \{\{\}\}$$

Como o último conjunto contém a cláusula vazia, podemos concluir que o conjunto $\{\{P\}, \{\neg P, Q\}, \{\neg Q\}\}$ não é satisfazível.

3.5.4. Use o algoritmo DP para provar que $\{P \vee Q, P \to R, Q \to R\} \models R$. Sugestão: use o teorema da refutação.

Resposta:

Usando o teorema de refutação, provaremos que o conjunto $\{P \vee Q, P \to R, Q \to R, \neg R\}$ não é satisfazível.

(a) *Passagem à forma clausal:*
$\{\{P, Q\}, \{\neg P, R\}, \{\neg Q, R\}, \{\neg R\}\}$.

(b) *Aplicação do algoritmo DP ao conjunto de cláusulas obtido:*
Por eliminação de P, Q e R, por esta ordem, obtemos, sucessivamente, os conjuntos:

$$\begin{aligned} \exists P(\Delta) &= \{\{Q, R\}, \{\neg Q, R\}, \{\neg R\}\} \\ \exists Q(\exists P(\Delta)) &= \{\{R\}, \{\neg R\}\} \\ \exists R(\exists Q(\exists P(\Delta))) &= \{\{\}\} \end{aligned}$$

Como o último conjunto contém a cláusula vazia, podemos concluir que o conjunto $\{P \vee Q, P \to R, Q \to R, \neg R\}$ não é satisfazível.

3.5.5. Considere o seguinte conjunto de cláusulas:

$$\{\{P, Q, \neg R\}, \{P, \neg Q, S\}, \{R, \neg S\}\}$$

Estabelecendo a ordem correspondente à ordem alfabética entre os símbolos de proposição, use o algoritmo DP recorrendo a baldes para determinar se o conjunto é satisfazível. Em caso afirmativo apresente uma testemunha.

Resposta:

(a) *Criação e preenchimento dos baldes:*
b_P : $\{P, Q, \neg R\}, \{P, \neg Q, S\}$
b_Q :
b_R : $\{R, \neg S\}$
b_S :

(b) *Processamento dos baldes:* Não são geradas novas cláusulas. Como não foi gerada a cláusula vazia, o conjunto é satisfazível.

(c) *Inspeção dos baldes:*
$I(S) = V$ (escolha)
$I(R) = V$ (obrigatório)
$I(Q) = V$ (escolha)
$I(P) = V$ (escolha)

3.5.6. Considere o seguinte conjunto de cláusulas:

$$\{\{\neg P, Q\}, \{\neg P, \neg Q\}, \{P, R\}, \{\neg R\}\}$$

Estabelecendo a ordem correspondente à ordem alfabética entre os símbolos de proposição, use o algoritmo DP recorrendo a baldes para determinar se o conjunto é satisfazível. Em caso afirmativo apresente uma testemunha.

Resposta:

(a) *Criação e preenchimento dos baldes:*
b_P : $\{\neg P, Q\}, \{\neg P, \neg Q\}, \{P, R\}$
b_Q :
b_R : $\{\neg R\}$

(b) *Processamento dos baldes:*
- Processamento do balde b_P:
  b_P : $\{\neg P, Q\}, \{\neg P, \neg Q\}, \{P, R\}$
  b_Q : $\{Q, R\}, \{\neg Q, R\}$
  b_R : $\{\neg R\}$
- Processamento do balde b_Q:
  b_P : $\{\neg P, Q\}, \{\neg P, \neg Q\}, \{P, R\}$
  b_Q : $\{Q, R\}, \{\neg Q, R\}$
  b_R : $\{\neg R\}$ $\{R\}$
- Processamento do balde b_R: É gerada a cláusula vazia, logo o conjunto não é satisfazível.

3.5.7. Considere o seguinte conjunto de cláusulas:

$$\{\{\neg P, Q\}, \{\neg Q, R\}, \{P, Q\}, \{\neg R\}\}$$

Estabelecendo a ordem correspondente à ordem alfabética entre os símbolos de proposição, use o algoritmo DP recorrendo a baldes para determinar se o conjunto é satisfazível. Em caso afirmativo apresente uma testemunha.

Resposta:

(a) *Criação e preenchimento dos baldes:*
b_P : $\{\neg P, Q\}, \{P, Q\}$
b_Q : $\{\neg Q, R\}$
b_R : $\{\neg R\}$

(b) *Processamento dos baldes:*

• Processamento do balde b_P:

$$\begin{array}{lll} b_P: & \{\neg P, Q\}, \{P, Q\} & \\ b_Q: & \{\neg Q, R\} & \{Q\} \\ b_R: & \{\neg R\} & \end{array}$$

• Processamento do balde b_Q:

$$\begin{array}{llll} b_P: & \{\neg P, Q\}, \{P, Q\} & & \\ b_Q: & \{\neg Q, R\} & \{Q\} & \\ b_R: & \{\neg R\} & & \{R\} \end{array}$$

• Processamento do balde b_R: É gerada a cláusula vazia, logo o conjunto não é satisfazível.

3.5.8. Considere o seguinte conjunto de cláusulas:

$$\Delta = \{\{\neg P, Q\}, \{\neg Q, R\}, \{P, Q\}, \{\neg R\}\}$$

Usando os resultados do exercício anterior, e sem fazer novos cálculos, complete as seguintes igualdades:

$$\begin{aligned} \exists P(\Delta) &= \\ \exists Q(\exists P(\Delta)) &= \\ \exists R(\exists Q(\exists P(\Delta))) &= \end{aligned}$$

Resposta:

$$\begin{aligned} \exists P(\Delta) &= \{\{\neg Q, R\}, \{Q\}, \{\neg R\}\} \\ \exists Q(\exists P(\Delta)) &= \{\{\neg R\}, \{R\}\} \\ \exists R(\exists Q(\exists P(\Delta))) &= \{\{\}\} \end{aligned}$$

3.5.9. Considere o seguinte conjunto de cláusulas:

$$\{\{P, Q, \neg R\}, \{\neg P, \neg Q\}, \{R\}\}$$

Estabelecendo a ordem correspondente à ordem alfabética entre os símbolos de proposição, use o algoritmo DP recorrendo a baldes para determinar se o conjunto é satisfazível. Em caso afirmativo apresente uma testemunha.

Resposta:

(a) *Criação e preenchimento dos baldes:*

$$\begin{array}{ll} b_P: & \{P, Q, \neg R\}, \{\neg P, \neg Q\} \\ b_Q: & \\ b_R: & \{R\} \end{array}$$

(b) *Processamento dos baldes:* A única cláusula gerada é a cláusula $\{Q, \neg R, \neg Q\}$; como se trata de um teorema pode ser ignorada. Logo, após o processamento os baldes permanecem iguais. Como não foi gerada a cláusula vazia, o conjunto é satisfazível.

(c) *Inspeção dos baldes:*
$I(R) = V$ (obrigatório)
$I(Q) = V$ (escolha)
$I(P) = F$ (obrigatório)

3.5.10. Considere o seguinte conjunto de cláusulas:

$$\{\{P, Q\}, \{P, \neg R\}, \{\neg R, Q\}, \{\neg R\}, \{\neg P, R\}, \{\neg Q, R\}\}$$

Estabelecendo a ordem correspondente à ordem alfabética entre os símbolos de proposição, use o algoritmo DP recorrendo a baldes para determinar se o conjunto é satisfazível. Em caso afirmativo apresente uma testemunha.

Resposta:

(a) *Criação e preenchimento dos baldes:*
b_P : $\{P, Q\}, \{P, \neg R\}, \{\neg P, R\}$
b_Q : $\{\neg R, Q\}, \{\neg Q, R\}$
b_R : $\{\neg R\}$

(b) *Processamento dos baldes:*

- Processamento do balde b_P:
  b_P : $\{P, Q\}, \{P, \neg R\}, \{\neg P, R\}$
  b_Q : $\{\neg R, Q\}, \{\neg Q, R\}$ $\{Q, R\}$
  b_R : $\{\neg R\}$
  Também foi gerada a cláusula $\{R, \neg R\}$ que, por ser um teorema, foi ignorada.

- Processamento do balde b_Q:
  b_P : $\{P, Q\}, \{P, \neg R\}, \{\neg P, R\}$
  b_Q : $\{\neg R, Q\}, \{\neg Q, R\}$ $\{Q, R\}$
  b_R : $\{\neg R\}$ $\{R\}$
  Novamente foi gerada a cláusula $\{R, \neg R\}$ que, por ser um teorema, foi ignorada.

- Processamento do balde b_R: É gerada a cláusula vazia, logo o conjunto não é satisfazível.

3.5.11. Usando os resultados do exercício anterior, diga qual a relação existente entre o conjunto de cláusulas

$$\Delta = \{\{P, Q\}, \{P, \neg R\}, \{\neg R, Q\}, \{\neg P, R\}, \{\neg Q, R\}\}$$

e a cláusula $\{R\}$. Justifique a sua resposta.

Resposta:

Temos que $\Delta \models \{R\}$. Com efeito, pelo teorema da dedução, $\Delta \models \{R\}$ se e só se $\Delta \cup \{\{\neg R\}\}$ não for satisfazível, o que foi provado no exercício anterior.

Capítulo 4

Lógica de Primeira Ordem (I)

4.1 Sistema dedutivo

4.1.1. Demonstre os seguintes argumentos, completando as provas apresentadas:

(a) $(\{\forall x[\neg P(x)]\}, \forall x[P(x) \to \neg Q(x)])$

1	$\forall x[\neg P(x)]$	Prem
2	x_0 $P(x_0)$	Hip
3	$Q(x_0)$	Hip

(b) $(\{\forall x[P(x) \wedge (Q(x) \vee \neg R(x))], \forall x[S(x) \to P(x)], \forall x[S(x) \to \neg P(x)]\}, \neg\forall x[R(x) \wedge S(x)])$

1	$\forall x[S(x) \to P(x)]$	Prem
2	$\forall x[S(x) \to \neg P(x)]$	Prem
3	$\forall x[R(x) \wedge S(x)]$	Hip

(c) $(\{\forall x[P(x) \to Q(x)], \forall x[P(x) \to \neg Q(x)], \exists x[P(x)]\}, \forall x[Q(x)])$

1	$\forall x[P(x) \to Q(x)]$	Prem
2	$\forall x[P(x) \to \neg Q(x)]$	Prem
3	$\exists x[P(x)]$	Prem
4	x_0 $P(x_0)$	Hip
5	$\neg\forall x[Q(x)]$	Hip

(d) $(\{\forall x[\neg P(x)]\}, \forall x[P(x) \to (\neg Q(x) \vee S(x))])$

1		$\forall x[\neg P(x)]$	Prem
2	x_0	$P(x_0)$	Hip
3		$Q(x_0)$	Hip

(e) $(\{\forall x[P(x) \to Q(x)], \forall x[P(x) \to \neg Q(x)], \forall x[Q(x) \to \neg R(x)]\}, \neg\forall x[R(x) \wedge P(x)])$

1	$\forall x[P(x) \to Q(x)]$	Prem
2	$\forall x[P(x) \to \neg Q(x)]$	Prem
3	$\forall x[R(x) \wedge P(x)]$	Hip

Resposta:

(a)

1		$\forall x[\neg P(x)]$	Prem
2	x_0	$P(x_0)$	Hip
3		$Q(x_0)$	Hip
4		$P(x_0)$	Rei, 2
5		$\forall x[\neg P(x)]$	Rei, 1
6		$\neg P(x_0)$	E$\forall$, 5
7		$\neg Q(x_0)$	I$\neg$, (3, (4, 6))
8		$P(x_0) \to \neg Q(x_0)$	I$\to$, (2, 7)
9		$\forall x[P(x) \to \neg Q(x)]$	I$\forall$, (2, 8)

(b)

1	$\forall x[S(x) \to P(x)]$	Prem
2	$\forall x[S(x) \to \neg P(x)]$	Prem
3	$\mid\ \underline{\forall x[R(x) \land S(x)]}$	Hip
4	$\mid\ R(a) \land S(a)$	E$\forall$, 3
5	$\mid\ S(a)$	E$\land$, 4
6	$\mid\ \forall x[S(x) \to P(x)]$	Rei, 1
7	$\mid\ \forall x[S(x) \to \neg P(x)]$	Rei, 2
8	$\mid\ S(a) \to P(a)$	E$\forall$, 6
9	$\mid\ S(a) \to \neg P(a)$	E$\forall$, 7
10	$\mid\ P(a)$	E$\to$, (5, 8)
11	$\mid\ \neg P(a)$	E$\to$, (5, 9)
12	$\neg\forall x[R(x) \land S(x)]$	I$\neg$, (3, (10, 11))

(c)

1	$\forall x[P(x) \to Q(x)]$	Prem
2	$\forall x[P(x) \to \neg Q(x)]$	Prem
3	$\exists x[P(x)]$	Prem
4	$x_0\ \mid\ \underline{P(x_0)}$	Hip
5	$\mid\ \mid\ \underline{\neg\forall x[Q(x)]}$	Hip
6	$\mid\ \mid\ P(x_0)$	Rei, 4
7	$\mid\ \mid\ \forall x[P(x) \to Q(x)]$	Rei, 1
8	$\mid\ \mid\ P(x_0) \to Q(x_0)$	E$\forall$, 7
9	$\mid\ \mid\ Q(x_0)$	E$\to$, (6, 8)
10	$\mid\ \mid\ \forall x[P(x) \to \neg Q(x)]$	Rei, 2
11	$\mid\ \mid\ P(x_0) \to \neg Q(x_0)$	E$\forall$, 10
12	$\mid\ \mid\ \neg Q(x_0)$	E$\to$, (6, 11)
13	$\mid\ \neg\neg\forall x[Q(x)]$	I$\neg$, (5, (9, 12))
14	$\mid\ \forall x[Q(x)]$	E$\neg$, 13
15	$\forall x[Q(x)]$	E$\exists$, (4, 14)

(d)

1		$\forall x[\neg P(x)]$	Prem
2	x_0	$P(x_0)$	Hip
3		$Q(x_0)$	Hip
4		$P(x_0)$	Rei, 2
5		$\forall x[\neg P(x)]$	Rei, 1
6		$\neg P(x_0)$	E$\forall$, 5
7		$\neg Q(x_0)$	I$\neg$, (3, (4, 6))
8		$\neg Q(x_0) \vee S(x_0)$	I$\vee$, 7
9		$P(x_0) \to (\neg Q(x_0) \vee S(x_0))$	I$\to$, (2, 8)
10		$\forall x[P(x) \to (\neg Q(x) \vee S(x))]$	I$\forall$, (2, 8)

(e)

1	$\forall x[P(x) \to Q(x)]$	Prem
2	$\forall x[P(x) \to \neg Q(x)]$	Prem
3	$\forall x[R(x) \wedge P(x)]$	Hip
4	$R(a) \wedge P(a)$	E$\forall$, 3
5	$P(a)$	E$\wedge$, 4
6	$\forall x[P(x) \to Q(x)]$	Rei, 1
7	$\forall x[P(x) \to \neg Q(x)]$	Rei, 2
8	$P(a) \to Q(a)$	E$\forall$, 6
9	$P(a) \to \neg Q(a)$	E$\forall$, 7
10	$Q(a)$	E$\to$, (5, 8)
11	$\neg Q(a)$	E$\to$, (5, 9)
12	$\neg\forall x[R(x) \wedge P(x)]$	I$\neg$, (3, (10, 11))

4.1.2. Demonstre os seguintes teoremas e argumentos usando o sistema de dedução natural. Apenas pode utilizar as regras de inferência básicas do sistema de dedução natural (Prem, Rep, Hip, Rei, I$\wedge$, E$\wedge$, I$\vee$, E$\vee$, I$\neg$, E$\neg$, I$\to$, E$\to$, I$\forall$, E$\forall$, I$\exists$, E$\exists$).

(a) $(\forall x[P(x)] \wedge \forall x[Q(x)]) \to \forall x[P(x) \wedge Q(x)]$

(b) $(\forall x[P(x)] \vee \forall x[Q(x)]) \to \forall x[P(x) \vee Q(x)]$

(c) $\{\forall x[P(x) \wedge Q(x)] \vee \forall x[P(x) \wedge R(x)]\} \vdash \forall x[P(x)]$

(d) $(\forall x[R(x)] \wedge \forall x[P(x)]) \rightarrow \forall x[(R(x) \wedge P(x)) \vee Q(x)]$

(e) $\{\forall x[\neg P(x)]\} \vdash \forall x[P(x) \rightarrow Q(x)]$

(f) $\{\exists x[P(x)], \forall x[P(x) \rightarrow Q(x)]\} \vdash \exists x[Q(x)]$

(g) $(\forall x[P(x) \rightarrow Q(x)] \wedge \forall x[P(x) \rightarrow \neg Q(x)]) \rightarrow \forall x[\neg P(x)]$

(h) $\{\forall x[P(x) \rightarrow R(x)], \forall x[Q(x) \rightarrow R(x)], P(a) \vee Q(a)\} \vdash R(a)$

(i) $\forall x[P(x) \vee \neg P(x)]$

(j) $\{\forall x[P(x) \rightarrow Q(x)], \forall x[P(x) \rightarrow \neg Q(x)])\} \vdash \neg\exists x[P(x)]$

Resposta:

(a)

1		$\forall x[P(x)] \wedge \forall x[Q(x)]$	Hip
2	x_0	$\forall x[P(x)] \wedge \forall x[Q(x)]$	Rei, 1
3		$\forall x[P(x)]$	E$\wedge$, 2
4		$\forall x[Q(x)]$	E$\wedge$, 2
5		$P(x_0)$	E$\forall$, 3
6		$Q(x_0)$	E$\forall$, 4
7		$P(x_0) \wedge Q(x_0)$	I$\wedge$, (5, 6)
8		$\forall x[P(x) \wedge Q(x)]$	I$\forall$, (2, 7)
9		$(\forall x[P(x)] \wedge \forall x[Q(x)]) \rightarrow \forall x[P(x) \wedge Q(x)]$	I$\rightarrow$, (1, 8)

(b)

1	$\forall x[P(x)] \vee \forall x[Q(x)]$	Hip
2	x_0 $\forall x[P(x)] \vee \forall x[Q(x)]$	Rei, 1
3	$\forall x[P(x)]$	Hip
4	$P(x_0)$	E$\forall$, 3
5	$P(x_0) \vee Q(x_0)$	I$\vee$, 4
6	$\forall x[Q(x)]$	Hip
7	$Q(x_0)$	E$\forall$, 6
8	$P(x_0) \vee Q(x_0)$	I$\vee$, 7
9	$P(x_0) \vee Q(x_0)$	E$\vee$, (2, (3, 5), (6, 8))
10	$\forall x[P(x) \vee Q(x)]$	I$\forall$, (3, 9)
11	$(\forall x[P(x)] \vee \forall x[Q(x)]) \to \forall x[P(x) \vee Q(x)]$	I$\to$, (1, 10)

(c)

1	$\forall x[P(x) \wedge Q(x)] \vee \forall x[P(x) \wedge R(x)]$	Prem
2	x_0 $\forall x[P(x) \wedge Q(x)] \vee \forall x[P(x) \wedge R(x)]$	Rei, 1
3	$\forall x[P(x) \wedge Q(x)]$	Hip
4	$P(x_0) \wedge Q(x_0)$	E$\forall$, 2
5	$P(x_0)$	E$\wedge$, 4
6	$\forall x[P(x) \wedge R(x)]$	Hip
7	$P(x_0) \wedge R(x_0)$	E$\forall$, 6
8	$P(x_0)$	E$\wedge$, 7
9	$P(x_0)$	E$\vee$, (2, (3, 5), (6, 8))
10	$\forall x[P(x)]$	I$\forall$, (2, 8)

(d)

1		$\forall x[R(x)] \wedge \forall x[P(x)]$	Hip
2	x_0	$\forall x[R(x)] \wedge \forall x[P(x)]$	Rei, 1
3		$\forall x[R(x)]$	E$\wedge$, 2
4		$\forall x[P(x)]$	E$\wedge$, 2
5		$R(x_0)$	E$\forall$, 3
6		$P(x_0)$	E$\forall$, 4
7		$R(x_0) \wedge P(x_0)$	I$\wedge$, (5, 6)
8		$(R(x_0) \wedge P(x_0)) \vee Q(x_0)$	I$\vee$, 7
9		$\forall x[(R(x) \wedge P(x)) \vee Q(x)]$	I$\forall$, (2, 8)
10		$(\forall x[R(x)] \wedge \forall x[P(x)]) \to \forall x[(R(x) \wedge P(x)) \vee Q(x)]$	I$\to$, (1, 9)

(e)

1		$\forall x[\neg P(x)]$	Prem
2	x_0	$P(x_0)$	Hip
3		$\neg Q(x_0)$	Hip
4		$P(x_0)$	Rei, 2
5		$\forall x[\neg P(x)]$	Rei, 1
6		$\neg P(x_0)$	E$\forall$, 5
7		$\neg\neg Q(x_0)$	I$\neg$, (3, (4, 6))
8		$Q(x_0)$	E$\neg$, 7
9		$P(x_0) \to Q(x_0)$	I$\to$, (2, 9)
10		$\forall x[P(x) \to Q(x)]$	I$\forall$, (2, 9)

(f)

1		$\exists x[P(x)]$	Prem
2		$\forall x[P(x) \to Q(x)]$	Prem
3	x_0	$P(x_0)$	Hip
4		$\forall x[P(x) \to Q(x)]$	Rei, 2
5		$P(x_0) \to Q(x_0)$	E$\forall$, 4
6		$Q(x_0)$	E$\to$, (3, 5)
7		$\exists x[Q(x)]$	I$\exists$, 6
8		$\exists x[Q(x)]$	E$\exists$, (1, (3, 7))

(g)

1		$\forall x[P(x) \to Q(x)] \land \forall x[P(x) \to \neg Q(x)]$	Hip
2		$\forall x[P(x) \to Q(x)]$	E$\land$, 1
3		$\forall x[P(x) \to \neg Q(x)]$	E$\land$, 1
4	x_0	$\forall x[P(x) \to Q(x)]$	Rei, 2
5		$\forall x[P(x) \to \neg Q(x)]$	Rei, 3
6		$P(x_0)$	Hip
7		$\forall x[P(x) \to Q(x)]$	Rei, 4
8		$P(x_0) \to Q(x_0)$	E$\forall$, 7
9		$Q(x_0)$	E$\to$, (6, 8)
10		$\forall x[P(x) \to \neg Q(x)]$	Rei, 5
11		$P(x_0) \to \neg Q(x_0)$	E$\forall$, 10
12		$\neg Q(x_0)$	E$\to$, (6, 11)
13		$\neg P(x_0)$	I$\neg$, (6, (9, 12))
14		$\forall x[\neg P(x)]$	I$\forall$, (4, 13)
15		$(\forall x[P(x) \to Q(x)] \land \forall x[P(x) \to \neg Q(x)]) \to \forall x[\neg P(x)]$	I$\to$, (1, 14)

(h)

1	$\forall x[P(x) \to R(x)]$	Prem
2	$\forall x[Q(x) \to R(x)]$	Prem
3	$P(a) \lor Q(a)$	Prem
4	$P(a) \to R(a)$	E$\forall$, 1
5	$Q(a) \to R(a)$	E$\forall$, 2
6	$\quad P(a)$	Hip
7	$\quad P(a) \to R(a)$	Rei, 4
8	$\quad R(a)$	E$\to$, (6, 7)
9	$\quad Q(a)$	Hip
10	$\quad Q(a) \to R(a)$	Rei, 5
11	$\quad R(a)$	E$\to$, (9, 10)
12	$R(a)$	E$\lor$, (3, (6, 8), (9, 11))

(i)

1	x_0	$\quad\quad \neg(P(x_0) \lor \neg P(x_0))$	Hip
2		$\quad\quad\quad P(x_0)$	Hip
3		$\quad\quad\quad P(x_0) \lor \neg P(x_0)$	I$\lor$, 2
4		$\quad\quad\quad \neg(P(x_0) \lor \neg P(x_0))$	Rei, 1
5		$\quad\quad \neg P(x_0)$	I$\neg$, (2, (3, 4))
6		$\quad\quad P(x_0) \lor \neg P(x_0)$	I$\lor$, 5
7		$\quad\quad \neg(P(x_0) \lor \neg P(x_0))$	Rep, 1
8		$\quad \neg\neg(P(x_0) \lor \neg P(x_0))$	I$\neg$, (1, (6, 7))
9		$\quad P(x_0) \lor \neg P(x_0)$	E$\neg$, 8
10		$\forall x[P(x) \lor \neg P(x)]$	I$\forall$, (1, 9)

(j) Na prova abaixo é usado o teorema $(P \wedge \neg P) \to Q$.

1	$\forall x[P(x) \to Q(x)]$	Prem
2	$\forall x[P(x) \to \neg Q(x)]$	Prem
3	$\exists x[P(x)]$	Hip
4	x_0 $P(x_0)]$	Hip
5	$\forall x[P(x) \to Q(x)]$	Rei, 1
6	$P(x_0) \to Q(x_0)$	E$\forall$, 5
7	$Q(x_0)$	E$\to$, (4, 6)
8	$\forall x[P(x) \to \neg Q(x)]$	Rei, 2
9	$P(x_0) \to \neg Q(x_0)$	E$\forall$, 8
10	$\neg Q(x_0)$	E$\to$, (4, 9)
11	$Q(x_0) \wedge \neg Q(x_0)$	I$\wedge$, (7, 10)
12	$(Q(x_0) \wedge \neg Q(x_0)) \to (R \wedge \neg R)$	Teo
13	$R \wedge \neg R$	E$\to$, (11, 12)
14	$R \wedge \neg R$	E$\exists$, (3, (4, 13))
15	R	E$\wedge$, 14
16	$\neg R$	E$\wedge$, 14
17	$\neg\exists x[P(x)]$	I$\neg$, (3, (15, 16))

4.2 Sistema semântico

4.2.1. Considere a conceptualização (D, F, R) em que:

$$D = \{\Diamond, \Box, \odot\}$$
$$F = \{\}$$
$$R = \{\{(\odot)\}, \{(\Diamond), (\Box)\}, \{(\Diamond, \odot), (\Diamond, \Box), (\Box, \odot), (\odot, \Box)\}\}.$$

Considere a interpretação I: $\{a, b, c, P, Q, S\} \mapsto D \cup F \cup R$, tal que:

$$I(a) = \Diamond$$
$$I(b) = \Box$$
$$I(c) = \odot$$
$$I(P) = \{(\odot)\}$$

$I(Q) = \{(\Diamond), (\Box)\}$
$I(S) = \{(\Diamond, \odot), (\Diamond, \Box), (\Box, \odot), (\odot, \Box)\}$

Considere o conjunto de *fbfs*

$$\Delta = \{P(c), Q(a) \wedge Q(b), S(c, b), \forall x[Q(x) \to S(x, c)]\}$$

(a) Diga, justificando, quais as *fbfs* de Δ que são satisfeitas pela interpretação I.

(b) A interpretação I é um modelo de Δ? Porquê?

Resposta:

(a)
- $I(c) = \odot$ e $(\odot) \in I(P)$; logo, $\models_I P(c)$.
- $I(a) = \Diamond$ e $(\Diamond) \in I(Q)$; logo, $\models_I Q(a)$.
  $I(b) = \Box$ e $(\Box) \in I(Q)$; logo, $\models_I Q(b)$.
  Logo, $\models_I Q(a) \wedge Q(b)$
- $I(c) = \odot$, $I(b) = \Box$ e $(\odot, \Box) \in I(S)$; logo, $\models_I S(c, b)$.
- Existem 3 constantes na linguagem: a, b e c. Assim, teremos que provar que,
  para qualquer $y \in \{a, b, c\}$, $\models_I ((Q(x) \to S(x, c)) \cdot \{y/x\})$.
  - $(Q(x) \to S(x, c)) \cdot \{a/x\} = Q(a) \to S(a, c)$:
    $I(a) = \Diamond$, $I(c) = \odot$ e $(\Diamond, \odot) \in I(S)$; logo, $\models_I S(a, c)$ e
    $\models_I Q(a) \to S(a, c)$.
  - $(Q(x) \to S(x, c)) \cdot \{b/x\} = Q(b) \to S(b, c)$:
    $I(b) = \Box$, $I(c) = \odot$ e $(\Box, \odot) \in I(S)$; logo, $\models_I S(b, c)$ e
    $\models_I Q(b) \to S(b, c)$.
  - $(Q(x) \to S(x, c)) \cdot \{c/x\} = Q(c) \to S(c, c)$:
    $I(c) = \odot$ e $(\odot) \notin I(Q)$; logo, $\not\models_I Q(c)$ e
    $\models_I Q(c) \to S(c, c)$.

(b) I é um modelo de Δ porque satisfaz todas as *fbfs* de Δ.

4.2.2. Considere a conceptualização (D, F, R) em que:

$D = \{\Diamond, \Box, \odot\}$
$F = \{\}$
$R = \{\ldots\}$.

Considere a interpretação I: $\{a, b, c, P, S\} \mapsto D \cup F \cup R$, tal que:

$I(a) = \Diamond$
$I(b) = \Box$

$I(c) = \odot$

Preencha a tabela abaixo, de forma a que a interpretação I seja um modelo do conjunto de *fbfs*

$$\Delta = \{P(c), P(a), \neg P(b), \forall x, y[S(x, y) \leftrightarrow x = a]\}.$$

$I(P)$	
$I(S)$	

Resposta:

$I(P)$	$\{(\odot), (\diamond)\}$
$I(S)$	$\{(\diamond, \diamond), (\diamond, \odot), (\diamond, \square)\}$

Capítulo 5

Lógica de Primeira Ordem (II)

5.1 Representação de conhecimento

5.1.1. Considere os seguintes predicados:

$$
\begin{gathered}
Fbf(x) = x \text{ é uma } \textit{fbf} \\
Var(x) = x \text{ é uma variável} \\
Contém(x, y) = x \text{ contém } y \\
Chã(x) = x \text{ é uma fórmula chã}
\end{gathered}
$$

Represente em Lógica de Primeira Ordem as seguintes proposições:

(a) Uma *fbf* que não contém variáveis é uma fórmula chã.

(b) a é uma *fbf* que não contém variáveis.

Resposta:

(a) $\forall x[(Fbf(x) \wedge \neg\exists y[Var(y) \wedge Contém(x, y)]) \rightarrow Chã(x)$ ou
$\forall x[(Fbf(x) \wedge \forall y[Var(y) \rightarrow \neg Contém(x, y)]) \rightarrow Chã(x)$

(b) $Fbf(a) \wedge \neg\exists x[Var(x) \wedge Contém(a, x)]$ ou
$Fbf(a) \wedge \forall x[Var(x) \rightarrow \neg Contém(a, x)]$

5.1.2. Considere os seguintes predicados:

$$
\begin{gathered}
Inteiro(x) = x \text{ é um número inteiro} \\
Natural(x) = x \text{ é um número natural} \\
Par(x) = x \text{ é um número par} \\
Ímpar(x) = x \text{ é um número ímpar} \\
Maior(x, y) = x \text{ é maior que } y
\end{gathered}
$$

$$Suc(x, y) = \text{o sucessor de } x \text{ é } y$$
$$Igual(x, y) = x \text{ é igual a } y$$

Represente em Lógica de Primeira Ordem as seguintes proposições:

(a) Todos os naturais são inteiros.

(b) Nem todos os inteiros são naturais.

(c) O sucessor de qualquer inteiro é maior do que esse inteiro.

(d) Todos os inteiros têm um sucessor, que também é inteiro.

(e) Para qualquer inteiro par, existe um inteiro ímpar maior do que esse número par.

(f) Não existe nenhum inteiro que seja maior que todos os inteiros.

(g) O sucessor de qualquer inteiro par é um inteiro ímpar.

(h) Zero é o único inteiro que não é natural, cujo sucessor é um natural.

Resposta:

(a) $\forall x[Natural(x) \rightarrow Inteiro(x)]$

(b) $\exists x[Inteiro(x) \wedge \neg Natural(x)]$

(c) $\forall x, y[(Inteiro(x) \wedge Suc(x, y)) \rightarrow Maior(y, x)$

(d) $\forall x[Inteiro(x) \rightarrow \exists y[Suc(x, y) \wedge Inteiro(x)]]$

(e) $\forall x[(Inteiro(x) \wedge Par(x)) \rightarrow \exists y[Inteiro(y) \wedge \acute{I}mpar(y) \wedge Maior(y, x)]]$

(f) $\neg\exists x[Inteiro(x) \wedge \forall y[Inteiro(y) \rightarrow Maior(x, y)]]$

(g) $\forall x, y[(Inteiro(x) \wedge Par(x) \wedge Suc(x, y)) \rightarrow (Inteiro(y) \wedge \acute{I}mpar(y))]$

(h) $\forall x, y[(Inteiro(x) \wedge \neg Natural(x) \wedge Suc(x, y) \wedge Natural(y)) \rightarrow Igual(x, 0)]$

5.1.3. Considere os seguintes predicados:

$$AP(x) = x \text{ é uma aula prática}$$

$$Al(x) = x \text{ é um aluno}$$

$$Prob(x, y) = x \text{ é um problema da aula prática } y$$

$$Res(x, y) = x \text{ resolveu } y$$

(a) Represente em Lógica de Primeira Ordem a seguinte proposição: *Em qualquer aula prática, dado qualquer problema dessa aula, existe pelo menos um aluno que o resolveu.*

(b) Diga qual o significado da *fbf*
$\forall x[AP(x) \rightarrow \exists y[Al(y) \wedge \forall z[Prob(z,x) \rightarrow \neg Res(y,z)]]]$.

Resposta:

(a) $\forall x, y[(AP(x) \wedge Prob(y,x)) \rightarrow \exists z[Al(z) \wedge Res(z,y)]]$

(b) Em qualquer aula prática, existe pelo menos um aluno que não resolveu nenhum problema dessa aula.

5.1.4. Considere os seguintes predicados e função:

$$\acute{E}\_m\tilde{a}e(x) = x \text{ é mãe}$$
$$M\tilde{a}e(x,y) = x \text{ é mãe de } y$$
$$m\tilde{a}e(x) = \text{a mãe de x}$$

Represente em Lógica de Primeira Ordem as relações entre:

(a) O predicado $M\tilde{a}e$ e a função $m\tilde{a}e$.

(b) O predicado $\acute{E}\_m\tilde{a}e$ e a função $m\tilde{a}e$.

(c) Os predicados $\acute{E}\_m\tilde{a}e$ e $M\tilde{a}e$.

Resposta:

(a) $\forall x[M\tilde{a}e(m\tilde{a}e(x),x)]$

(b) $\forall x[\acute{E}\_m\tilde{a}e(m\tilde{a}e(x))]$

(c) $\forall x[\acute{E}\_m\tilde{a}e(x) \leftrightarrow \exists y[M\tilde{a}e(x,y)]]$

5.1.5. Represente em Lógica de Primeira Ordem as seguintes proposições. Para cada proposição use os predicados que achar conveniente, definindo cada um deles. Não use funções.

(a) Todas as pessoas gostam de alguém.

(b) Os pacifistas condenam todas as guerras.

(c) Ser mãe significa que se tem pelo menos um filho ou uma filha.

(d) Dadas duas pessoas quaisquer pode não existir uma terceira pessoa de quem ambas gostam.

(e) Em todas as empresas existe um funcionário que tem um filho gosta de um desporto.

Resposta:

Predicados usados:

$$\begin{aligned}
Pessoa(x) &= x \text{ é uma pessoa}\\
Gosta(x,y) &= x \text{ gosta de } y\\
Pacifista(x) &= x \text{ é pacifista}\\
Guerra(x) &= x \text{ é uma guerra}\\
Condena(x,y) &= x \text{ condena } y\\
M\tilde{a}e(x) &= x \text{ é mãe}\\
Filho(x,y) &= x \text{ é filho de } y\\
Filha(x,y) &= x \text{ é filha de } y\\
Empresa(x) &= x \text{ é uma empresa}\\
Funcion\acute{a}rio(x,y) &= x \text{ é funcionário de } y\\
Desporto(x) &= x \text{ é um desporto}\\
Dif(x,y) &= x \text{ é diferente de } y
\end{aligned}$$

(a) $\forall x[Pessoa(x) \to \exists y[Pessoa(y) \land Gosta(x,y)]]$

(b) $\forall x[Pacifista(x) \to \forall y[Guerra(y) \to Condena(x,y)]]$, ou
$\forall x,y[(Pacifista(x) \land Guerra(y)) \to Condena(x,y)]$

(c) $\forall x[M\tilde{a}e(x) \leftrightarrow \exists y[Filho(y,x) \lor Filha(y,x)]]$

(d) $\exists x,y[Pessoa(x) \land Pessoa(y) \land Dif(x,y) \land \neg\exists z[Pessoa(z) \land Dif(x,z) \land Dif(y,z) \land Gosta(x,z) \land Gosta(y,z)]]$

(e) $\forall x[Empresa(x) \to$
$\quad\exists y[Funcion\acute{a}rio(y,x) \land$
$\quad\quad\exists z[Filho(z,y) \land \exists w[Desporto(w) \land Gosta(z,w)]]]]$

5.2 Forma clausal, unificação e resolução

5.2.1. Obtenha a forma clausal das seguintes *fbfs* :

(a) $\forall x[(P(x) \land \neg\exists y[Q(y) \land R(x,y)]) \to S(x)$

(b) $P(a) \land \forall x[Q(x) \to \neg R(a,x)]$

(c) $\forall x[P(x) \to \exists y[P(y) \land Q(x,y)]]$

(d) $\forall x[P(x) \to \forall y[Q(y) \to R(x,y)]]$

(e) $\forall x,y[(P(x) \land Q(y)) \to R(x,y)]$

(f) $\forall x[P(x) \leftrightarrow \exists y[Q(y,x) \lor R(y,x)]]$

(g) $\exists x,y[P(x) \land P(y) \land \neg\exists z[P(z) \land Q(x,z) \land Q(y,z)]]$

(h) $\forall x[P(x) \to \exists y[Q(y,x) \land \exists z[R(z,y) \land \exists w[S(w) \land T(z,w)]]]]$

Resposta:

(a) Passagem à forma clausal de $\forall x[(P(x) \land \neg\exists y[Q(y) \land R(x,y)]) \to S(x)$:

- *Eliminação do símbolo* $\rightarrow$

$$\forall x[\neg(P(x) \land \neg\exists y[Q(y) \land R(x,y)]) \lor S(x)]$$

- *Redução do domínio do símbolo* $\neg$

$$\forall x[\neg P(x) \lor \exists y[Q(y) \land R(x,y)]) \lor S(x)]$$

- *Normalização de variáveis*
  Não aplicável.
- *Eliminação dos quantificadores existenciais*

$$\forall x[\neg P(x) \lor (Q(f(x)) \land R(x, f(x))) \lor S(x)]$$

  em que f é uma função de Skolem.
- *Conversão para a forma "Prenex" normal*
  Não aplicável.
- *Eliminação da quantificação universal*

$$\neg P(x) \lor (Q(f(x)) \land R(x, f(x))) \lor S(x)$$

- *Obtenção da forma conjuntiva normal*

$$((\neg P(x) \lor Q(f(x))) \land (\neg P(x) \lor R(x, f(x)))) \lor S(x)$$
$$(\neg P(x) \lor Q(f(x)) \lor S(x)) \land (\neg P(x) \lor R(x, f(x)) \lor S(x))$$

- *Eliminação do símbolo* $\land$

$$\{\neg P(x) \lor Q(f(x)) \lor S(x), \neg P(x) \lor R(x, f(x)) \lor S(x)\}$$

- *Eliminação do símbolo* $\lor$

$$\{\{\neg P(x), Q(f(x)), S(x)\}, \{\neg P(x), R(x, f(x)), S(x)\}\}$$

(b) Passagem à forma clausal de $P(a) \land \forall x[Q(x) \rightarrow \neg R(a,x)]$:

- *Eliminação do símbolo* $\rightarrow$

$$P(a) \land \forall x[\neg Q(x) \lor \neg R(a,x)]$$

- *Redução do domínio do símbolo* $\neg$
  Não aplicável.
- *Normalização de variáveis*
  Não aplicável.
- *Eliminação dos quantificadores existenciais*
  Não aplicável.
- *Conversão para a forma "Prenex" normal*

$$\forall x[P(a) \land (\neg Q(x) \lor \neg R(a,x))]$$

- *Eliminação da quantificação universal*

$$P(a) \wedge (\neg Q(x) \vee \neg R(a,x))$$

- *Obtenção da forma conjuntiva normal*
  Não aplicável.
- *Eliminação do símbolo* $\wedge$

$$\{P(a), \neg Q(x) \vee \neg R(a,x)\}$$

- *Eliminação do símbolo* $\vee$

$$\{\{P(a)\}, \{\neg Q(x), \neg R(a,x)\}\}$$

(c) Passagem à forma clausal de $\forall x[P(x) \rightarrow \exists y[P(y) \wedge Q(x,y)]]$:

- *Eliminação do símbolo* $\rightarrow$

$$\forall x[\neg P(x) \vee \exists y[P(y) \wedge Q(x,y)]]$$

- *Redução do domínio do símbolo* $\neg$
  Não aplicável.
- *Normalização de variáveis*
  Não aplicável.
- *Eliminação dos quantificadores existenciais*

$$\forall x[\neg P(x) \vee (P(f(y)) \wedge Q(x, f(y))]$$

  em que $f(x)$ é uma função de Skolem.
- *Conversão para a forma "Prenex" normal*
  Não aplicável.
- *Eliminação da quantificação universal*

$$\neg P(x) \vee (P(f(y)) \wedge Q(x, f(y))$$

- *Obtenção da forma conjuntiva normal*

$$(\neg P(x) \vee P(f(y))) \wedge (\neg P(x) \vee Q(x, f(y)))$$

- *Eliminação do símbolo* $\wedge$

$$\{\neg P(x) \vee P(f(y)), \neg P(x) \vee Q(x, f(y))\}$$

- *Eliminação do símbolo* $\vee$

$$\{\{\neg P(x), P(f(y))\}, \{\neg P(x), Q(x, f(y))\}\}$$

(d) Passagem à forma clausal de $\forall x[P(x) \rightarrow \forall y[Q(y) \rightarrow R(x,y)]]$:

- *Eliminação do símbolo* $\rightarrow$

$$\forall x[\neg P(x) \vee \forall y[Q(y) \rightarrow R(x,y)]]$$

$$\forall x[\neg P(x) \vee \forall y[\neg Q(y) \vee R(x,y)]]$$

- *Redução do domínio do símbolo* $\neg$
  Não aplicável.
- *Normalização de variáveis*
  Não aplicável.
- *Eliminação dos quantificadores existenciais*
  Não aplicável.
- *Conversão para a forma "Prenex" normal*

$$\forall x, y[\neg P(x) \vee (\neg Q(y) \vee R(x,y))]$$

- *Eliminação da quantificação universal*

$$\neg P(x) \vee (\neg Q(y) \vee R(x,y))$$

- *Obtenção da forma conjuntiva normal*
  Não aplicável.
- *Eliminação do símbolo* $\wedge$

$$\{\neg P(x) \vee \neg Q(y) \vee R(x,y)\}$$

- *Eliminação do símbolo* $\vee$

$$\{\{\neg P(x), \neg Q(y), R(x,y)\}\}$$

(e) Passagem à forma clausal de $\forall x, y[(P(x) \wedge Q(y)) \rightarrow R(x,y)]$:

- *Eliminação do símbolo* $\rightarrow$

$$\forall x, y[\neg(P(x) \wedge Q(y)) \vee R(x,y)]$$

- *Redução do domínio do símbolo* $\neg$

$$\forall x, y[(\neg P(x) \vee \neg Q(y)) \vee R(x,y)]$$

- *Normalização de variáveis*
  Não aplicável.
- *Eliminação dos quantificadores existenciais*
  Não aplicável.
- *Conversão para a forma "Prenex" normal*
  Não aplicável.
- *Eliminação da quantificação universal*

$$\neg P(x) \vee \neg Q(y) \vee R(x,y)$$

- *Obtenção da forma conjuntiva normal*
  Não aplicável.
- *Eliminação do símbolo* $\wedge$

$$\{\neg P(x) \vee \neg Q(y) \vee R(x,y)\}$$

- *Eliminação do símbolo* $\vee$

$$\{\{\neg P(x), \neg Q(y), R(x,y)\}\}$$

(f) Passagem à forma clausal de $\forall x[P(x) \leftrightarrow \exists y[Q(y,x) \vee R(y,x)]]$.
Pela definição da equivalência, $\leftrightarrow$, esta *fbf* é equivalente à *fbf*
$\forall x[(P(x) \rightarrow \exists y[Q(y,x) \vee R(y,x)]) \wedge (\exists y[Q(y,x) \vee R(y,x)] \rightarrow P(x))]$.

- *Eliminação do símbolo* $\rightarrow$

$$\forall x[(\neg P(x) \vee \exists y[Q(y,x) \vee R(y,x)]) \wedge (\neg \exists y[Q(y,x) \vee R(y,x)] \vee P(x))]$$

- *Redução do domínio do símbolo* $\neg$

$$\forall x[(\neg P(x) \vee \exists y[Q(y,x) \vee R(y,x)]) \wedge (\forall y[\neg(Q(y,x) \vee R(y,x))] \vee P(x))]$$

$$\forall x[(\neg P(x) \vee \exists y[Q(y,x) \vee R(y,x)]) \wedge (\forall y[\neg Q(y,x) \wedge \neg R(y,x)] \vee P(x))]$$

- *Normalização de variáveis*

$$\forall x[(\neg P(x) \vee \exists y[Q(y,x) \vee R(y,x)]) \wedge (\forall z[\neg Q(z,x) \wedge \neg R(z,x)] \vee P(x))]$$

- *Eliminação dos quantificadores existenciais*

$$\forall x[(\neg P(x) \vee (Q(f(x),x) \vee R(f(x),x))) \wedge (\forall z[\neg Q(z,x) \wedge \neg R(z,x)] \vee P(x))]$$

  em que $f(x)$ é uma função de Skolem.
- *Conversão para a forma "Prenex" normal*

$$\forall x,z[(\neg P(x) \vee Q(f(x),x) \vee R(f(x),x)) \wedge ((\neg Q(z,x) \wedge \neg R(z,x)) \vee P(x))]$$

- *Eliminação da quantificação universal*

$$(\neg P(x) \vee Q(f(x),x) \vee R(f(x),x)) \wedge ((\neg Q(z,x) \wedge \neg R(z,x)) \vee P(x))$$

- *Obtenção da forma conjuntiva normal*

$$(\neg P(x) \vee Q(f(x),x) \vee R(f(x),x)) \wedge (\neg Q(z,x) \vee P(x)) \wedge (\neg R(z,x) \vee P(x))$$

- *Eliminação do símbolo* $\wedge$

$$\{\neg P(x) \vee Q(f(x),x) \vee R(f(x),x), \neg Q(z,x) \vee P(x), \neg R(z,x) \vee P(x)\}$$

- *Eliminação do símbolo* $\vee$

$$\{\{\neg P(x), Q(f(x),x), R(f(x),x)\}, \{\neg Q(z,x), P(x)\}, \{\neg R(z,x), P(x)\}\}$$

(g) Passagem à forma clausal de
$\exists x, y[P(x) \wedge P(y) \wedge \neg\exists z[P(z) \wedge Q(x, z) \wedge Q(y, z)]]$:

- *Eliminação do símbolo* $\rightarrow$
  Não aplicável.
- *Redução do domínio do símbolo* $\neg$

$$\exists x, y[P(x) \wedge P(y) \wedge \forall z[\neg(P(z) \wedge Q(x, z) \wedge Q(y, z))]]$$

$$\exists x, y[P(x) \wedge P(y) \wedge \forall z[\neg P(z) \vee \neg Q(x, z) \vee \neg Q(y, z)]]$$

- *Normalização de variáveis*
  Não aplicável.
- *Eliminação dos quantificadores existenciais*

$$P(a) \wedge P(b) \wedge \forall z[\neg P(z) \vee \neg Q(a, z) \vee \neg Q(b, z)]$$

  em que a e b são constantes de Skolem.
- *Conversão para a forma "Prenex" normal*

$$\forall z[P(a) \wedge P(b) \wedge (\neg P(z) \vee \neg Q(a, z) \vee \neg Q(b, z))]$$

- *Eliminação da quantificação universal*

$$P(a) \wedge P(b) \wedge (\neg P(z) \vee \neg Q(a, z) \vee \neg Q(b, z))$$

- *Obtenção da forma conjuntiva normal*
  Não aplicável.
- *Eliminação do símbolo* $\wedge$

$$\{P(a), P(b), \neg P(z) \vee \neg Q(a, z) \vee \neg Q(b, z)\}$$

- *Eliminação do símbolo* $\vee$

$$\{\{P(a)\}, \{P(b)\}, \{\neg P(z), \neg Q(a, z), \neg Q(b, z)\}\}$$

(h) Passagem à forma clausal de
$\forall x[P(x) \rightarrow \exists y[Q(y, x) \wedge \exists z[R(z, y) \wedge \exists w[S(w) \wedge T(z, w)]]]]$:

- *Eliminação do símbolo* $\rightarrow$

$$\forall x[\neg P(x) \vee \exists y[Q(y, x) \wedge \exists z[R(z, y) \wedge \exists w[S(w) \wedge T(z, w)]]]]$$

- *Redução do domínio do símbolo* $\neg$
  Não aplicável.
- *Normalização de variáveis*
  Não aplicável.

- *Eliminação dos quantificadores existenciais*

$$\forall x[\neg P(x) \vee (Q(f(x), x) \wedge \exists z[R(z, f(x)) \wedge \exists w[S(w) \wedge T(z, w)]])]$$
$$\forall x[\neg P(x) \vee (Q(f(x), x) \wedge (R(g(x), f(x)) \wedge \exists w[S(w) \wedge T(g(x), w)]))]$$
$$\forall x[\neg P(x) \vee (Q(f(x), x) \wedge (R(g(x), f(x)) \wedge (S(h(x)) \wedge T(g(x), h(x)))))]$$

  em que $f(x)$, $g(x)$ e $h(x)$ são funções de Skolem.
- *Conversão para a forma "Prenex" normal*
  Não aplicável.
- *Eliminação da quantificação universal*

$$\neg P(x) \vee (Q(f(x), x) \wedge R(g(x), f(x)) \wedge S(h(x)) \wedge T(g(x), h(x)))$$

- *Obtenção da forma conjuntiva normal*
  $(\neg P(x) \vee Q(f(x), x)) \wedge (\neg P(x) \vee R(g(x), f(x))) \wedge$
  $(\neg P(x) \vee S(h(x))) \wedge (\neg P(x) \vee T(g(x), h(x)))$
- *Eliminação do símbolo* $\wedge$
  $\{\neg P(x) \vee Q(f(x), x), \neg P(x) \vee R(g(x), f(x)),$
  $\neg P(x) \vee S(h(x)), \neg P(x) \vee T(g(x), h(x))\}$
- *Eliminação do símbolo* $\vee$
  $\{\{\neg P(x), Q(f(x), x)\}, \{\neg P(x), R(g(x), f(x))\},$
  $\{\neg P(x), S(h(x))\}, \{\neg P(x), T(g(x), h(x))\}\}$

5.2.2. Utilize o algoritmo de unificação para determinar se os seguintes conjuntos de *fbfs* são unificáveis, e, no caso de o serem, determine o unificador mais geral. Mostre todos os passos intermédios usados nos cálculos. Considere que x, y, z e w são variáveis.

(a) $\{P(a, f(a)), P(w, f(z))\}$
(b) $\{P(a, f(a)), P(w, f(f(z)))\}$
(c) $\{P(a, f(f(a))), P(w, f(z))\}$
(d) $\{P(g(z), z), P(w, f(a))\}$
(e) $\{P(g(z), f(x, z), x, z), P(w, y, a, w)\}$

Resposta:

(a)

Conjunto de *fbfs*	Conj. desac.	Subst.
$\{P(a, f(a)), P(w, f(z))\}$	$\{a, w\}$	$\{a/w\}$
$\{P(a, f(a)), P(a, f(z))\}$	$\{a, z\}$	$\{a/z\}$
$\{P(a, f(a))\}$		

O conjunto é unificável, e o unificador mais geral é $\{a/w\} \circ \{a/z\} = \{a/w, a/z\}$.

(b)

Conjunto de *fbfs*	Conj. desac.	Subst.
$\{P(a, f(a)), P(w, f(f(z)))\}$	$\{a, w\}$	$\{a/w\}$
$\{P(a, f(a)), P(a, f(f(z)))\}$	$\{a, f(z)\}$	–

O conjunto não é unificável, pois no último conjunto de desacordo, $\{a, f(z)\}$, não existe nenhuma variável.

(c)

Conjunto de *fbfs*	Conj. desac.	Subst.
$\{P(a, f(f(a))), P(w, f(z))\}$	$\{a, w\}$	$\{a/w\}$
$\{P(a, f(f(a))), P(a, f(z))\}$	$\{f(a), z\}$	$\{f(a)/z\}$
$\{P(a, f(f(a)))\}$		

O conjunto é unificável, e o unificador mais geral é $\{a/w\} \circ \{f(a)/z\} = \{a/w, f(a)/z\}$.

(d)

Conjunto de *fbfs*	Conj. desac.	Subst.
$\{P(g(z), z), P(w, f(a))\}$	$\{g(z), w\}$	$\{g(z)/w\}$
$\{P(g(z), z), P(g(z), f(a))\}$	$\{f(a), z\}$	$\{f(a)/z\}$
$\{P(g(f(a)), f(a))\}$		

O conjunto é unificável, e o unificador mais geral é $\{g(z)/w\} \circ \{f(a)/z\} = \{g(f(a))/w, f(a)/z\}$.

(e)

Conjunto de *fbfs*	Conj. desac.	Subst.
$\{P(g(z), f(x, z), x, z), P(w, y, a, w)\}$	$\{g(z), w\}$	$\{g(z)/w\}$
$\{P(g(z), f(x, z), x, z), P(g(z), y, a, g(z))\}$	$\{f(x, z), y\}$	$\{f(x, z)/y\}$
$\{P(g(z), f(x, z), x, z), P(g(z), f(x, z), a, g(z))\}$	$\{x, a\}$	$\{a/x\}$
$\{P(g(z), f(a, z), a, z), P(g(z), f(a, z), a, g(z))\}$	$\{z, g(z)\}$	—

O conjunto não é unificável, pois no último conjunto de desacordo, $\{z, g(z)\}$, não existem uma variável e um termo que não mencione a variável.

5.2.3. Usando resolução, demonstre o seguinte argumento

$$\{\{\neg P(x), Q(f(x)), S(x)\}, \{\neg P(x), R(x, f(x)), S(x)\}, \{P(a)\}, \{\neg Q(x), \neg R(a, x)\}\} \vdash \{S(a)\}$$

Resposta:

1	$\{\neg P(x), Q(f(x)), S(x)\}$	Prem
2	$\{\neg P(x), R(x, f(x)), S(x)\}$	Prem
3	$\{P(a)\}$	Prem
4	$\{\neg Q(x), \neg R(a, x)\}$	Prem
5	$\{Q(f(a)), S(a)\}$	Res, (1,3), $\{a/x\}$
6	$\{R(a, f(a)), S(a)\}$	Res, (2,3), $\{a/x\}$
7	$\{\neg R(a, f(a)), S(a)\}$	Res, (4,5), $\{f(a)/x\}$
8	$\{S(a)\}$	Res, (6,7), $\{\}$

5.2.4. Demonstre os seguintes teoremas usando resolução.

(a) $(\exists x[P(x)] \wedge \forall x[P(x) \rightarrow Q(x)]) \rightarrow \exists x[Q(x)]$.

(b) $\forall x[\neg P(x)] \rightarrow \forall x[P(x) \rightarrow Q(x)]$.

(c) $(\forall x[P(x) \rightarrow Q(x)] \wedge \forall x[P(x) \rightarrow \neg Q(x)]) \rightarrow \forall x[\neg P(x)]$.

(d) $(\forall x[P(x) \rightarrow R(x)] \wedge \forall x[Q(x) \rightarrow R(x)] \wedge (P(a) \vee Q(a))) \rightarrow R(a)$.

(e) $\forall x[P(x) \vee \neg P(x)]$.

(f) $(\forall x[P(x)] \wedge \forall x[Q(x)]) \rightarrow \forall x[P(x) \wedge Q(x)]$.

(g) $\forall x[Q(x)] \rightarrow \forall x[P(x) \rightarrow Q(x)])$.

Resposta:

Para provar os teoremas dados, faremos provas por refutação a partir da negação de cada um dos teoremas. Serão omitidos todos os passos da passagem à forma clausal que não forem aplicáveis.

(a) Negação do teorema:

$$\neg((\exists x[P(x)] \wedge \forall x[P(x) \rightarrow Q(x)]) \rightarrow \exists x[Q(x)]).$$

- Passagem à forma clausal:
  - *Eliminação do símbolo* $\rightarrow$

$$\neg(\neg(\exists x[P(x)] \wedge \forall x[\neg P(x) \vee Q(x)]) \vee \exists x[Q(x)])$$

  - *Redução do domínio do símbolo* $\neg$

$$(\exists x[P(x)] \wedge \forall x[\neg P(x) \vee Q(x)]) \wedge \neg\exists x[Q(x)]$$

$$(\exists x[P(x)] \wedge \forall x[\neg P(x) \vee Q(x)]) \wedge \forall x[\neg Q(x)]$$

  - *Normalização de variáveis*

$$(\exists x[P(x)] \wedge \forall y[\neg P(y) \vee Q(y)]) \wedge \forall z[\neg Q(z)]$$

  - *Eliminação dos quantificadores existenciais*

$$(P(a) \wedge \forall y[\neg P(y) \vee Q(y)]) \wedge \forall z[\neg Q(z)]$$

    em que a é uma constante de Skolem.
  - *Conversão para a forma "Prenex" normal*

$$\forall y \forall z[(P(a) \wedge (\neg P(y) \vee Q(y))) \wedge \neg Q(z)]$$

  - *Eliminação da quantificação universal*

$$P(a) \wedge (\neg P(y) \vee Q(y)) \wedge \neg Q(z)$$

  - *Eliminação do símbolo* $\wedge$

$$\{P(a), \neg P(y) \vee Q(y), \neg Q(z)\}$$

  - *Eliminação do símbolo* $\vee$

$$\{\{P(a)\}, \{\neg P(y), Q(y)\}, \{\neg Q(z)\}\}$$

- Prova:

1	$\{P(a)\}$	Prem
2	$\{\neg P(y), Q(y)\}$	Prem
3	$\{\neg Q(z)\}$	Prem
4	$\{Q(a)\}$	Res, (1,2), $\{a/y\}$
5	$\{\}$	Res, (3,4), $\{a/z\}$

(b) Negação do teorema:

$$\neg(\forall x[\neg P(x)] \rightarrow \forall x[P(x) \rightarrow Q(x)]).$$

- Passagem à forma clausal:
  - *Eliminação do símbolo* $\rightarrow$

$$\neg(\neg\forall x[\neg P(x)] \vee \forall x[P(x) \rightarrow Q(x)])$$

$$\neg(\neg\forall x[\neg P(x)] \vee \forall x[\neg P(x) \vee Q(x)])$$

  - *Redução do domínio do símbolo* $\neg$

$$\forall x[\neg P(x)] \wedge \neg\forall x[\neg P(x) \vee Q(x)]$$

$$\forall x[\neg P(x)] \wedge \exists x[\neg(\neg P(x) \vee Q(x)])$$

$$\forall x[\neg P(x)] \wedge \exists x[P(x) \wedge \neg Q(x)])$$

  - *Normalização de variáveis*

$$\forall x[\neg P(x)] \wedge \exists y[P(y) \wedge \neg Q(y)])$$

  - *Eliminação dos quantificadores existenciais*

$$\forall x[\neg P(x)] \wedge (P(a) \wedge \neg Q(a))$$

    em que a é uma constante de Skolem.
  - *Eliminação da quantificação universal*

$$\neg P(x) \wedge P(a) \wedge \neg Q(a)$$

  - *Eliminação dos símbolos* $\wedge$ *e* $\vee$

$$\{\{\neg P(x)\}, \{P(a)\}, \{\neg Q(a)\}\}$$

- Prova:

1	$\{\neg P(x)\}$	Prem
2	$\{P(a)\}$	Prem
3	$\{\}$	Res, (1,2), $\{a/x\}$

(c) Negação do teorema:

$$\neg((\forall x[P(x) \to Q(x)] \land \forall x[P(x) \to \neg Q(x)]) \to \forall x[\neg P(x)]).$$

- Passagem à forma clausal:
  - *Eliminação do símbolo* $\to$

$$\neg(\neg(\forall x[\neg P(x) \lor Q(x)] \land \forall x[\neg P(x) \lor \neg Q(x)]) \lor \forall x[\neg P(x)])$$

  - *Redução do domínio do símbolo* $\neg$

$$(\forall x[\neg P(x) \lor Q(x)] \land \forall x[\neg P(x) \lor \neg Q(x)]) \land \neg\forall x[\neg P(x)]$$

$$\forall x[\neg P(x) \lor Q(x)] \land \forall x[\neg P(x) \lor \neg Q(x)] \land \exists x[P(x)]$$

  - *Normalização de variáveis*

$$\forall x[\neg P(x) \lor Q(x)] \land \forall y[\neg P(y) \lor \neg Q(y)] \land \exists z[P(z)]$$

  - *Eliminação dos quantificadores existenciais*

$$\forall x[\neg P(x) \lor Q(x)] \land \forall y[\neg P(y) \lor \neg Q(y)] \land P(a)$$

    em que a é uma constante de Skolem.
  - *Conversão para a forma "Prenex" normal*

$$\forall x, y[(\neg P(x) \lor Q(x)) \land (\neg P(y) \lor \neg Q(y)) \land P(a)]$$

  - *Eliminação da quantificação universal*

$$(\neg P(x) \lor Q(x)) \land (\neg P(y) \lor \neg Q(y)) \land P(a)$$

  - *Eliminação dos símbolos* $\land$ *e* $\lor$
    $\{\{\neg P(x), Q(x))\}, \{\neg P(y), \neg Q(y)\}, \{P(a)\}\}$
- Prova:

1	$\{\neg P(x), Q(x))\}$	Prem
2	$\{\neg P(y), \neg Q(y)\}$	Prem
3	$\{P(a)\}$	Prem
4	$\{Q(a)\}$	Res, (1,3), $\{a/x\}$
5	$\{\neg P(a)\}$	Res, (2,4), $\{a/y\}$
6	$\{\}$	Res, (3,5), $\{\}$

(d) Negação do teorema:

$$\neg((\forall x[P(x) \to R(x)] \wedge \forall x[Q(x) \to R(x)] \wedge (P(a) \vee Q(a))) \to R(a)).$$

- Passagem à forma clausal:
  - *Eliminação do símbolo* $\to$

    $$\neg(\neg(\forall x[\neg P(x) \vee R(x)] \wedge \forall x[\neg Q(x) \vee R(x)] \wedge (P(a) \vee Q(a))) \vee R(a))$$
  - *Redução do domínio do símbolo* $\neg$

    $$(\forall x[\neg P(x) \vee R(x)] \wedge \forall x[\neg Q(x) \vee R(x)] \wedge (P(a) \vee Q(a))) \wedge \neg R(a)$$
  - *Normalização de variáveis*

    $$\forall x[\neg P(x) \vee R(x)] \wedge \forall y[\neg Q(y) \vee R(y)] \wedge (P(a) \vee Q(a)) \wedge \neg R(a)$$
  - *Conversão para a forma "Prenex" normal*

    $$\forall x, y[(\neg P(x) \vee R(x)) \wedge (\neg Q(y) \vee R(y)) \wedge (P(a) \vee Q(a)) \wedge \neg R(a)]$$
  - *Eliminação da quantificação universal*

    $$(\neg P(x) \vee R(x)) \wedge (\neg Q(y) \vee R(y)) \wedge (P(a) \vee Q(a)) \wedge \neg R(a)$$
  - *Eliminação dos símbolos* $\wedge$ *e* $\vee$
    $\{\{\neg P(x), R(x)\}, \{\neg Q(y), R(y)\}, \{P(a), Q(a)\}, \{\neg R(a)\}\}$
- Prova:

1	$\{\neg P(x), R(x))\}$	Prem
2	$\{\neg Q(y), R(y)\}$	Prem
3	$\{P(a), Q(a)\}$	Prem
4	$\{\neg R(a)\}$	Prem
5	$\{\neg P(a)\}$	Res, (1,4), $\{a/x\}$
6	$\{\neg Q(a)\}$	Res, (2,4), $\{a/y\}$
7	$\{Q(a)\}$	Res, (3,5), $\{\}$
8	$\{\}$	Res, (6,7), $\{\}$

(e) Negação do teorema:

$$\neg\forall x[P(x) \vee \neg P(x)].$$

- Passagem à forma clausal:
  - *Redução do domínio do símbolo* $\neg$

    $$\exists x[\neg(P(x) \vee \neg P(x))]$$

    $$\exists x[\neg P(x) \wedge P(x))]$$

– *Eliminação dos quantificadores existenciais*

$$\neg P(a) \wedge P(a)$$

– *Eliminação dos símbolos* $\wedge$ *e* $\vee$
$\{\{\neg P(a)\}, \{P(a)\}\}$

- Prova:

1	$\{\neg P(a)\}$	Prem
2	$\{P(a)\}$	Prem
3	$\{\}$	Res, (1,2), $\{\}$

(f) Negação do teorema:

$$\neg((\forall x[P(x)] \wedge \forall x[Q(x)]) \rightarrow \forall x[P(x) \wedge Q(x)]).$$

- Passagem à forma clausal:
  – *Eliminação do símbolo* $\rightarrow$

$$\neg(\neg(\forall x[P(x)] \wedge \forall x[Q(x)]) \vee \forall x[P(x) \wedge Q(x)])$$

  – *Redução do domínio do símbolo* $\neg$

$$\neg\neg(\forall x[P(x)] \wedge \forall x[Q(x)]) \wedge \neg\forall x[P(x) \wedge Q(x)]$$

$$\forall x[P(x)] \wedge \forall x[Q(x)] \wedge \neg\forall x[P(x) \wedge Q(x)]$$

$$\forall x[P(x)] \wedge \forall x[Q(x)] \wedge \exists x[\neg(P(x) \wedge Q(x))]$$

$$\forall x[P(x)] \wedge \forall x[Q(x)] \wedge \exists x[\neg P(x) \vee \neg Q(x)]$$

  – *Normalização de variáveis*

$$\forall x[P(x)] \wedge \forall y[Q(y)] \wedge \exists z[\neg P(z) \vee \neg Q(z)]$$

  – *Eliminação dos quantificadores existenciais*

$$\forall x[P(x)] \wedge \forall y[Q(y)] \wedge (\neg P(a) \vee \neg Q(a))$$

  em que a é uma constante de Skolem.

  – *Conversão para a forma "Prenex" normal*

$$\forall x, y[P(x) \wedge Q(y) \wedge (\neg P(a) \vee \neg Q(a))]$$

  – *Eliminação da quantificação universal*

$$P(x) \wedge Q(y) \wedge (\neg P(a) \vee \neg Q(a))$$

  – *Eliminação do símbolo* $\wedge$

$$\{P(x), Q(y), \neg P(a) \vee \neg Q(a)\}$$

- *Eliminação do símbolo* $\vee$

$$\{\{P(x)\}, \{Q(y)\}, \{\neg P(a), \neg Q(a)\}\}$$

- Prova:

1	$\{P(x)\}$	Prem
2	$\{Q(y)\}$	Prem
3	$\{\neg P(a), \neg Q(a)\}$	Prem
4	$\{\neg Q(a)\}$	Res, (1,3), $\{a/x\}$
5	$\{\}$	Res, (2,4), $\{a/y\}$

(g) Negação do teorema:

$$\neg(\forall x[Q(x)] \to \forall x[P(x) \to Q(x)]).$$

- Passagem à forma clausal:
  - *Eliminação do símbolo* $\to$

$$\neg(\neg\forall x[Q(x)] \vee \forall x[P(x) \to Q(x)])$$

$$\neg(\neg\forall x[Q(x)] \vee \forall x[\neg P(x) \vee Q(x)])$$

  - *Redução do domínio do símbolo* $\neg$

$$\forall x[Q(x)] \wedge \neg\forall x[\neg P(x) \vee Q(x)]$$

$$\forall x[Q(x)] \wedge \exists x[\neg(\neg P(x) \vee Q(x)])$$

$$\forall x[Q(x)] \wedge \exists x[P(x) \wedge \neg Q(x)])$$

  - *Normalização de variáveis*

$$\forall x[Q(x)] \wedge \exists y[P(y) \wedge \neg Q(y)])$$

  - *Eliminação dos quantificadores existenciais*

$$\forall x[Q(x)] \wedge (P(a) \wedge \neg Q(a))$$

  em que a é uma constante de Skolem.
  - *Eliminação da quantificação universal*

$$Q(x) \wedge P(a) \wedge \neg Q(a)$$

  - *Eliminação dos símbolos* $\wedge$ *e* $\vee$

$$\{\{Q(x)\}, \{P(a)\}, \{\neg Q(a)\}\}$$

- Prova:

1	$\{Q(x)\}$	Prem
2	$\{\neg Q(a)\}$	Prem
3	$\{\}$	Res, (1,2), $\{a/x\}$

5.3 O método de Herbrand

Capítulo 6

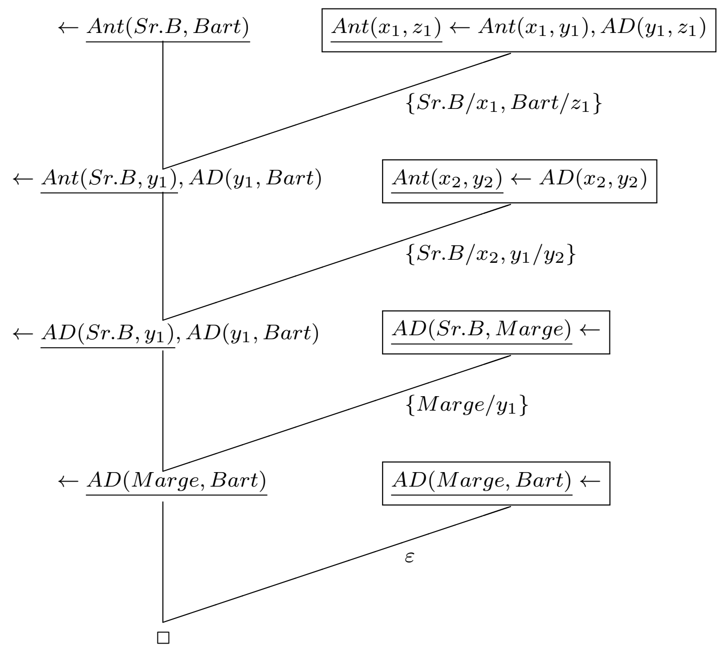
Programação em Lógica

6.1 Resolução SLD

6.1.1. Considere o seguinte programa:
$Ant(x, z) \leftarrow Ant(x, y), AD(y, z)$
$Ant(x, y) \leftarrow AD(x, y)$
$AD(Sr.B, Marge) \leftarrow$
$AD(Marge, Bart) \leftarrow$

Apresente uma prova por refutação SLD para o objetivo $\leftarrow Ant(Sr.B, Bart)$. Use a função de seleção S_1 que escolhe o primeiro literal no objetivo, isto é, $S_1(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_1$. Indique a resposta calculada.

Resposta:

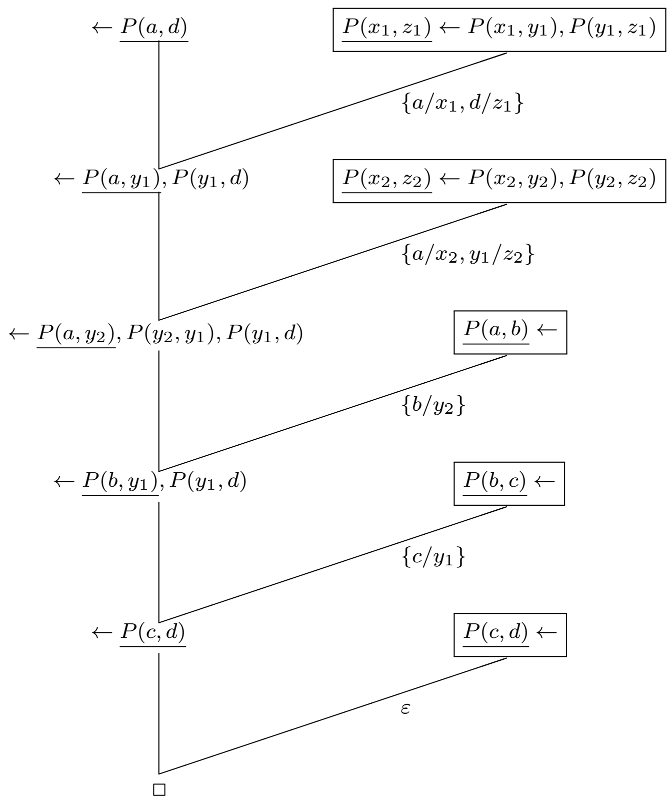


Uma vez que o objetivo a provar não tem variáveis, a resposta calculada é a substituição vazia.

6.1.2. Considere o seguinte programa:
$P(x, z) \leftarrow P(x, y), P(y, z)$
$P(a, b) \leftarrow$
$P(b, c) \leftarrow$
$P(c, d) \leftarrow$

Apresente uma prova por refutação SLD para o objetivo $\leftarrow P(a, d)$. Use a função de seleção S_1 que escolhe o primeiro literal no objetivo, isto é, $S_1(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_1$. Indique a resposta calculada.

Resposta:

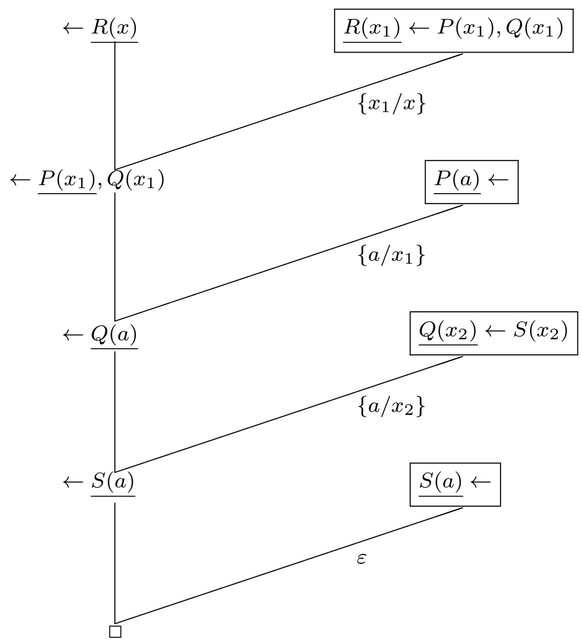


Uma vez que o objetivo a provar não tem variáveis, a resposta calculada é a substituição vazia.

6.1.3. Considere o seguinte programa:
$R(x) \leftarrow P(x), Q(x)$
$Q(x) \leftarrow S(x)$
$P(a) \leftarrow$
$S(a) \leftarrow$

Apresente uma prova por refutação SLD para o objetivo $\leftarrow R(x)$. Use a função de seleção S_1 que escolhe o primeiro literal no objetivo, isto é, $S_1(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_1$. Indique a resposta calculada.

Resposta:



Resposta calculada:

$$(\{x_1/x\} \circ \{a/x_1\} \circ \{a/x_2\} \circ \varepsilon) \mid_{\{x\}} = \{a/x, a/x_1, a/x_2\} \mid_{\{x\}} = \{a/x\}$$

6.1.4. Demonstre que

$$\{\forall x[(P(x) \wedge Q(x)) \to R(x)], \forall x[S(x) \to Q(x)], P(a), S(a)\} \vdash R(a) \wedge S(a).$$

usando resolução SLD, com as seguintes funções de seleção:

(a) $S(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_1$.

(b) $S(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_n$.

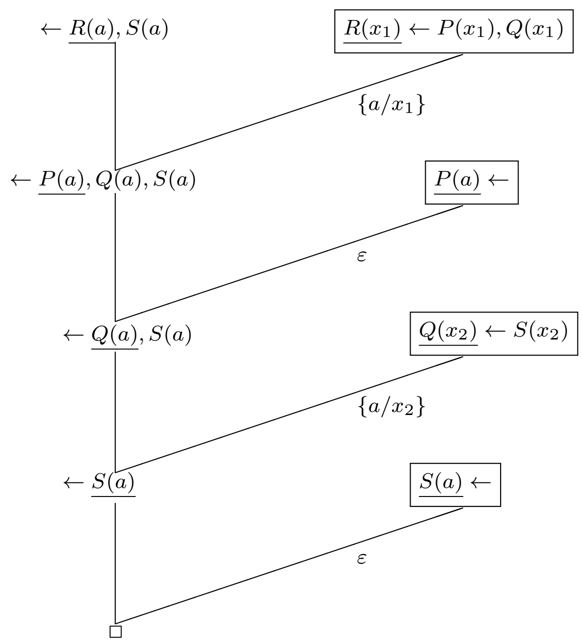
Resposta:

Cláusulas correspondentes às premissas do argumento:
$R(x) \leftarrow P(x), Q(x)$
$Q(x) \leftarrow S(x)$
$P(a) \leftarrow$

$S(a) \leftarrow$

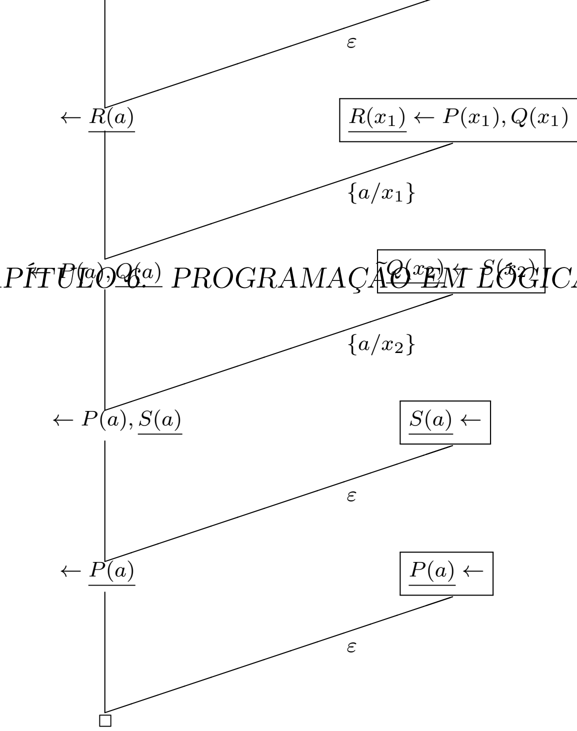
Objetivo correspondente à negação da conclusão do argumento:
$\leftarrow R(a), S(a)$

(a)



Uma vez que o objetivo a provar não tem variáveis, a resposta calculada é a substituição vazia.

(b)



Uma vez que o objetivo a provar não tem variáveis, a resposta calculada é a substituição vazia.

6.1.5. Demonstre que

$$\{\forall x[P(x) \to Q(x)], \forall x[Q(x) \to R(x)]\} \vdash \forall x[P(x) \to R(x)].$$

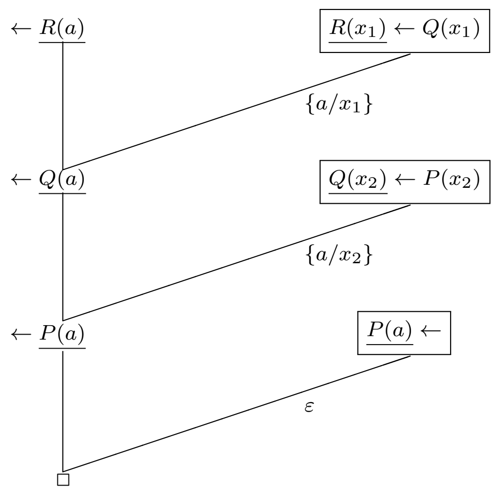
usando resolução SLD. Indique a resposta calculada.

Resposta:

Cláusulas correspondentes às premissas do argumento:
$Q(x) \leftarrow P(x)$
$R(x) \leftarrow Q(x)$

Cláusulas correspondentes à negação da conclusão do argumento:
$\neg\forall x[P(x) \to R(x)]$
$\neg\forall x[\neg P(x) \vee R(x)]$
$\exists x[\neg(\neg P(x) \vee R(x))]$
$\exists x[P(x) \wedge \neg R(x)]$
$P(a) \wedge \neg R(a)$ (em que a é uma constante de Skolem)
$P(a) \leftarrow$
$\leftarrow R(a)$

Neste caso, a negação da conclusão dá origem a uma afirmação, $P(a) \leftarrow$, e um objetivo, $\leftarrow R(a)$. A afirmação tem de ser adicionada ao programa.



Uma vez que o objetivo a provar não tem variáveis, a resposta calculada é a substituição vazia.

6.1.6. Demonstre que

$$\{\forall x[P(x) \to Q(x)], \forall x[P(x) \to \neg Q(x)]\} \vdash \neg\exists x[P(x)].$$

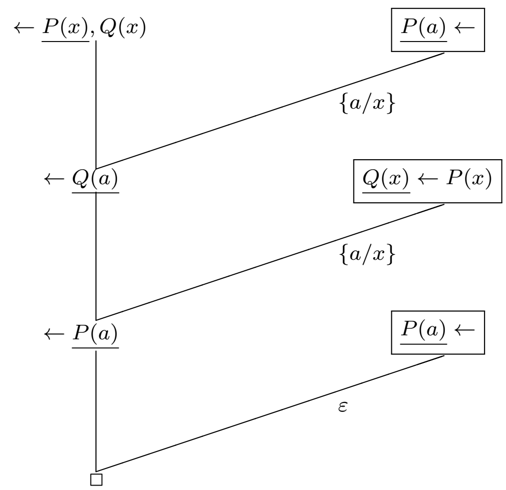
usando resolução SLD. Indique a resposta calculada.

Resposta:

Cláusulas correspondentes às premissas do argumento:
$Q(x) \leftarrow P(x)$
$\leftarrow P(x), Q(x)$

Objetivo correspondente à negação da conclusão do argumento:

$P(a) \leftarrow$



Resposta calculada:

$$(\{a/x\} \circ \{a/x\} \circ \varepsilon) \mid_{\{x\}} = \{a/x\} \mid_{\{x\}} = \{a/x\}$$

6.1.7. Demonstre que

$$\{\forall x, y, z[(Maior(x,y) \wedge Maior(y,z)) \rightarrow Maior(x,z)],$$
$$\forall x[Maior(suc(x),x)]\}$$
$$\vdash \forall x[Maior(suc(suc(x)),x)]$$

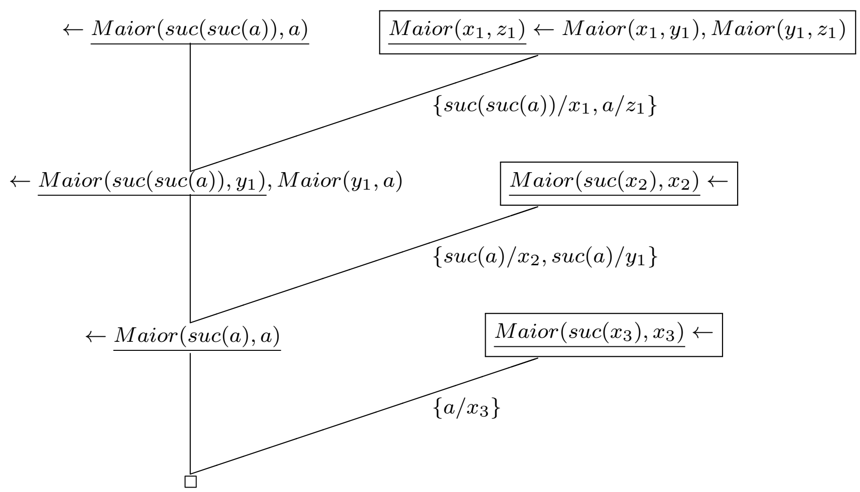
usando resolução SLD. Indique a resposta calculada.

Resposta:

Cláusulas correspondentes às premissas do argumento:
$Maior(x,z) \leftarrow Maior(x,y), Maior(y,z)$
$Maior(suc(x),x) \leftarrow$

Objetivo correspondente à negação da conclusão do argumento:

$\leftarrow Maior(suc(suc(a)), a)$



Uma vez que o objetivo a provar não tem variáveis, a resposta calculada é a substituição vazia.

6.2 Árvores SLD

6.2.1. Considere o seguinte programa:

$P(x) \leftarrow Q(x), R(x)$

$Q(a) \leftarrow$

$Q(b) \leftarrow$

$Q(c) \leftarrow$

$R(a) \leftarrow$

$R(c) \leftarrow$

Desenhe árvores SLD para calcular a(s) resposta(s) deste programa ao objetivo $\leftarrow P(x)$, usando as seguintes funções de seleção:

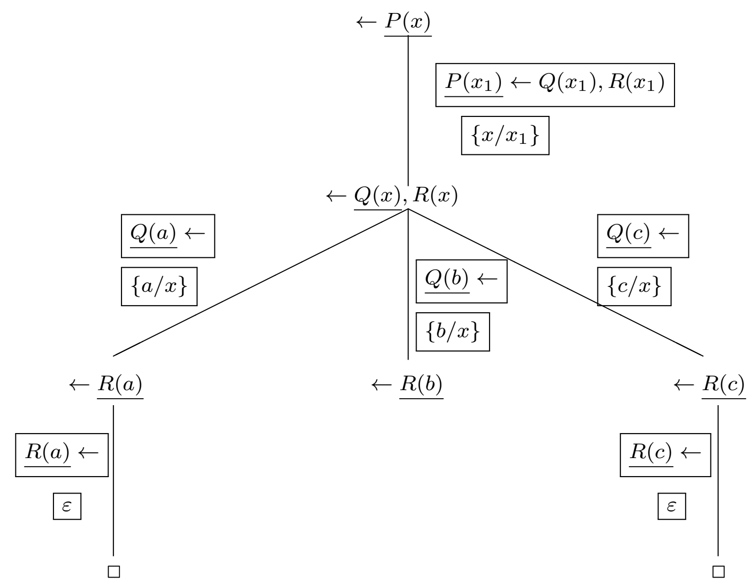
(a) $S(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_1$.

(b) $S(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_n$.

Indique a(s) resposta(s) calculada(s).

Resposta:

(a)

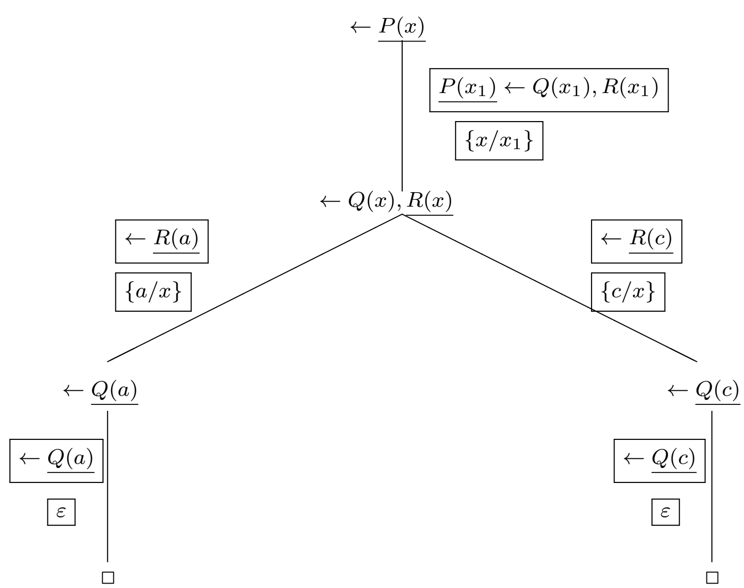


Respostas calculadas:

$$(\{x/x_1\} \circ \{a/x\} \circ \varepsilon) \mid_{\{x\}} = \{a/x_1, a/x\} \mid_{\{x\}} = \{a/x\}$$

$$(\{x/x_1\} \circ \{c/x\} \circ \varepsilon) \mid_{\{x\}} = \{c/x_1, c/x\} \mid_{\{x\}} = \{c/x\}$$

(b)



Respostas calculadas:

$$(\{x/x_1\} \circ \{a/x\} \circ \varepsilon) \mid_{\{x\}} = \{a/x_1, a/x\} \mid_{\{x\}} = \{a/x\}$$

$$(\{x/x_1\} \circ \{c/x\} \circ \varepsilon) \mid_{\{x\}} = \{c/x_1, c/x\} \mid_{\{x\}} = \{c/x\}$$

6.2.2. Considere o seguinte programa:

$R(x) \leftarrow P(x), Q(x)$
$Q(x) \leftarrow S(x)$
$P(a) \leftarrow$
$P(b) \leftarrow$
$S(a) \leftarrow$
$Q(b) \leftarrow$

Desenhe árvores SLD para calcular a(s) resposta(s) deste programa ao objetivo $\leftarrow R(x)$, usando as seguintes funções de seleção:

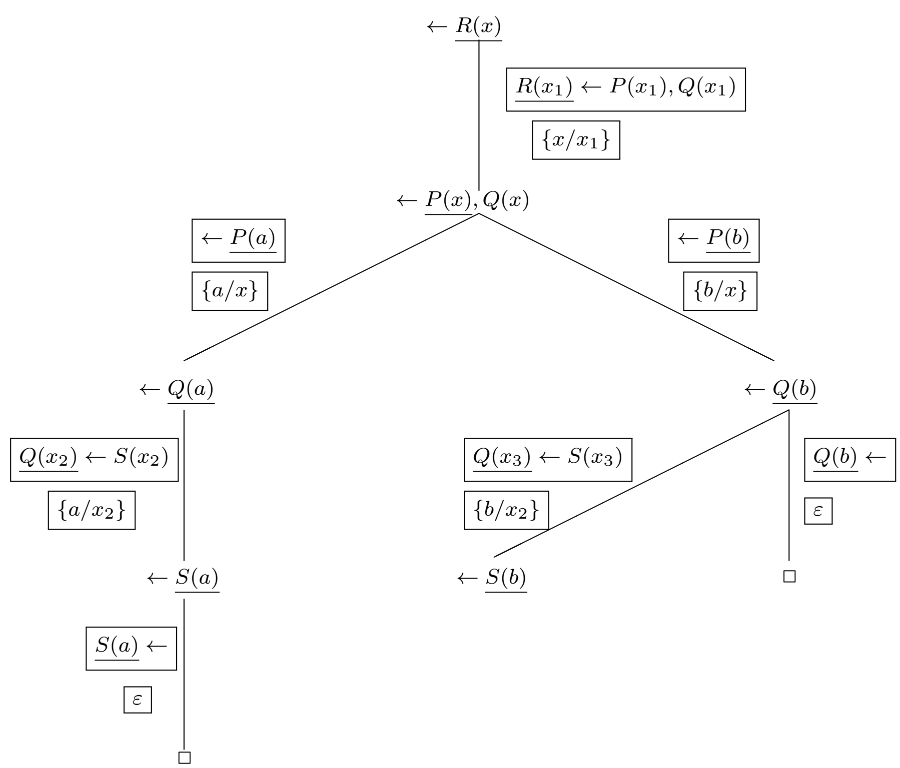
(a) $S(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_1$.

(b) $S(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_n$.

Indique a(s) resposta(s) calculada(s).

Resposta:

(a)

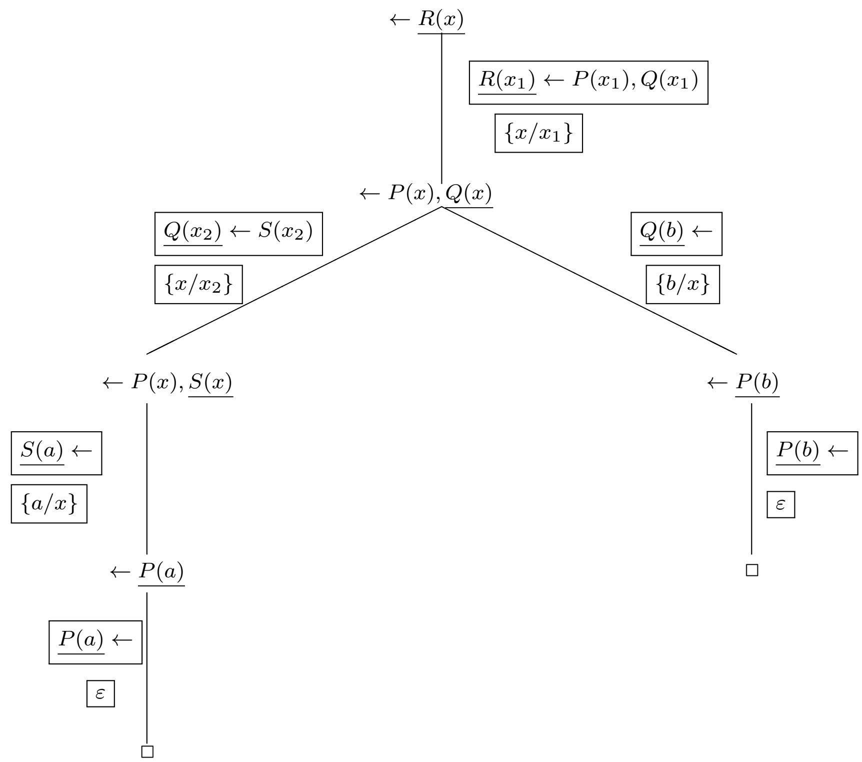


Respostas calculadas:

$$(\{x/x_1\} \circ \{a/x\} \circ \{a/x_2\} \circ \varepsilon) \mid_{\{x\}} = \{a/x_1, a/x, a/x_2\} \mid_{\{x\}} = \{a/x\}$$

$$(\{x/x_1\} \circ \{b/x\} \circ \varepsilon) \mid_{\{x\}} = \{b/x_1, b/x\} \mid_{\{x\}} = \{b/x\}$$

(b)



Respostas calculadas:

$$(\{x/x_1\} \circ \{x/x_2\} \circ \{a/x\} \circ \varepsilon) \mid_{\{x\}} = \{a/x_1, a/x_2, a/x\} \mid_{\{x\}} = \{a/x\}$$

$$(\{x/x_1\} \circ \{b/x\} \circ \varepsilon) \mid_{\{x\}} = \{b/x_1, b/x\} \mid_{\{x\}} = \{b/x\}$$

6.2.3. Considere o seguinte programa:
$P(a, y) \leftarrow Q(y)$
$P(x, b) \leftarrow Q(x)$
$P(x, y) \leftarrow Q(x), Q(y)$
$Q(c) \leftarrow$

Desenhe árvores SLD para calcular a(s) resposta(s) deste programa ao objetivo $\leftarrow P(x, y)$, usando as seguintes funções de seleção:

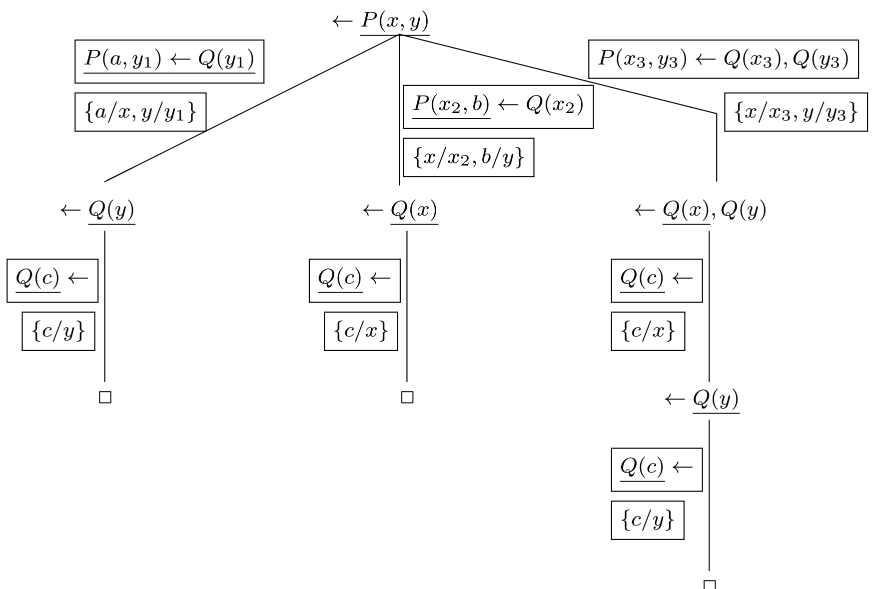
(a) $S(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_1$.

(b) $S(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_n$.

Indique a(s) resposta(s) calculada(s).

Resposta:

(a)



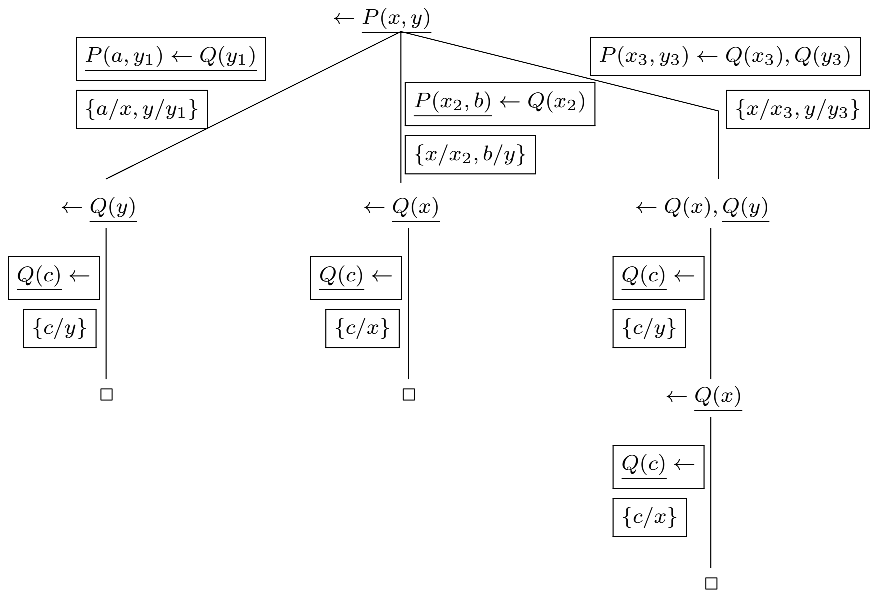
Respostas calculadas:

$$(\{a/x, y/y_1\} \circ \{c/y\}) \mid_{\{x,y\}} = \{a/x, c/y_1, c/y\} \mid_{\{x,y\}} = \{a/x, c/y\}$$

$$(\{x/x_2, b/y\} \circ \{c/x\}) \mid_{\{x,y\}} = \{c/x_2, b/y, c/x\} \mid_{\{x,y\}} = \{c/x, b/y\}$$

$$(\{x/x_3, y/y_3\} \circ \{c/x\} \circ \{c/y\}) \mid_{\{x,y\}} = (\{c/x_3, y/y_3, c/x\} \circ \{c/y\}) \mid_{\{x,y\}}$$
$$= \{c/x_3, c/y_3, c/x, c/y\} \mid_{\{x,y\}} = \{c/x, c/y\}$$

(b)



Respostas calculadas:

$$(\{a/x, y/y_1\} \circ \{c/y\}) \mid_{\{x,y\}} = \{a/x, c/y_1, c/y\} \mid_{\{x\}} = \{a/x, c/y\}$$

$$(\{x/x_2, b/y\} \circ \{c/x\}) \mid_{\{x,y\}} = \{c/x_2, b/y, c/x\} \mid_{\{x\}} = \{c/x, b/y\}$$

$$\begin{aligned}(\{x/x_3, y/y_3\} \circ \{c/y\} \circ \{c/x\}) \mid_{\{x\}} &= (\{x/x_3, c/y_3, c/y\} \circ \{c/x\}) \mid_{\{x,y\}} \\ &= \{c/x_3, c/y_3, c/y, c/x\} \mid_{\{x,y\}} = \{c/x, c/y\}\end{aligned}$$

6.2.4. Considere o seguinte programa:
$Q(x) \leftarrow P(x,y), R(y)$
$R(x) \leftarrow S(f(x), a)$
$S(f(b), a) \leftarrow$
$P(c, b) \leftarrow$
$P(c, d) \leftarrow$
$R(d) \leftarrow$

Desenhe árvores SLD para calcular a(s) resposta(s) deste programa ao objetivo $\leftarrow Q(x)$, usando as seguintes funções de seleção:

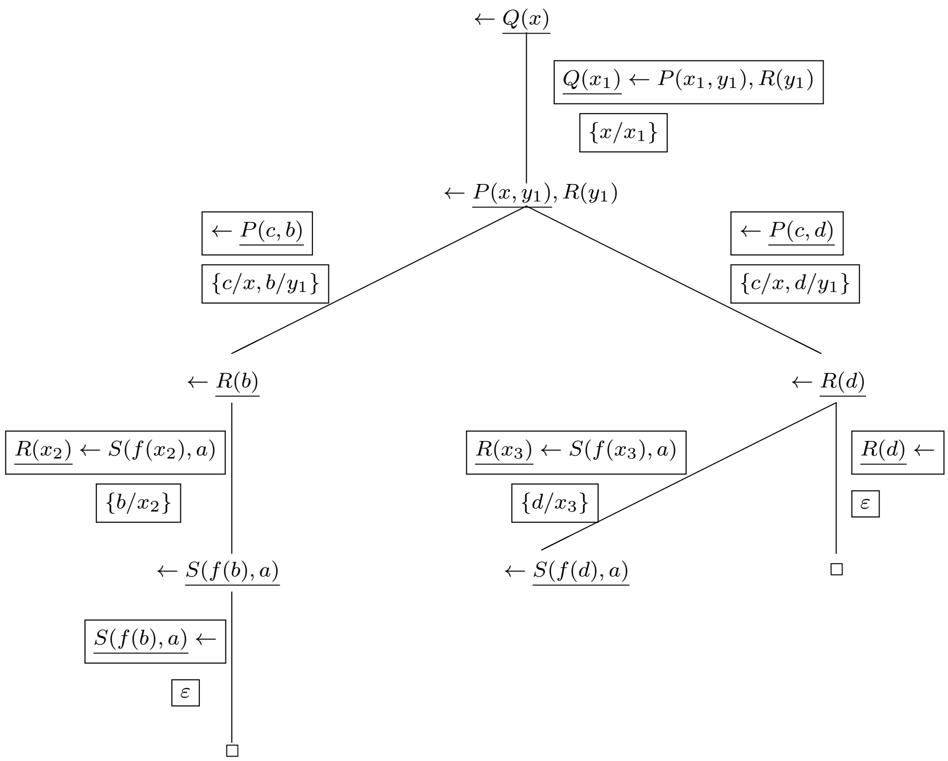
(a) $S(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_1$.

(b) $S(\leftarrow \alpha_1, \ldots, \alpha_n) = \alpha_n$.

Indique a(s) resposta(s) calculada(s).

Resposta:

(a)



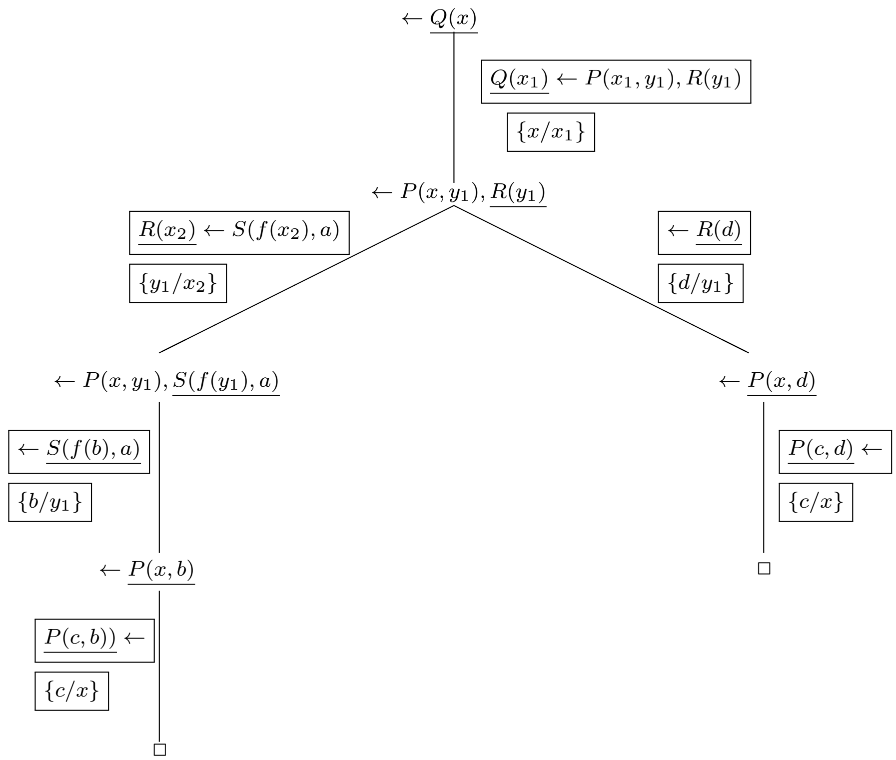
Respostas calculadas:

$$(\{x/x_1\} \circ \{c/x, b/y_1\} \circ \{b/x_2\} \circ \varepsilon) \mid_{\{x\}} = (\{c/x_1, c/x, b/y_1\} \circ \{b/x_2\} \circ \varepsilon) \mid_{\{x\}} =$$

$$\{c/x_1, c/x, b/y_1, b/x_2\} \mid_{\{x\}} = \{c/x\}$$

$$(\{x/x_1\} \circ \{c/x, d/y_1\} \circ \varepsilon) \mid_{\{x\}} = (\{c/x_1, c/x, d/y_1\} \circ \varepsilon) \mid_{\{x\}} = \{c/x\}$$

(b)



Respostas calculadas:

$$(\{x/x_1\}\circ\{y_1/x_2\}\circ\{b/y_1\}\circ\{c/x\})\ |_{\{x\}}= \{c/x_1, b/x_2, b/y_1, c/x\}\ |_{\{x\}}= \{c/x\}$$

$$(\{x/x_1\} \circ \{d/y_1\} \circ \{c/x\})\ |_{\{x\}}= \{c/x_1, d/y_1, c/x\}\ |_{\{x\}}= \{c/x\}$$

Capítulo 7

Prolog

7.1 Componentes básicos, unificação e comparação de termos

7.1.1. Complete a frase:

`a(p)` *é um termo composto se* `a` *for um* ________,
e é um literal se `a` *for um* ________.

Resposta:

`a(p)` *é um termo composto se* `a` *for um functor,*
e é um literal se `a` *for um predicado.*

7.1.2. Classifique as seguintes expressões em afirmações, regras ou objetivos. Se alguma expressão estiver sintaticamente incorreta, indique a razão.

```
p(a).
p(_).
p(a), p(b).
p(X) :- q(X); r(X).
p(X), s(X) :- q(X), r(X).
?- p(a), p(b).
```

Resposta:

- "`p(a).`" é uma afirmação.
- "`p(_).`" é uma afirmação.
- "`p(a), p(b).`" errado, porque uma afirmação só pode conter um literal.

- “p(X) :- q(X); r(X)." é uma regra.
- “p(X), s(X) :- q(X), r(X)." errado, porque a cabeça de uma regra só pode conter um literal.
- “?- p(a), p(b)." é um objetivo.

7.1.3. Diga quais as respostas do PROLOG aos seguintes objetivos:

(a) ?- 'a' = a.
(b) ?- '1' = 1.
(c) ?- _1 = 1.
(d) ?- _1 = '1'.
(e) ?- f(_,_a) = f(a,b).

Resposta:

(a) true.
(b) false.
(c) _1 = 1.
(d) _1 = '1'.
(e) ?- _a = b.

7.1.4. Classifique as seguintes afirmações em verdadeiras ou falsas. Considere que <termo>, <termo1> e <termo2> representam quaisquer termos.

(a) Se <termo1> = <termo2>, então <termo1> == <termo2>.
Resposta: __
(b) Se <termo1> == <termo2>, então <termo1> = <termo2>.
Resposta: __
(c) X = <termo> é sempre verdadeiro.
Resposta: __
(d) f(X) = <termo> é sempre verdadeiro.
Resposta: __
(e) X = <termo>, X == <termo> é sempre verdadeiro.
Resposta: __

Resposta:

(a) Resposta: F
(b) Resposta: V
(c) Resposta: V
(d) Resposta: F
(e) Resposta: V

7.2 A semântica do PROLOG

7.2.1. Considere o seguinte programa, que define frases constituídas por um artigo, um sujeito, um verbo, outro artigo e um complemento.

```
frase(Pal1,Pal2,Pal3,Pal4,Pal5) :-
       palavra(artigo,Pal1),
       palavra(sujeito,Pal2),
       palavra(verbo,Pal3),
       palavra(artigo,Pal4),
       palavra(complemento,Pal5).
palavra(artigo,um).
palavra(artigo,qualquer).
palavra(sujeito,estudante).
palavra(sujeito,professor).
palavra(complemento, livro).
palavra(verbo,compra).
palavra(verbo,consulta).
```

(a) Que objetivo deve ser dado ao PROLOG , para obter todas as frases possíveis? Indique as duas primeiras respostas do PROLOG a esse objetivo.

(b) Que objetivo deve ser dado ao PROLOG , para obter todas as frases possíveis com o verbo `compra`? Indique as duas primeiras respostas do PROLOG a esse objetivo.

(c) Que objetivo deve ser dado ao PROLOG , para obter todas as palavras correspondentes a verbos? Indique todas as respostas do PROLOG a esse objetivo.

(d) Que objetivo deve ser dado ao PROLOG , para obter todos os tipos de palavras (artigo, sujeito, ...)? Indique todas as respostas do PROLOG a esse objetivo.

(e) Que objetivo deve ser dado ao PROLOG , para obter todos as palavras (um, qualquer, ...)? Indique todas as respostas do PROLOG a esse objetivo.

Resposta:

(a)
```
?- frase(_1,_2,_3,_4,_5).
_1 = um,
_2 = estudante,
```

```
_3 = compra,
_4 = um,
_5 = livro ;
_1 = um,
_2 = estudante,
_3 = compra,
_4 = qualquer,
_5 = livro .
```

(b)
```
?- frase(_1,_2,compra,_4,_5).
_1 = um,
_2 = estudante,
_4 = um,
_5 = livro ;
_1 = um,
_2 = estudante,
_4 = qualquer,
_5 = livro
```

(c)
```
?- palavra(verbo,P).
P = compra ;
P = consulta.
```

(d)
```
?- palavra(Tipo,_).
Tipo = artigo ;
Tipo = artigo ;
Tipo = sujeito ;
Tipo = sujeito ;
Tipo = complemento ;
Tipo = verbo ;
Tipo = verbo.
```

(e)
```
?- palavra(_,P).
P = um ;
P = qualquer ;
P = estudante ;
P = professor ;
P = livro ;
P = compra ;
P = consulta.
```

7.2.2. Considere a seguinte informação referente a disciplinas e docentes:

Código	Nome	Ano	Semestre
FP	Fundamentos da Programação	1	1
LP	Lógica para Programação	1	2
IAC	Introdução à Arquitetura de Computadores	1	1
FP	Fundamentos da Programação	1	2
PO	Programação com Objetos	2	1
SO	Sistemas Operativos	2	1
ASA	Análise e Síntese de Algoritmos	2	2
IPM	Interfaces Pessoa Máquina	2	2

Tabela 1: Informação sobre anos/semestres de funcionamento de disciplinas.

Número	Nome
1	António Barros
2	Sara Santos
3	Pedro Silva
4	Joana Alves

Tabela 2: Informação sobre docentes.

Código	Ano letivo	Semestre	Número de docente
FP	2012-2013	1	3
FP	2012-2013	1	1
FP	2013-2014	1	3
LP	2012-2013	2	4
FP	2013-2014	1	2
ASA	2012-2013	2	4
PO	2012-2013	1	1
IPM	2012-2013	2	2
IPM	2012-2013	1	2

Tabela 3: Informação sobre docentes que lecionaram disciplinas, em cada ano letivo/semestre.

(a) Para cada tabela, defina um predicado para representar a informação da tabela. Represente num programa em PROLOG a informação das duas primeiras linhas de cada tabela.

(b) Escreva objetivos para responder às seguintes questões:

   i. Em que anos/semestres é lecionada a disciplina de Fundamentos da Programação?

   ii. Quais os nomes das disciplinas lecionadas pela docente Sara Santos?

   iii. Quais os nomes das disciplinas lecionadas pela docente Joana Alves no ano letivo de 2012-13?

Resposta:

(a)
- *Tabela 1*: `disciplina(Cod,Nome,Ano,Semestre)` significa que a disciplina de código `Cod`, tem o nome `Nome`, e funciona no ano `Ano` e no semestre `Semestre`.

```
disciplina('FP', 'Fundamentos da Programacao',1,1).
disciplina('LP', 'Logica para Programacao',1,2).
```

- *Tabela 2*: `docente(Num, Nome)` significa que o docente de número `Num` tem nome `Nome`.

```
docente(1, 'Antonio Barros').
docente(2, 'Sara Santos').
```

- *Tabela 3*: `lecionou(Cod_Disc, Ano, Semestre, Num_Docente)` significa que no ano letivo `Ano`, no semestre `Semestre`, o docente de numero `Num_Docente`, lecionou a disciplina de codigo `Cod_Disc`.

```
lecionou('FP', '2012-13', 1, 3).
lecionou('FP', '2012-13', 1, 1).
```

(b)

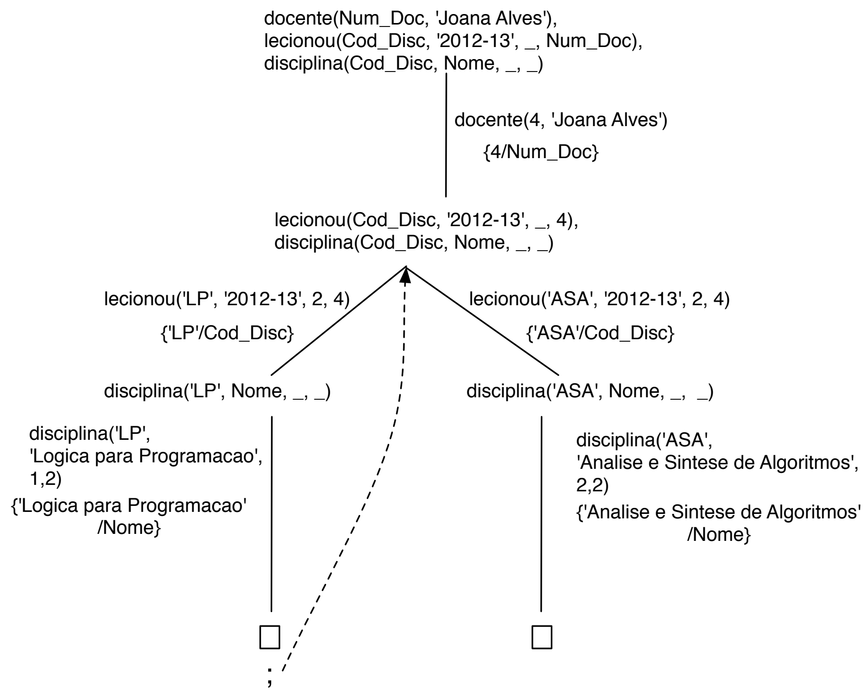
i.
```
?- disciplina(_, 'Fundamentos da Programacao', Ano, Semestre).
```

ii.
```
?- docente(Num_Doc, 'Sara Santos'),
   lecionou(Cod_Disc, _, _, Num_Doc),
   disciplina(Cod_Disc, Nome, _, _).
```

iii.
```
?- docente(Num_Doc, 'Joana Alves'),
   lecionou(Cod_Disc, '2012-13', _, Num_Doc),
   disciplina(Cod_Disc, Nome, _, _).
```

7.2.3. Desenhe a árvore SLD gerada para o último objetivo do exercício anterior, supondo que se pedem todas as respostas. Indique as respostas calculadas apresentadas pelo PROLOG .

Resposta:

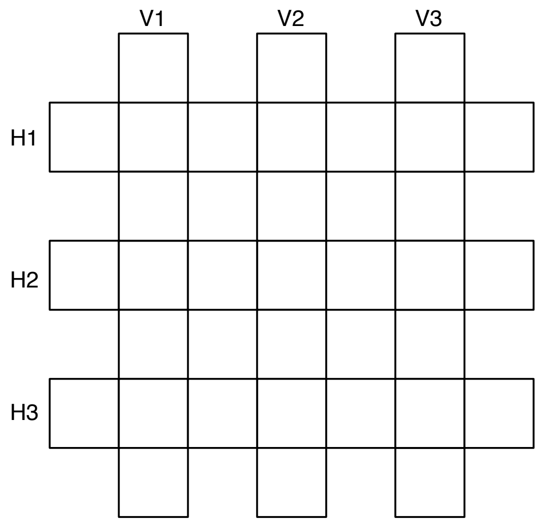


Respostas calculadas:

- $(\{4/Num\_Doc\} \circ \{'LP'/Cod\_Disc\} \circ$
  $\{'Logica\ para\ Programacao'/Nome\}) \mid_{\{Num\_Doc, Cod\_Disc, Nome\}} =$
  $\{4/Num\_Doc, 'LP'/Cod\_Disc, 'Logica\ para\ Programacao'/Nome\}$.

- $(\{4/Num\_Doc\} \circ \{'ASA'/Cod\_Disc\} \circ$
  $\{'Analise\ e\ Sintese\ de\ Algoritmos'/Nome\}) \mid_{\{Num\_Doc, Cod\_Disc, Nome\}}$
  $= \{4/Num\_Doc, 'ASA'/Cod\_Disc, 'Analise\ e\ Sintese\ de\ Algoritmos'/Nome\}$.

7.2.4. Considere as seguintes palavras:
albarda, amarelo, sordido, ameados, tratado, hastear.
Estas palavras devem ser colocadas na seguinte grelha:



(a) Considere o predicado `letras/8`, tal que `letras(Pal, L1, L2, L3, L4, L5,L6,L7)` significa que a palavra `Pal` é constituída pelas letras `L1, L2, L3, L4, L5, L6, L7`, por esta ordem.
Por exemplo, `letras(pardais, p, a, r, d, a,i,s)`.

Usando este predicado escreva as afirmações que indicam quais as letras de cada uma das palavras a colocar na grelha.

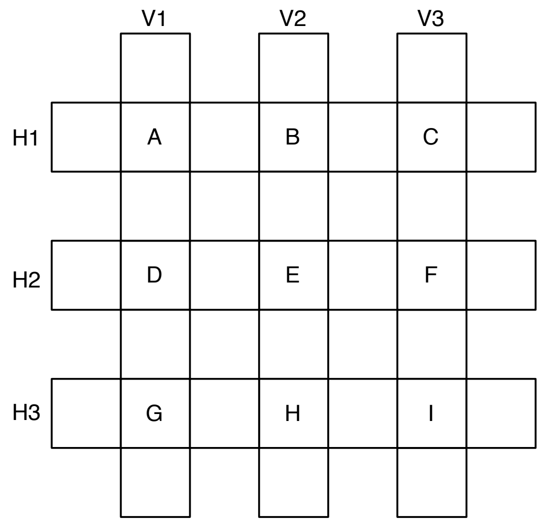
(b) Escreva agora um predicado `solucao/6` para calcular as soluções do puzzle; `solucao(V1, V2, ,V3, H1, H2, H3)` significa que `V1, V2, V3` são as palavras a colocar na vertical, e `H1, H2, H3` são as palavras a colocar na horizontal (veja a grelha acima).

(c) Calcule as soluções do puzzle.

Resposta:

(a)
```
letras(albarda,a,l,b,a,r,d,a).
letras(amarelo,a,m,a,r,e,l,o).
letras(sordido,s,o,r,d,i,d,o).
letras(ameados,a,m,e,a,d,o,s).
letras(tratado,t,r,a,t,a,d,o).
letras(hastear,h,a,s,t,e,a,r).
```

(b) Usamos as letras `A, B, C, D, E, F, G, H, I` para designar as letras na interseção de palavras, como se mostra abaixo:




```
solucao(V1,V2,V3,H1,H2,H3) :-
     letras(V1,_,A,_,D,_,G,_),
     letras(V2,_,B,_,E,_,H,_),
     letras(V3,_,C,_,F,_,I,_),
     letras(H1,_,A,_,B,_,C,_),
     letras(H2,_,D,_,E,_,F,_),
     letras(H3,_,G,_,H,_,I,_).
```

(c)
```
?- solucao(V1,V2,V3,H1,H2,H3).
V1 = amarelo,
V2 = hastear,
V3 = sordido,
H1 = ameados,
H2 = tratado,
H3 = albarda ;
V1 = ameados,
V2 = tratado,
V3 = albarda,
H1 = amarelo,
H2 = hastear,
H3 = sordido ;
false.
```

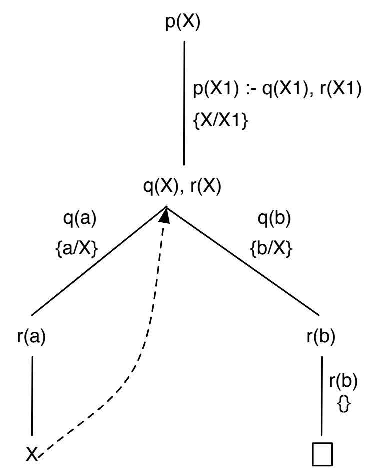
7.2.5. Considere o seguinte programa em PROLOG :

```
p(X) :- q(X), r(X).
```

```
q(a).
q(b).
r(b).
```

Desenhe a árvore SLD gerada para o objetivo `?- p(X).` Indique a resposta calculada apresentada pelo PROLOG .

Resposta:



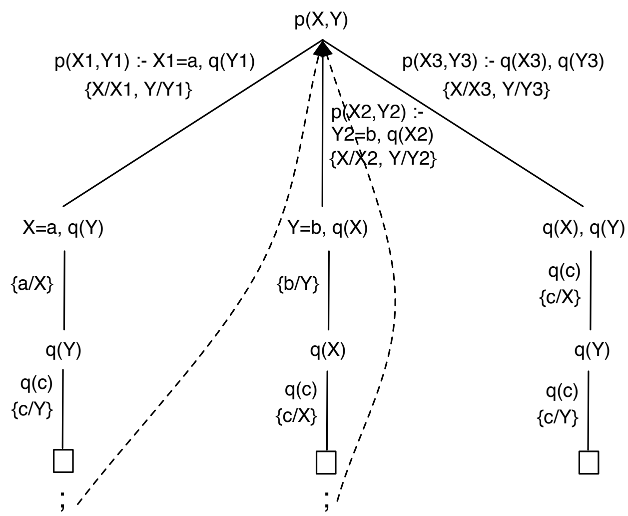
Resposta calculada: $(\{X/X1\} \circ \{b/X\}) \mid_{\{X\}} = (\{b/X1, b/X\}) \mid_{\{X\}} = \{b/X\}$.

7.2.6. Considere o seguinte programa em PROLOG :

```
p(X,Y) :- X=a, q(Y).
p(X,Y) :- Y=b, q(X).
p(X,Y) :- q(X), q(Y).
q(c).
```

Desenhe a árvore SLD gerada para o objetivo `?- p(X,Y).` supondo que se pedem todas as respostas. Indique as respostas calculadas apresentadas pelo PROLOG .

Resposta:



Respostas calculadas:

- $(\{X/X1, Y/Y1\} \circ \{a/X\} \circ \{c/Y\}) \mid_{\{X,Y\}} = \{a/X, c/Y\}$.
- $(\{X/X2, Y/Y2\} \circ \{b/y\} \circ \{c/X\}) \mid_{\{X,Y\}} = \{c/X, b/Y\}$.
- $(\{X/X3, Y/Y3\} \circ \{c/X\} \circ \{c/Y\}) \mid_{\{X,Y\}} = \{c/X, c/Y\}$.

7.2.7. Considere o seguinte programa:

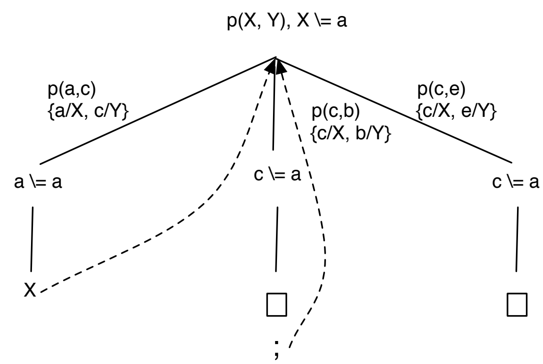
```
p(a,c).
p(c,b).
p(c,e).
p(X, Z) :- p(X, Y), p(Y, Z).
```

Desenhe a árvore SLD gerada para o objetivo

```
?- p(X, Y), X \= a.
```

supondo que se pedem duas respostas. Indique as respostas calculadas apresentadas pelo PROLOG .

Resposta:



Respostas calculadas:

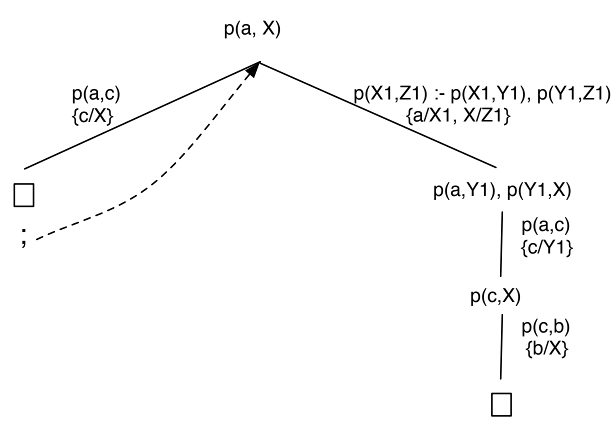
- $\{c/X, b/Y\} \mid_{\{X,Y\}} = \{c/X, b/Y\}$.
- $\{c/X, e/Y\} \mid_{\{X,Y\}} = \{c/X, e/Y\}$.

7.2.8. Considere o seguinte programa:

```
p(a,c).
p(c,b).
p(c,e).
p(X, Z) :- p(X, Y), p(Y, Z).
```

Desenhe a árvore SLD gerada para o objetivo `?- p(a, X).` supondo que se pedem duas respostas. Indique as respostas calculadas apresentadas pelo PROLOG .

Resposta:



Respostas calculadas:

- $\{c/X\} \mid_{\{X\}} = \{c/X\}$.
- $(\{a/X1, X/Z1\} \circ \{c/Y1\} \circ \{b/X\}) \mid_{\{X\}} = \{b/X\}$.

7.2.9. Considere o seguinte programa:

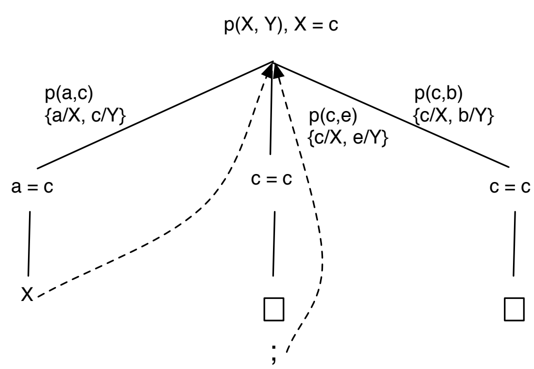
```
p(a,c).
p(c,e).
p(c,b).
p(X, Z) :- p(X, Y), p(Y, Z).
```

Desenhe a árvore SLD gerada para o objetivo

```
?- p(X, Y), X = c.
```

supondo que se pedem duas respostas. Indique as respostas calculadas apresentadas pelo PROLOG .

Resposta:



Respostas calculadas:

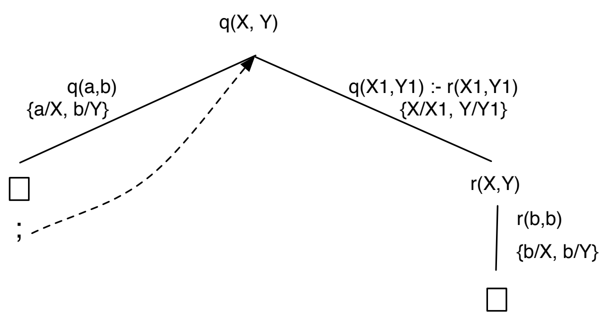
- $\{c/X, e/Y\} \mid_{\{X,Y\}} = \{c/X, e/Y\}$.
- $\{c/X, b/Y\} \mid_{\{X,Y\}} = \{c/X, b/Y\}$.

7.2.10. Considere o seguinte programa:

```
q(a, b).
q(X, Y) :- r(X, Y).
p(a).
r(b, b).
r(a, c).
```

Desenhe a árvore SLD gerada para o objetivo `?- q(X, Y).` supondo que se pedem duas respostas. Indique as respostas calculadas apresentadas pelo PROLOG .

Resposta:



Respostas calculadas:

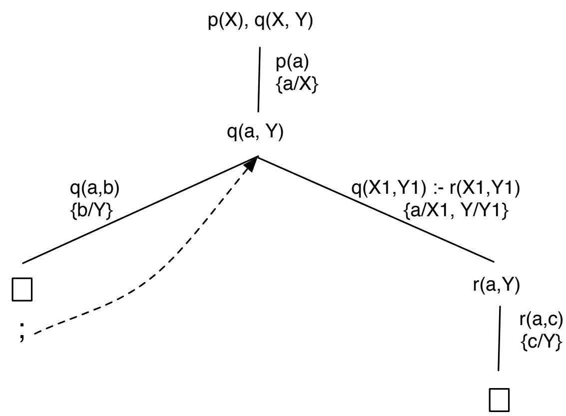
- $\{a/X, b/Y\} \mid_{\{X,Y\}} = \{a/X, b/Y\}$.
- $(\{X/X1, Y/Y1\} \circ \{b/X, b/Y\}) \mid_{\{X,Y\}} = \{b/X, b/Y\}$.

7.2.11. Considere o seguinte programa:

```
q(a, b).
q(X, Y) :- r(X, Y).
p(a).
r(b, b).
r(a, c).
```

Desenhe a árvore SLD gerada para o objetivo `?- p(X), q(X, Y).` supondo que se pedem duas respostas. Indique as respostas calculadas apresentadas pelo PROLOG .

Resposta:



Respostas calculadas:

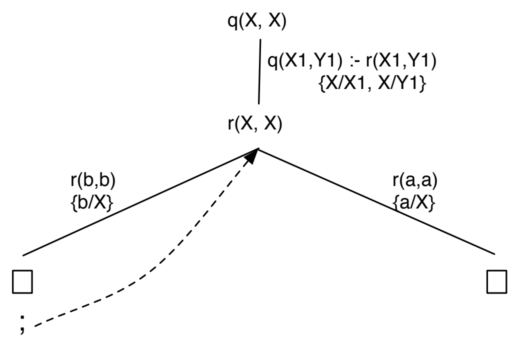
- $(\{a/X\} \circ \{b/Y\}) \mid_{\{X,Y\}} = \{a/X, b/Y\}$.
- $(\{a/X\} \circ \{a/X1, Y/Y1\} \circ \{c/Y\}) \mid_{\{X,Y\}} = \{a/X, c/Y\}$.

7.2.12. Considere o seguinte programa:

```
q(a, b).
q(X, Y) :- r(X, Y).
p(a).
r(b, b).
r(a, a).
```

Desenhe a árvore SLD gerada para o objetivo `?- q(X, X).` supondo que se pedem todas as respostas. Indique as respostas calculadas apresentadas pelo PROLOG .

Resposta:



Respostas calculadas:

- $(\{X/X1, X/Y1\} \circ \{b/X\}) \mid_{\{X\}} = \{b/X\}$.

- $(\{X/X1, X/Y1\} \circ \{a/X\}) \mid_{\{X\}} = \{a/X\}$.

7.2.13. Considere o seguinte programa:

```
p(X) :- q(X), r(X, Y), s(Y).
q(a).
q(b).
r(X, Y) :- q(X), q(Y).
s(b).
s(c).
```

Indique todas as respostas calculadas apresentadas pelo PROLOG para o objetivo `?- r(X, Y).` pela ordem correta.

Resposta:

Respostas calculadas:

- $\{a/X, a/Y\}$.

- $\{a/X, b/Y\}$.

- $\{b/X, a/Y\}$.

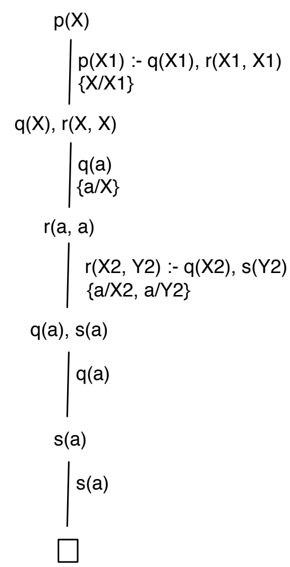
- $\{b/X, b/Y\}$.

7.2.14. Considere o seguinte programa:

```
p(X) :- q(X), r(X, X).
q(a).
q(b).
r(X, Y) :- q(X), s(Y).
s(a).
s(c).
```

Desenhe a árvore SLD gerada para o objetivo `?- p(X).` supondo que se pede uma resposta. Indique a resposta calculada apresentada pelo PROLOG .

Resposta:



Resposta calculada:

- $(\{X/X1\} \circ \{a/X\} \circ \{a/X2, a/Y2\}) \mid_{\{X\}} = \{a/X\}$.

7.2.15. Considere o seguinte programa:

```
p(X) :- q(X), r(X, Y), s(Y).
q(a).
q(b).
```

```
r(X, Y) :- q(X), s(Y).
s(b).
s(c).
```

Indique todas as respostas calculadas apresentadas pelo PROLOG para o objetivo `?- r(X, Y).` pela ordem correta.

Resposta:

Respostas calculadas:

- $\{a/X, b/Y\}$.
- $\{a/X, c/Y\}$.
- $\{b/X, b/Y\}$.
- $\{b/X, c/Y\}$.

7.2.16. Considere o seguinte programa:

```
p(X) :- q(X), r(X, Y), s(Y).
r(X, Y) :-  s(Y), q(X).
s(b).
s(c).
q(a).
q(b).
```

Indique todas as respostas calculadas apresentadas pelo PROLOG para o objetivo `?- r(X, Y).` pela ordem correta.

Resposta:

Respostas calculadas:

- $\{a/X, b/Y\}$.
- $\{b/X, b/Y\}$.
- $\{a/X, c/Y\}$.
- $\{b/X, c/Y\}$.

7.2.17. Considere a seguinte representação para números naturais, usando o functor `suc/1`, tal que `suc(X)` $=$ o sucessor de X: o natural `1` é representado por `1`, o natural `2` é representado por `suc(1)`, e assim sucessivamente. Escreva os seguintes predicados cujos argumentos são todos números naturais usando a representação descrita.

(a) Predicado `maior/2`: `maior(X,Y)` significa que `X` é maior que `Y`. Por exemplo,

```
?- maior(suc(1),suc(1)).
false.

?- maior(suc(suc(1)),suc(1)).
true .

?- maior(1,X).
false.
```

(b) Predicado `igual/2`: `igual(X,Y)` significa que `X` é igual a `Y`.

(c) Predicado `soma/3`: `soma(X,Y,Z)` significa que `Z` é a soma de `X` com `Y`. Por exemplo,

```
?- soma(suc(1), suc(suc(1)),S).
S = suc(suc(suc(suc(1)))).
```

(d) Predicado `dif/3`: `soma(X,Y,Z)` significa que `Z` é a diferença entre `X` e `Y`. Por exemplo,

```
?- dif(suc(1), suc(suc(1)),S).
false.

?- dif(suc(1), suc(1),S).
false.

?- dif(suc(suc(1)), suc(1),S).
S = 1 .
```

Resposta:

(a)
```
maior(suc(_),1).
maior(suc(X),suc(Y)) :- maior(X,Y).
```

(b)
```
igual(1,1).
igual(suc(X),suc(Y)) :- igual(X,Y).
```

(c)
```
soma(1,Y,suc(Y)).
soma(suc(X),Y,suc(Z)) :- soma(X,Y,Z).
```

(d)
```
dif(suc(X),1,X).
dif(suc(X),suc(Y),Z) :- dif(X,Y,Z).
```

7.3 Aritmética em PROLOG

7.3.1. Diga quais as respostas do PROLOG aos seguintes objetivos:

(a) `X = 3 + 2.`
(b) `X is 3 + 2.`
(c) `+(3,2) = 3 + 2.`
(d) `+(3,2) is 3 + 2.`
(e) `Y is X + 5.`
(f) `X = 3, Y is X + 5.`
(g) `4 mod 2 == 0.`
(h) `4 mod 2 =:= 0.`

Resposta:

(a) `X = 3+2.`
(b) `X = 5.`
(c) `true.`
(d) `false.`
(e) `Erro: a variável X não está instanciada`
(f) `X = 3,`
`Y = 8.`
(g) `false.`
(h) `true.`

7.3.2. Defina os predicados `suc/2` e `ant/2`, tais que:

- `suc(N,M)` significa que o sucessor do inteiro `N` é `M`.
- `ant(N,M)` significa que o antecessor do inteiro `N` é `M`.

Resposta:

```
suc(N,M) :- M is N + 1.
ant(N,M) :- M is N - 1.
```

7.3.3. Defina o predicado `perimetro/2`, tal que `perimetro(R,P)` significa que o perímetro da circunferência de raio `R` é `P`. Use a constante `pi` do PROLOG .

Resposta:

```
perimetro(R,P) :-
     P is 2 * pi * R.
```

7.3.4. Defina o predicado `divisor/2`, tal que `divisor(D,N)` (sendo `D` e `N` dois números naturais) significa que `D` é divisor de `N`.

Resposta:

```
divisor(D,N) :-
     mod(N,D) =:=  0.
```

7.3.5. Defina o predicado `aplica_op/4`, tal que `aplica_op(Op,Val1,Val2,R)`, em que `Op` é um dos átomos `+`, `-`, `*` ou `/`, e `Val1` e `Val2` são números, significa que `R` é o resultado de aplicar o operador `Op` a `Val1` e `Val2`. Por exemplo,

```
?- aplica_op(+,8,9,R).
R = 17
```

Resposta:

```
aplica_op(+,Val1,Val2,Res) :- Res is Val1 + Val2.
aplica_op(-,Val1,Val2,Res) :- Res is Val1 - Val2.
aplica_op(*,Val1,Val2,Res) :- Res is Val1 * Val2.
aplica_op(/,Val1,Val2,Res) :- Res is Val1 / Val2.
```

7.3.6. Defina o predicado `raizes/5`, tal que `raizes(A,B,C,X1,X2)` significa que as raízes da equação $Ax^2 + Bx + C$ são `X1` e `X2`. Assuma que as raízes são sempre reais.

Resposta:

```
raizes(A,B,C,X1,X2) :-
     Raiz_quad is sqrt(B ** 2 - 4 * A * C),
     _2A is 2 * A,
     X1 is (-B + Raiz_quad) / _2A,
     X2 is (-B - Raiz_quad) / _2A.
```

7.3.7. Altere o predicado do exercício 7.3.6 de forma a devolver `false` no caso das raízes não serem reais.

Resposta:

```
raizes2(A,B,C,X1,X2) :-
     B2_4AC is B ** 2 - 4 * A * C,
     B2_4AC >= 0,
     Raiz_quad is sqrt(B2_4AC),
     _2A is 2 * A,
     X1 is (-B + Raiz_quad) / _2A,
     X2 is (-B - Raiz_quad) / _2A.
```

7.3.8. Considere o seguinte programa em PROLOG :

```
misterio(X,Y,R) :- misterio(X,Y,0,R).
misterio(_,0,R,R).
misterio(X,Y,Aux,R) :-
     Y_1 is Y - 1,
     Aux_novo is X + Aux,
     misterio(X,Y_1, Aux_novo,R).
```

(a) Desenhe a árvore SLD gerada pelo objetivo `misterio(3,2,R)`, e indique a resposta calculada pelo PROLOG .

(b) Desenhe a árvore SLD gerada se forem pedidas mais respostas.

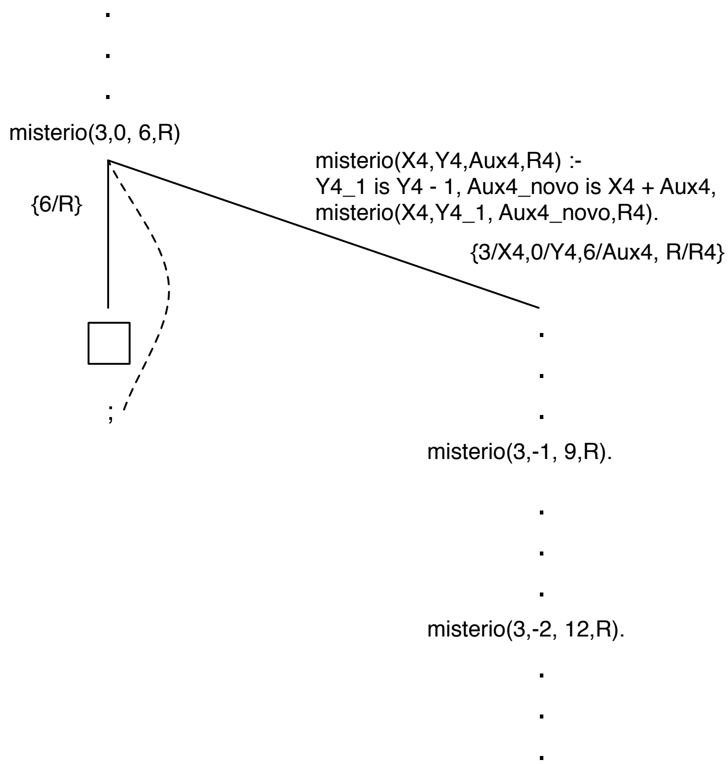
(c) Altere o programa de forma a evitar o problema detetado na alínea anterior.

Resposta:

(a)

```
misterio(3,2,R)
    |  misterio(X1,Y1,R1) :- misterio(X1,Y1,0,R1)
    |  {3/X1,2/Y1,R/R1}
misterio(3,2,0,R)
    |  misterio(X2,Y2,Aux2,R2) :-   Y2_1 is Y2 - 1, Aux2_novo is X2 + Aux2,
    |                               misterio(X2,Y2_1, Aux2_novo,R2).
    |  {3/X2,2/Y2,0/Aux2, R/R2}
Y2_1 is 2 - 1, Aux2_novo is 3 + 0,
misterio(3,Y2_1, Aux2_novo,R)
    |  {1/Y2_1}
Aux2_novo is 3 + 0,
misterio(3,1, Aux2_novo,R)
    |  {3/Aux2_novo}
misterio(3,1, 3,R)
    |  misterio(X3,Y3,Aux3,R3) :-   Y3_1 is Y3 - 1, Aux3_novo is X3 + Aux3,
    |                               misterio(X3,Y3_1, Aux3_novo,R3).
    |  {3/X3,1/Y3,3/Aux3, R/R3}
Y3_1 is 1 - 1, Aux3_novo is 3 + 3,
misterio(3,Y3_1, Aux3_novo,R)
    |  {0/Y3_1}
Aux3_novo is 3 + 3,
misterio(3,0, Aux3_novo,R)
    |  {6/Aux3_novo}
misterio(3,0, 6,R)
    |  {6/R}
    □
```

(b)



(c)
```
misterio(X,Y,Aux,R) :-
        Y > 0,
        Y_1 is Y - 1,
        Aux_novo is X + Aux,
        misterio(X,Y_1, Aux_novo,R).
```

7.3.9. Defina o predicado `digitos_1_N/2`, tal que `digitos_1_N(N, I)` significa que `I` é o inteiro constituído pelos dígitos de `1` a `N`, sendo `N` um dígito de 1 a 9. Por exemplo,

```
?- digitos_1_N(3, I).
I = 123

?- digitos_1_N(0, I).
false.
```

```
?- digitos_1_N(10, I).
false.
```

Baseie-se na seguinte definição:

$$digitos\_1\_N(N) = \begin{cases} 1 & \text{se } N = 1 \\ digitos\_1\_N(N-1) \times 10 + N & \text{se } N > 1 \text{ e } N < 10 \end{cases}$$

Resposta:

```
digitos_1_N(1, 1).
digitos_1_N(N, I) :-
        N > 1,
        N < 10,
        N_menos_um is N - 1,
        digitos_1_N(N_menos_um, I_1),
        I is I_1 * 10 + N.
```

7.3.10. Defina o predicado `digitos_N_1/2`, tal que `digitos_N_1(N, I)` significa que `I` é o inteiro constituído pelos dígitos de `N` a `1`, sendo `N` um dígito de 1 a 9. Por exemplo,

```
?- digitos_N_1(3, I).
I = 321 .

?- digitos_N_1(0, I).
false.

?- digitos_N_1(10, I).
false.
```

Baseie-se no seguinte processo de cálculo:

N	Aux
3	0
2	3
1	32
0	321

Resposta:

```
digitos_N_1(N, I) :-
        N > 0,
        N < 10,
        digitos_N_1(N, I, 0).
digitos_N_1(0, I, I).
digitos_N_1(N, I, Aux) :-
        N > 0,
        N_Aux is Aux * 10 + N,
        N_menos_um is N - 1,
        digitos_N_1(N_menos_um, I, N_Aux).
```

7.3.11. Defina o predicado `soma_digitos/2`, tal que `soma_digitos(N, S)`, em que `N` é um inteiro positivo, significa que `S` é a soma dos dígitos de `N`. Por exemplo,

```
?- soma_digitos(123, S).
S = 6
```

(a) Gerando um processo recursivo.

(b) Gerando um processo iterativo.

Resposta:

(a)
```
soma_digitos(N, N) :- N < 10.
soma_digitos(N, S) :-
        N >= 10,
        Dig_direita is N mod 10,
        N_sem_dig_direita is N // 10,
        soma_digitos(N_sem_dig_direita, S_1),
        S is S_1 + Dig_direita.
```

(b)
```
soma_digitos(N, S) :-
        soma_digitos(N, S, 0).
soma_digitos(0, S, S).
soma_digitos(N, S, Aux) :-
        N > 0,
        Dig_direita is N mod 10,
        N_sem_dig_direita is N // 10,
        N_Aux is Aux + Dig_direita,
        soma_digitos(N_sem_dig_direita, S, N_Aux).
```

7.3.12. Defina o predicado `inverte/2`, tal que `inverte(N, Inv)`, em que `N` é um inteiro positivo, significa que `Inv` é o resultado de inverter os dígitos de `N`. Por exemplo,

```
?- inverte(123, Inv).
Inv = 321 .
```

Resposta:

```
inverte(N, Inv) :-
        inverte(N, Inv, 0).
inverte(0, Inv, Inv).
inverte(N, Inv, Aux) :-
        N > 0,
        Dig_direita is N mod 10,
        N_sem_dig_direita is N // 10,
        N_Aux is Aux * 10 + Dig_direita,
        inverte(N_sem_dig_direita, Inv, N_Aux).
```

7.3.13. Um número natural, n, diz-se triangular se existir um natural m tal que $n = 1 + 2 + ... + (m - 1) + m$.

Defina o predicado `triangular/1`, tal que `triangular(N)` significa que `N` é triangular. Por exemplo,

```
?- triangular(0).
false.

?- triangular(1).
true .

?- triangular(6).
true .

?- triangular(7).
false.
```

Resposta:

```
triangular(N) :- N >= 1, triangular(N,1).

triangular(0,_).
triangular(N,Sub) :-
     N >= Sub,
     N_seg is N - Sub,
     Sub_seg is Sub + 1,
     triangular(N_seg,Sub_seg).
```

7.3.14. Defina o predicado `digitos_pares/1`, tal que `digitos_pares(N)`, em que `N` é um inteiro não negativo, significa que todos os dígitos de `N` são pares. Por exemplo,

```
?- digitos_pares(246).
true .

?- digitos_pares(2416).
false.
```

Resposta:

```
digitos_pares(0).
digitos_pares(N) :-
        (N mod 10) mod 2 =:= 0,
        N_sem_dig_direita is N // 10,
        digitos_pares(N_sem_dig_direita).
```

7.3.15. Defina o predicado `pares/2`, tal que `pares(N, N_pares)`, em que `N` é um inteiro não negativo, significa que `N _pares` é o inteiro constituído por todos os dígitos pares de `N`. Assuma que `N` tem sempre pelo menos um dígito par. Por exemplo,

```
?- pares(862, N).
N = 862 .

?- pares(81632, N).
N = 862 .
```

Baseie-se na seguinte definição:

$$pares(N) = \begin{cases} 0 & \text{se } N = 0 \\ pares(N//10) \times 10 + N \ mod\ 10 & \text{se } N \ mod\ 10 \text{ é par} \\ pares(N//10) & \text{senão} \end{cases}$$

Resposta:

```
pares(0, 0).
pares(N, N_pares) :-
        Dig_direita is N mod 10,
        Dig_direita mod 2 =:= 0,
        N_sem_dig_direita is N // 10,
```

```
        pares(N_sem_dig_direita, N_pares_1),
        N_pares is N_pares_1 * 10 + Dig_direita.
pares(N, N_pares) :-
        Dig_direita is N mod 10,
        Dig_direita mod 2 =\= 0,
        N_sem_dig_direita is N // 10,
        pares(N_sem_dig_direita, N_pares).
```

7.3.16. Defina o predicado `soma_quadrados/2`, tal que `soma_quadrados(N, S)`, em que `N` é um inteiro positivo, significa que `S` é a soma dos quadrados de 1 a `N`. Por exemplo,

```
?- soma_quadrados(1, S).
S = 1 .

?- soma_quadrados(3, S).
S = 14 .
```

(a) Gerando um processo recursivo.

(b) Gerando um processo iterativo.

Resposta:

(a)
```
soma_quadrados(1,1).

soma_quadrados(N,S) :-
        N > 1,
        N_um is N - 1,
        soma_quadrados(N_um,S_N_um),
        S is N ** 2 + S_N_um.
```

(b)
```
soma_quadrados(N,S) :- soma_quadrados(N,1,S).

soma_quadrados(1,S,S).
soma_quadrados(N,Ac,S) :-
        N > 1,
        N_um is N - 1,
        Ac_act is Ac + N ** 2,
        soma_quadrados(N_um,Ac_act,S).
```

7.4 Instruções de leitura e escrita

7.4.1. Complete a tabela abaixo:

Objetivo	Termo introduzido	Resposta
`read(X)`	`f(a,b)`	
`read(f(a,b))`	`a`	
`read(f(X,Y)),R is Y mod X`	`f(2,8)`	
`X=3,read(X+1)`	`3+1`	
`X=3,read(X+1)`	`2+1`	
`read(X+3)`	`+(9,3)`	

Resposta:

Objetivo	Termo introduzido	Resposta
`read(X)`	`f(a,b)`	`X = f(a, b).`
`read(f(a,b))`	`a`	`false.`
`read(f(X,Y)),R is Y mod X`	`f(2,8)`	`X = 2,Y = 8,R = 0.`
`X=3,read(X+1)`	`3+1`	`X = 3.`
`X=3,read(X+1)`	`2+1`	`false.`
`read(X+3)`	`+(9,3)`	`X = 9.`

7.4.2. Complete a tabela abaixo, mostrando *apenas o que é escrito* por cada um dos objetivos:

Objetivo	Escrito
`X = +(2,3), write(X)`	
`X is +(2,3), write(X)`	
`X=3,write(X+1)`	
`X=3,Y = X+1,write(Y)`	
`X=3,Y is X+1,write(Y)`	

Resposta:

Objetivo	Escrito
`X = +(2,3), write(X)`	`2+3`
`X is +(2,3), write(X)`	`5`
`X=3,write(X+1)`	`3+1`
`X=3,Y = X+1,write(Y)`	`3+1`
`X=3,Y is X+1,write(Y)`	`4`

7.4.3. Defina o predicado `escreve_digitos/1`, tal que `escreve_digitos(N)`, em que `N` é um inteiro positivo, provoca a escrita dos dígitos de `N`, um por linha, por ordem inversa. Por exemplo,

```
?- escreve_digitos(1203).
3
0
2
1
```

Resposta:

```
escreve_digitos(N) :-
        N > 0,
        Dig_direita is N mod 10,
        writeln(Dig_direita),
        N_sem_dig_direita is N // 10,
        escreve_digitos(N_sem_dig_direita).
```

7.4.4. Defina o predicado `escreve_N/2`, tal que `escreve_N(N, Car)`, em que `N` é um inteiro positivo, e `Car` é um caractere, provoca a escrita de `Car`, `N` vezes. Por exemplo,

```
?- escreve_N(3,8), escreve_N(2,'*').
888**
```

Resposta:

```
escreve_N(0, _).
escreve_N(N, Car) :-
        N > 0, write(Car),
        N_menos_um is N - 1,
        escreve_N(N_menos_um, Car).
```

7.4.5. Defina o predicado `triangulo/1`, tal que `triangulo(N)`, em que `N` é um inteiro positivo, provoca a escrita de um triângulo, como se mostra no seguinte exemplo,

```
?- triangulo(3).
*
**
***
```

Sugestão: Utilize o predicado `escreve_N/2` do exercício 7.4.4.

Resposta:

```
triangulo(N) :- triangulo(1, N).
triangulo(Cont, N) :- Cont > N.
triangulo(Cont, N) :-
        escreve_N_ast(Cont),
        Cont_mais_um is Cont + 1,
        triangulo(Cont_mais_um, N).
```

7.4.6. Defina o predicado `triangulo/1`, tal que `triangulo(N)`, em que `N` é um inteiro positivo, provoca a escrita de um triângulo, como se mostra no seguinte exemplo,

```
?- triangulo(3).
***
**
*
```

Sugestão: Utilize o predicado `escreve_N/2` do exercício 7.4.4.

Resposta:

```
triangulo(0).
triangulo(N) :-
        escreve_N_ast(N),
        N_menos_um is N - 1,
        triangulo(N_menos_um).
```

7.4.7. Defina o predicado `triangulo/1`, tal que `triangulo(N)`, em que `N` é um inteiro positivo, provoca a escrita de um triângulo, como se mostra no seguinte exemplo,

```
?- triangulo(3).
  *
 ***
*****
```

Sugestão: Utilize o predicado `escreve_N/2` do exercício 7.4.4.

Resposta:

```
triangulo(N) :- triangulo(N, 1).
triangulo(N, L) :- L > N.
triangulo(N, L) :-
        Num_esp is N - L,
```

```
        Num_ast is 2 * L - 1,
        escreve_N(Num_esp, ' '),
        escreve_N(Num_ast, '*'), nl,
        L_mais_1 is L + 1,
        triangulo(N, L_mais_1).
```

7.4.8. Escreva um predicado `beleza_matematica/0`, que escreve o seguinte:

```
?- beleza_matematica.
1 x 8 + 1 = 9
12 x 8 + 2 = 98
123 x 8 + 3 = 987
1234 x 8 + 4 = 9876
12345 x 8 + 5 = 98765
123456 x 8 + 6 = 987654
1234567 x 8 + 7 = 9876543
12345678 x 8 + 8 = 98765432
123456789 x 8 + 9 = 987654321
```

Resposta:

```
beleza_matematica :- beleza_matematica(1,1).

beleza_matematica(Linha,N) :-
    Linha < 10,
    Res is N * 8 + Linha,
    write(N), write(' x 8 + '), write(Linha),
    write(' = '), writeln(Res),
    Prox_Linha is Linha + 1,
    Prox_N is N * 10 + Prox_Linha,
    beleza_matematica(Prox_Linha,Prox_N).
```

7.5 Estruturas

7.5.1. Defina o tipo árvore binária em PROLOG , com os seguintes predicados:

- `nova_arv(A)` significa que `A` é a árvore vazia.
  `cria_arv(R,AE,AD,A)` significa que `A` é a árvore de raiz `R`, árvore esquerda `AE` e árvore direita `AD`.
- `raiz(A,R)` significa que a raiz de `A` é `R`.
  `arv_esq(A,AE)` significa que a árvore esquerda de `A` é `AE`.
  `arv_dir(A,AD)` significa que a árvore direita de `A` é `AD`.

- `e_arv(A)` significa que `A` é uma árvore.
  `nao_vazia(A)` significa que `A` não é a árvore vazia.
  `folha(A)` significa que `A` é uma folha.

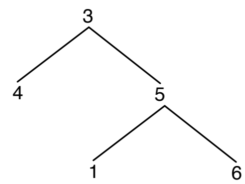
Resposta:

```
% Representacao:
%  arvore vazia: '--'
%  arvore nao vazia: arv(R,AE,AD)

nova_arv('--').
cria_arv(R,AE,AD,arv(R,AE,AD)).


raiz(arv(R,_,_),R).
arv_esq(arv(_,AE,_),AE).
arv_dir(arv(_,_,AD),AD).

e_arv('--').
e_arv(arv(_,AE,AD)) :-
    e_arv(AE),
    e_arv(AD).

nao_vazia(A) :- A \== '--'.

folha(A) :- arv_esq(A,AE),  arv_esq(A,AD),
            nova_arv(AE), nova_arv(AD).
```

7.5.2. Considere a seguinte representação externa para árvores:

- A árvore vazia é representada por `--`.
- Uma árvore não vazia é representada escrevendo a raiz numa linha, na linha seguinte a árvore esquerda com uma indentação de 3 espaços, e na linha seguinte a árvore direita com a mesma indentação.

Por exemplo, a árvore



será representada por

```
3
   4
      --
      --
   5
      1
         --
         --
      6
         --
         --
```

Escreva os transformadores de entrada/saída, `le_arv/1` e `escreve_arv/1`, para o tipo árvore, considerando esta representação externa.

Resposta:

```
le_arv(A) :-
   read(R),
   le_arv(R,A).

le_arv('--',A) :- nova_arv(A).
le_arv(R,A) :-
   R \== '--',
   le_arv(AE),
   le_arv(AD),
   cria_arv(R,AE,AD,A).

escreve_arv(A) :- escreve_arv(A,0).
escreve_arv('--',N) :- escreve_brancos(N),writeln('--').
escreve_arv(A,N) :-
   A \== '--',
   raiz(A,R),
   arv_esq(A,AE),
```

```
   arv_dir(A,AD),
   escreve_brancos(N),
   writeln(R),
   N_seg is N + 3,
   escreve_arv(AE,N_seg),
   escreve_arv(AD,N_seg).

escreve_brancos(0).
escreve_brancos(N) :-
   N > 0,
   write(' '),
   N_1 is N - 1,
   escreve_brancos(N_1).
```

7.5.3. Escreva o predicado `soma_arv/2`, tal que `soma_arv(A,S)` significa que `S` é a soma das raízes da árvore `A`. Assuma que todas as raízes são números.

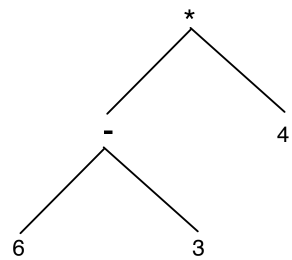
Resposta:

```
soma_arv('--',0).
soma_arv(A,S) :-
   A \== '--',
   arv_esq(A,AE),
   arv_dir(A,AD),
   soma_arv(AE,SAE),
   soma_arv(AD,SAD),
   raiz(A,R),
   S is R + SAE + SAD.
```

7.5.4. Considere a seguinte representação para expressões usando os quatro operadores aritméticos ($+$, $-$, $*$ e $/$), através de árvores binárias:

- Um número é representado por uma árvore cuja raiz é esse número, e cujas árvores esquerda e direita são vazias.
- A expressão $<exp_1>$ $<op>$ $<exp_2>$ é representada por uma árvore cuja raiz é $<op>$, cuja árvore esquerda é a representação de $<exp_1>$ e cuja árvore direita é a representação de $<exp_2>$.

Por exemplo, a expressão $(6-3)*4$ é representada pela árvore



Defina o predicado `avalia_arv/2`, tal que `avalia_arv(A,R)`, em que `A` é uma árvore representando uma expressão aritmética, significa que `R` é o valor da expressão representada por `A`. Por exemplo,

```
?- le_arv(A), avalia_arv(A,R).
|: '*'.
|: '-'.
|: 6.
|: '--'.
|: '--'.
|: 3.
|: '--'.
|: '--'.
|: 4.
|: '--'.
|: '--'.
A = arv(*, arv(-, arv(6, --, --), arv(3, --, --)), arv(4, --, --)),
R = 12
```

Sugestão: Utilize o predicado `aplica_op` do exercício 7.3.5.

Resposta:

```
avalia_arv(A,R) :- folha(A), raiz(A,R).

avalia_arv(A,R) :- arv_esq(A,AE),arv_dir(A,AD),raiz(A,Op),
                   avalia_arv(AE,RAE),avalia_arv(AD,RAD),
                   aplica_op(Op,RAE,RAD,R).
```

7.6 Listas

7.6.1. Diga quais as respostas do PROLOG aos seguintes objetivos:

(a) `[P | R] = [[]].`
(b) `[_ | R] = [].`
(c) `X = [a | [b]].`
(d) `[P,S,T | R] = [1,2,3,4,5].`
(e) `[_,_S,[T | R1] | R2] = [[1],2,[3,[4,5]],8].`
(f) `[_,_S,[T,R1],R2] = [[1],2,[3,[4,5]],8].`

Resposta:

(a) `P = R, R = [].`
(b) `false.`
(c) `X = [a,b].`
(d) `P = 1, S = 2, T = 3, R = [4, 5].`
(e) `_S = 2, T = 3, R1 = [[4, 5]], R2 = [8].`
(f) `_S = 2, T = 3, R1 = [4, 5], R2 = 8.`

7.6.2. Escreva um predicado `nao_membro/2`, que corresponda à negação do predicado `membro/2`, definido na pág. 312 do livro. Assim, `nao_membro(X,L)`, em que `L` é uma lista, significa que `X` não unifica com nenhum dos elementos de `L`. Por exemplo,

```
?- nao_membro(b,[a,b,c]).
false.

?- nao_membro(b,[a,c]).
true

?- nao_membro(X,[a,b,c]).
false.

?- nao_membro(a,[X,Y]).
false.

?- nao_membro(X,[]).
true .
```

Resposta:

```
nao_membro(_,[]).
nao_membro(X,[Y | R]) :-  X \= Y, nao_membro(X,R).
```

7.6.3. Defina o predicado `pertence/2`, tal que `pertence(E,L)` significa que o elemento `E` pertence à lista `L`, isto é, é *igual* a um dos elementos de `L`. Por exemplo,

```
?- pertence(2, [1,2,3]).
true ;
false.

?- pertence(5, [1,2,3]).
false.

?- pertence(X, [1,2,3]).
false.

?- pertence(2, [1,X,3]).
false.

?- pertence(X, [1,X,3]).
true ;
false.

?- pertence(X, [1,Y,3]).
false.
```

Resposta:

```
pertence(P, [Q | _]) :- P == Q.
pertence(P, [_ | R]) :- pertence(P, R).
```

7.6.4. Defina o predicado `nao_pertence/2`, que corresponda à negação do predicado `pertence/2`, definido no exercício 7.6.3. Assim, `nao_pertence(E, L)` significa que o elemento `E` não pertence à lista `L`, isto é, não é *igual* a nenhum dos elementos de `L`. Por exemplo,

```
?- nao_pertence(2, [1,2,3]).
false.

?- nao_pertence(5, [1,2,3]).
true ;
false.
```

```
?- nao_pertence(X, [1,2,3]).
true ;
false.

?- nao_pertence(2, [1,X,3]).
true ;
false.

?- nao_pertence(X, [1,X,3]).
false.

?- nao_pertence(X, [1,Y,3]).
true ;
false.
```

Resposta:

```
nao_pertence(_, []).
nao_pertence(E, [P | R]) :-
     E \== P,
     nao_pertence(E, R).
```

7.6.5. Utilizando os predicados `pertence/2` e `nao_pertence/2` definidos nos exercícios 7.6.3 e 7.6.4, respetivamente, Ddefina o predicado `diferenca/3`, tal que `diferenca(L1, L2, Dif)` significa que `Dif` é a diferença entre as listas `L1` e `L2`, isto é, os elementos de `Dif` são os elementos de `L1` que não são *iguais* a nenhum dos elementos de `L2`. Por exemplo,

```
?- diferenca([1,2,3], [1,3], L).
L = [2] ;
false.

?- diferenca([1,2,3], [11,32], L).
L = [1, 2, 3] ;
false.

?- diferenca([1,2,3], [X], L).
L = [1, 2, 3] ;
false.
```

```
?- diferenca([1,X,3], [X], L).
L = [1, 3] ;
false.
```

Resposta:

```
diferenca([], _, []).
diferenca([P | R], L2, D) :-
     pertence(P, L2),
     diferenca(R, L2, D).
diferenca([P | R], L2, [P | D]) :-
     nao_pertence(P, L2),
     diferenca(R, L2, D).
```

7.6.6. Defina o predicado `subconj/2`, tal que `subconj(L1,L2)` significa que

(a) para cada elemento `El` de `L1` existe um elemento de `L2` que *unifica* com `El`. Por exemplo,

```
?- subconj([1,2,3], [1,2,3,4]).
true ;
false.

?- subconj([1,2,3], [1,2,X,4]).
X = 3 ;
false.
```

(b) para cada elemento `El` de `L1` existe um elemento de `L2` que *é igual* a `El`. Por exemplo,

```
?- subconj([1,2,3], [1,2,3,4]).
true ;
false.

?- subconj([1,2,3], [1,2,X,4]).
false.
```

Resposta:

(a)
```
subconj([],_).
subconj([P | R],L) :- membro(P,L), subconj(R,L).
```

(b)
```
subconj([],_).
subconj([P | R],L) :- pertence(P,L), subconj(R,L).
```

7.6.7. Defina o predicado `lista/1`, tal que `lista(X)` significa que `X` é uma lista. Sugestão: utilize o predicado predifinido `nonvar`. Por exemplo,

```
?- lista(A).
false.

?- lista([A]).
true.

?- lista([]).
true

?- lista([1,2,3]).
true.
```

Resposta:

```
lista(X) :- X == [].
lista(X) :- nonvar(X), X = [_ | _].
```

7.6.8. Defina o predicado `escreve_lista/1`, tal que `escreve_lista(L)` escreve os elementos da lista `L` , um por linha. Por exemplo,

```
?- escreve_lista([6,7,a8]).
6
7
a8
true.
```

Resposta:

```
escreve_lista([]).
escreve_lista([P | R]) :-
     writeln(P),
     escreve_lista(R).
```

7.6.9. Defina o predicado `escreve_lista_num/1`, tal que `escreve_lista_num(L)` escreve os elementos da lista `L` , um por linha, precedidos de `[n]:`, em que `n` é o índice do elemento. Por exemplo,

```
?- escreve_lista_num([6,7,a8]).
[1]: 6
[2]: 7
[3]: a8
true.
```

Resposta:

```
escreve_lista_num(L) :- escreve_lista_num(L,1).
escreve_lista_num([], _).
escreve_lista_num([P | R], N) :-
     write('['), write(N), write(']: '), writeln(P),
     N1 is N + 1,
     escreve_lista_num(R,N1).
```

7.6.10. Defina o predicado `mult_N/3`, tal que `mult_N(L1,N,L2)`, em que `L1` é uma lista de inteiros, significa que `L2` é a lista resultante de multiplicar todos os elementos de `L1` por `N`. Por exemplo,

```
?- mult_N([3,4,1,5,2],3,L).
L = [9, 12, 3, 15, 6].
```

Resposta:

```
mult_N([],_,[]).
mult_N([P | R],N,[P_N | R_N]) :-
     P_N is P * N,
     mult_N(R,N,R_N).
```

7.6.11. (a) Defina o predicado `el_maior_5/1`, tal que `el_maior_5(L)`, em que `L` é uma lista de inteiros, significa que `L` tem pelo menos um elemento maior que 5.

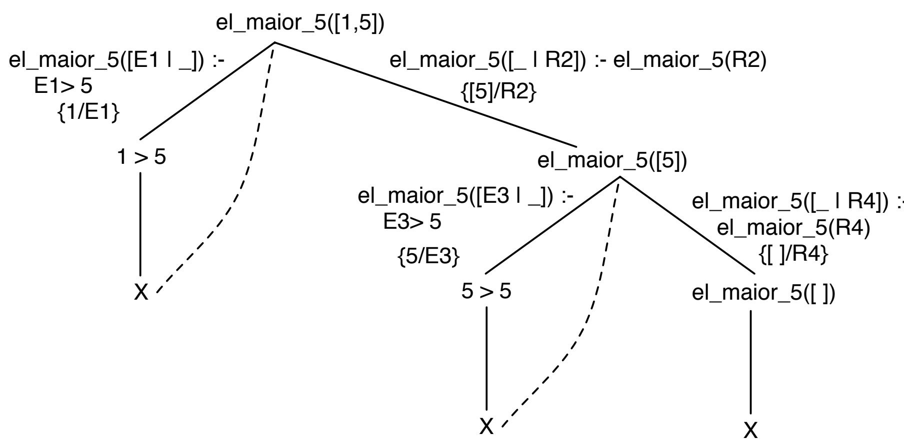
(b) Desenhe a árvore SLD gerada pelo objetivo `el_maior_5([1,5])`.

(c) Desenhe a árvore SLD gerada pelo objetivo `el_maior_5([1,8,9])`, supondo que só é pedida uma resposta.

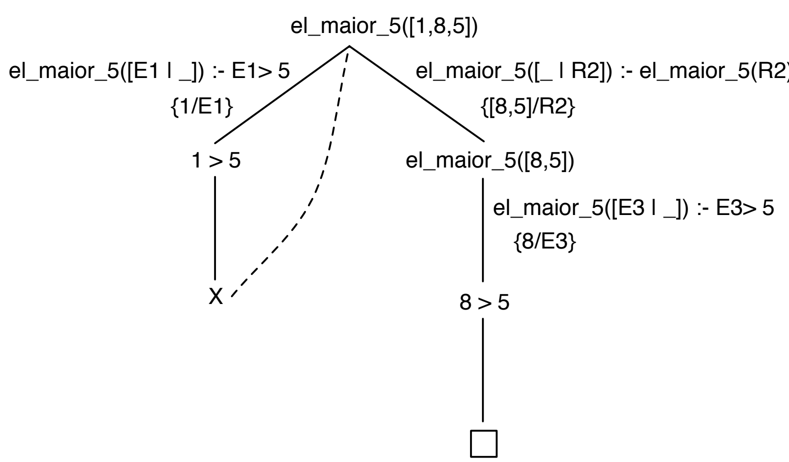
Resposta:

(a)
```
el_maior_5([E | _]) :- E > 5.
el_maior_5([_ | R]) :- el_maior_5(R).
```

(b)



(c)



7.6.12. Defina o predicado `produto/2`, tal que `produto(L, Prod)`, em que `L` é uma lista de inteiros, significa que `Prod` é o produto dos elementos de `L`. Por exemplo,

```
?- produto([], Prod).
Prod = 1.

?- produto([2, 4, 5], Prod).
Prod = 40.
```

(a) Gerando um processo recursivo.

(b) Gerando um processo iterativo.

Resposta:

(a)
```
produto([], 1).

produto([P | R], Prod) :-
     produto(R, Prod_R),
     Prod is P * Prod_R.
```
(b)
```
produto_iter(L, Prod) :-
     produto_iter(L, 1, Prod).

produto_iter([], Prod, Prod).

produto_iter([P | R], Prod_act, Prod) :-
     Novo_Prod_act is Prod_act * P,
     produto_iter(R, Novo_Prod_act, Prod).
```

7.6.13. Defina o predicado `repete_el/3`, tal que `repete_el(El,N,L)` significa que `L` é a lista constituída por `N` ocorrências do elemento `El`. Por exemplo,

```
?- repete_el(a,3,L).
L = [a, a, a]
```

Resposta:

```
repete_el(El,1,[El]).
repete_el(El,N,[El | R]) :-
     N > 1,
     N_1 is N - 1,
     repete_el(El,N_1,R).
```

7.6.14. Defina o predicado `n_vezes/3`, tal que `n_vezes(L1,N,L2)` significa que `L2` é a lista resultante de repetir cada elemento de `L1` `N` vezes. Por exemplo,

```
?- n_vezes([a,b,c],3,L).
L = [a, a, a, b, b, b, c, c, c]
```

Sugestão: use o predicado `repete_el/3` do exercício 7.6.13, e o predicado `junta/3` definido no livro.

Resposta:

```
n_vezes([],_,[]).
n_vezes([P | R],N,L) :-
     repete_el(P,N,P_N),
     n_vezes(R,N,R_N),
     junta(P_N,R_N,L).
```

7.6.15. Defina o predicado `remove_pares/2`, tal que `remove_pares(L1,L2)`, em que `L1` é uma lista de inteiros, significa que `L2` é a lista resultante de remover de `L1` os elementos pares. Por exemplo,

```
?- remove_pares([3,4,1,5,2],L).
L = [3, 1, 5] ;
false.
```

Resposta:

```
remove_pares([],[]).

remove_pares([P | R],[P | Rem_R]) :-
     P mod 2 =\= 0,
     remove_pares(R,Rem_R).

remove_pares([P | R],Rem_R) :-
     P mod 2 =:= 0,
     remove_pares(R,Rem_R).
```

7.6.16. Defina o predicado `conta_maiores_N/3`, tal que `conta_maiores_N(N, L, Cont)`, em que `N` é um inteiro e `L1` é uma lista de inteiros, significa que `Cont` é o número de elementos de `L1` maiores do que `N`. Por exemplo,

```
?- conta_maiores_N(5, [1,5,6,3,7], C).
C = 2
```

Resposta:

```
conta_maiores_N(_, [], 0).
conta_maiores_N(N, [P | R], Cont) :-
     P > N,
     conta_maiores_N(N, R, Cont_R),
     Cont is Cont_R + 1.
conta_maiores_N(N, [P | R], Cont) :-
     P =< N,
     conta_maiores_N(N, R, Cont).
```

7.6.17. Defina o predicado `lista_maiores_N/3`, tal que `lista_maiores_N(N, L1, L2)`, em que `N` é um inteiro e `L1` é uma lista de inteiros, significa que `L2` é a lista constituída pelos elementos de `L1` maiores do que `N`. Por exemplo,

```
?- lista_maiores_N(5, [1,5,6,3,7], C).
C = [6, 7]
```

Resposta:

```
lista_maiores_N(_, [], []).
lista_maiores_N(N, [P | R], [P | R_maiores]) :-
     P > N,
     lista_maiores_N(N, R, R_maiores).
lista_maiores_N(N, [P | R], R_maiores) :-
     P =< N,
     lista_maiores_N(N, R, R_maiores).
```

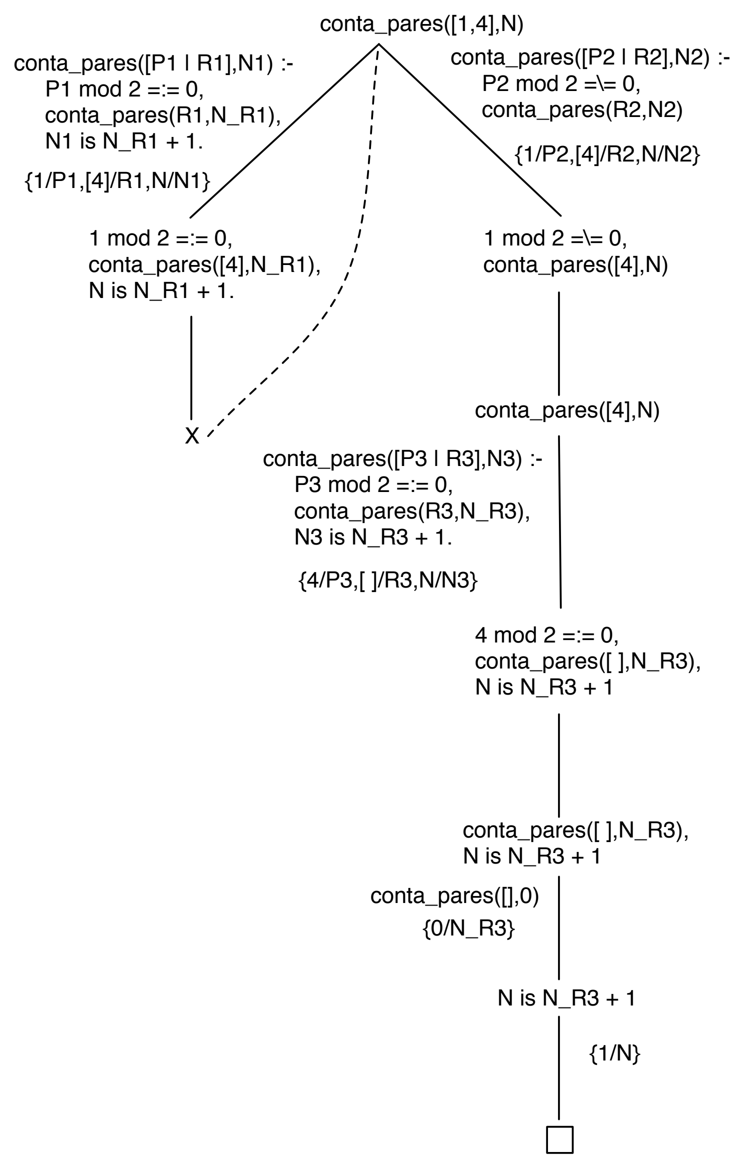
7.6.18. (a) Defina o predicado `conta_pares/2`, tal que `conta_pares(L,N)`, em que `L` é uma lista de inteiros, significa que o número de elementos pares de `L` é `N` .

(b) Desenhe a árvore SLD gerada pelo objetivo `conta_pares([1,4])`.

Resposta:

(a)
```
conta_pares([],0).

conta_pares([P | R],N) :-
     P mod 2 =:= 0,
     conta_pares(R,N_R),
     N is N_R + 1.

conta_pares([P | R],N) :-
     P mod 2 =\= 0,
     conta_pares(R,N).
```

(b)

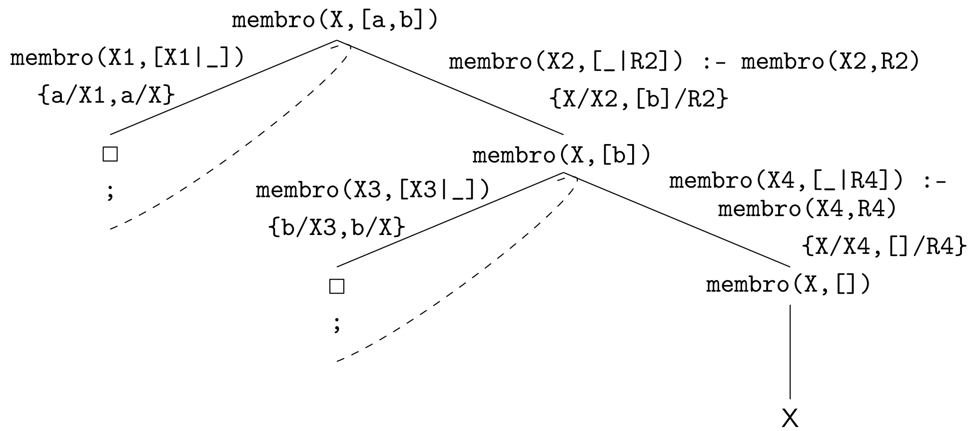


7.6.19. Considere o seguinte programa:

```
membro(X, [X | _]).
membro(X, [_ | R]) :- membro(X, R).
```

Desenhe a árvore SLD gerada pelo objetivo `membro(X,[a,b])`.

Resposta:



7.6.20. Defina o predicado `maximo/2`, tal que `maximo(L,M)`, em que `L` é uma lista de inteiros, significa que o maior dos elementos de `L` é `M`. Por exemplo,

```
?- maximo([2,4,5,1],M).
M = 5 ;
false.

?- maximo([],M).
false.
```

Resposta:

```
maximo([P | R],M) :- maximo(R,P,M).

maximo([],M,M).

maximo([P | R],M_act,M) :-
     P > M_act,
     maximo(R,P,M).

maximo([P | R],M_act,M) :-
     P =< M_act,
     maximo(R,M_act,M).
```

7.6.21. Defina o predicado `comp_maior_lista/2`, tal que `comp_maior_lista(L, C)`, em que `L` é uma lista de listas, significa que o maior comprimento das listas de `L` é `C`. Por exemplo,

```
?- comp_maior_lista([[1,2,3], [4,3,1,3], []], C).
C = 4
```

Sugestão: use o predicado `comprimento(L, C)` definido no livro.

Resposta:

```
comp_maior_lista([P_L | R_L], C) :-
     comprimento(P_L, Comp),
     comp_maior_lista(R_L, Comp, C).

comp_maior_lista([],C,C).

comp_maior_lista([P_L | R], Comp_act, C) :-
     comprimento(P_L, Comp),
     Comp > Comp_act,
     comp_maior_lista(R, Comp, C).

comp_maior_lista([P_L | R], Comp_act, C) :-
     comprimento(P_L, Comp),
     Comp =< Comp_act,
     comp_maior_lista(R, Comp_act, C).
```

7.6.22. Defina o predicado `junta_N/3`, tal que `junta_N(N, L1, L2)`, em que `N` é um inteiro maior ou igual a 1 e `L1` é uma lista, significa que `L2` é o resultado de juntar `L1` a si própria `N` vezes. Por exemplo,

```
?- junta_N(1, [1,2], L).
L = [1, 2, 1, 2] .

?- junta_N(3, [1,2], L).
L = [1, 2, 1, 2, 1, 2, 1, 2] .
```

Sugestão: Use o predicado `junta(L1, L2, L3)` definido no livro.

Resposta:

```
junta_N(1, L1, L2) :-
     junta(L1, L1, L2).
junta_N(N, L1, L2) :-
```

```
     N_1 is N - 1,
     junta_N(N_1, L1, L2_1),
     junta(L1, L2_1, L2).
```

7.6.23. Defina o predicado `remove/3`, tal que `remove(L1,E,L2)`, em que `L1` é uma lista, significa que `L2` é o resultado de remover todas as ocorrências do elemento `E` da lista `L1`. Por exemplo,

```
?- remove([1,2,1,3,4,1],1,L).
L = [2, 3, 4]

?- remove([X],1,L).
L = [X].
```

Resposta:

```
remove([], _,[]).
remove([P | R], E, L) :-
     P == E,
     remove(R, P, L).
remove([P | R], E, [P | L]) :-
     P \== E,
     remove(R, E, L).
```

7.6.24. Defina o predicado `ordenada/1`, tal que `ordenada(L)`, em que `L` é uma lista de inteiros, significa que os elementos de `L` estão ordenados por ordem crescente. Por exemplo,

```
?- ordenada([1,2,3]).
true ;
false.

?- ordenada([1,2,2,3]).
true ;
false.

?- ordenada([1,2,2,1,3]).
false.
```

Resposta:

```
ordenada([]).
ordenada([_]).
ordenada([P, Q | R]) :-
     P =< Q,
     ordenada([Q | R]).
```

7.6.25. Defina o predicado `associa/3`, tal que `associa(Lst, Vals, Res)` em que `Lst` é uma lista que pode conter variáveis e `Vals` é uma lista com o mesmo número de elementos de `Lst`, significa que a lista `Res` resulta de associar os elementos de `Vals` às variáveis de `Lst`, pela mesma ordem. Por exemplo,

```
?- associa([1,X,3, _,5],[2,4], Res).
Res = [1, 2, 3, 4, 5] ;
false.

?- associa([1,X,3, _,5],[2,Y], Res).
Res = [1, 2, 3, Y, 5] ;
false.
```

Resposta:

```
associa([], _, []).
associa([P1 | R1], Vals, [P1 | R]):-
     nonvar(P1),
     associa(R1, Vals, R).
associa([_ | R1], [Val | R_Vals], [Val | R]):-
     associa(R1, R_Vals, R).
```

7.6.26. Defina o predicado `preenche/3`, tal que `preenche(L1, Vals_Possiveis, L2)` em que `L1` é uma lista que pode conter variáveis e `Vals_Possiveis` é uma lista, significa que a lista `L2` resulta de associar às variáveis de `L1` os elementos de `Vals_Possiveis` que não pertencem a `L1`, por uma ordem qualquer. Sugestão: use os predicados `perm`, `diferenca` e `associa`, definidos na pág. 320 do livro, e nos exercícios 7.6.5 e 7.6.25, respetivamente. Por exemplo,

```
?- preenche([1,X,Y,4,5], [1,2,3,4,5],L2).
L2 = [1, 2, 3, 4, 5] ;
L2 = [1, 3, 2, 4, 5] ;
```

```
false.

?- preenche([1,X,Y,4,Z], [1,2,3,4,5],L2).
L2 = [1, 2, 3, 4, 5] ;
L2 = [1, 2, 5, 4, 3] ;
L2 = [1, 3, 2, 4, 5] ;
L2 = [1, 3, 5, 4, 2] ;
L2 = [1, 5, 2, 4, 3] ;
L2 = [1, 5, 3, 4, 2] ;
false.
```

Resposta:

```
preenche(L1, Vals_Possiveis, L2) :-
     diferenca(Vals_Possiveis, L1, Dif),
     perm(Dif, Perm),
     associa(L1, Perm, L2).
```

7.6.27. Considere o tipo dicionário, definido como um conjunto de entradas, sendo cada entrada um par constituído por uma chave e um valor. Num dicionário não podem existir duas entradas com a mesma chave. Defina os seguintes predicados para dicionários:

- `novo_dic(D)` significa que `D` é um dicionário vazio.
  `insere_dic(D,Ch,Val,Novo)`, em que `D` é um dicionário, `Ch` é uma chave e `Val` é um valor, significa que `Novo` é o dicionário resultante de inserir a entrada `(Ch,Val)` em `D`. Se o dicionário original já contiver uma entrada de chave `Ch`, `Novo` é igual a `D`. Por exemplo,

  ```
  ?- novo_dic(D1),insere_dic(D1,a,1,D2),insere_dic(D2,b,2,D3).
  D1 = [],
  D2 = [ (a, 1)],
  D3 = [ (a, 1), (b, 2)].

  ?- novo_dic(D1),insere_dic(D1,a,1,D2),insere_dic(D2,a,2,D3).
  D1 = [],
  D2 = D3, D3 = [ (a, 1)] ;
  false.
  ```

- `procura_dic(D,Ch,Val)`, em que `D` é um dicionário e `Ch` é uma chave, significa que `Val` é o valor associado à chave `Ch` no dicionário D. Se o dicionário original não contiver uma entrada de chave Ch, o predicado devolve falso. Por exemplo,

```
?- D = [(c, 3), (a, 1), (b, 2)], procura_dic(D,a,Val).
D = [ (c, 3), (a, 1), (b, 2)],
Val = 1 ;
false.

?- D = [(c, 3), (a, 1), (b, 2)], procura_dic(D,d,Val).
false.
```

- `atualiza_dic(D,Ch,Val,Novo)`, em que `D` é um dicionário, `Ch` é uma chave e `Val` é um valor, significa que `Novo` é o dicionário resultante de atualizar a entrada de chave `Ch` para `(Ch,Val)` em `D`. Se o dicionário original não contiver uma entrada de chave `Ch`, o predicado devolve falso. Por exemplo,

```
?- D = [(c, 3), (a, 1), (b, 2)], atualiza_dic(D,b,22,Novo).
D = [ (c, 3), (a, 1), (b, 2)],
Novo = [ (c, 3), (a, 1), (b, 22)] ;
false.

?- D = [(c, 3), (a, 1), (b, 2)], atualiza_dic(D,d,22,Novo).
false.
```

Resposta:

```
novo_dic([]).

insere_dic([], Ch, Val,[(Ch,Val)]).
insere_dic([(Ch,Val) | R],  Ch, _,[(Ch,Val) | R]).
insere_dic([E | R],  Ch, Val,[E | R1]) :-
     E \= (Ch,_),
     insere_dic(R,Ch,Val,R1).

procura_dic([(Ch,Val) | _],  Ch, Val).
procura_dic([E | R],  Ch, Val) :-
     E \= (Ch,_),
     procura_dic(R,Ch,Val).

atualiza_dic([(Ch,_) | R],  Ch, Val,[(Ch,Val) | R]).
atualiza_dic([E | R],  Ch, Val,[E | R1]) :-
     E \= (Ch,_),
     atualiza_dic(R,Ch,Val,R1).
```

7.6.28. Defina o predicado `alisa/2`, tal que `alisa(L1,L2)`, em que `L1` é uma lista cujos elemenos podem ser listas, significa que `L2` é a lista de elementos atómicos de `L1`. Por exemplo,

```
?- alisa([9,[8,[[a,b]]],[]], L).
L = [9, 8, a, b] ;
false.

?- alisa([9,[8,[[X,Y]]],[]], L).
L = [9, 8, X, Y] ;
false.
```

Sugestão: Use o predicado `lista` do exercício 7.6.7, e os predicados predefinidos `var` e `nonvar`.

Resposta:

```
alisa([],[]).
alisa([P | R], [P | AR]) :-  var(P),
                             alisa(R,AR).
alisa([P | R], [P | AR]) :-  nonvar(P), P \= [], P \= [_ | _],
                             alisa(R,AR).
alisa([P | R], L) :- lista(P),
                     alisa(P,AP), alisa(R,AR),
                     junta(AP,AR,L).
```

7.6.29. Defina o predicado `combina/3`, tal que `combina(L1,L2,L3)` significa que `L3` é uma lista de listas de 2 elementos, o 1º dos quais pertence a `L1`, e o 2º a `L2`. Por exemplo,

```
combina([a,b], [d,e,f],
         [[a, d], [a, e], [a, f], [b, d], [b, e], [b, f]])
```

Resposta:

Definimos primeiro o predicado `combina_1/3`, tal que `combina_1(E,L1,L2)` significa que `L2` é uma lista de listas de 2 elementos, o 1º dos quais é `E`, e o 2º pertence a `L2`. Por exemplo, `combina_1(a, [d,e,f], [[a, d], [a, e], [a, f]])`.

```
combina_1(_,[], []).

combina_1(E, [P | R],  [[E, P] | C_1_R]) :-
```

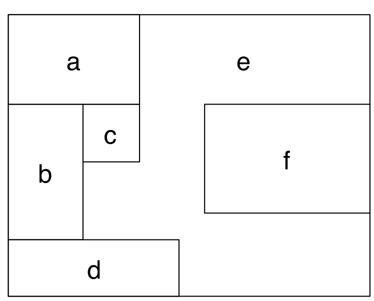
```
        combina_1(E, R, C_1_R).

combina([],_, []).

combina([P | R], L2, L3) :-
        combina_1(P, L2, C_P_L2),
        combina(R, L2, C_R_L2),
        junta(C_P_L2, C_R_L2, L3).
```

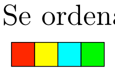
7.6.30. Prova-se que 4 cores são suficientes para colorir qualquer mapa, de forma a que regiões adjacentes tenham cores diferentes.

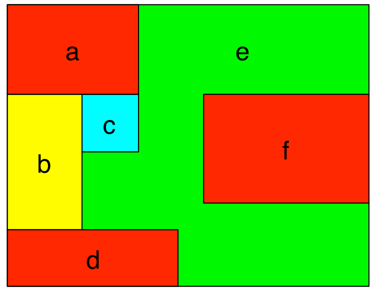
Considere o seguinte algoritmo para colorir um mapa dado, com 4 cores também dadas:

- Ordenar as regiões e as cores por quaisquer ordens.
- Para cada região R, colori-la com a primeira cor da lista de cores C, tal que não existe nenhuma região adjacente a R com a cor C.

Por exemplo, considere-se o mapa da figura abaixo e 4 cores, vermelho, amarelo, azul e verde.



Se ordenarmos as regiões por ordem alfabética, e as cores pela ordem , a aplicação do algoritmo acima leva ao seguinte mapa:



Pretende-se escrever um programa em PROLOG para colorir mapas de acordo com este algoritmo.

Em primeiro lugar, deve definir-se uma representação para cada região, contendo a informação relevante para o algoritmo, ou seja, o nome da região, a sua cor, e as cores das regiões adjacentes. Podemos usar a seguinte estrutura para representar uma região:

```
regiao(<nome>,<cor>,<cores das regiões adjacentes>).
```

Por exemplo, antes de ser atribuída qualquer cor, a região `c` é representada pela estrutura `regiao(c,C_c,[C_a,C_b,C_e])`. Esta mesma região, após atribuídas as cores vermelho e amarelo às regiões `a` e `b`, respetivamente, será representada por `regiao(c,C_c,[vermelho,amarelo,C_e])`.

Sendo um mapa um conjunto de regiões, pode ser representado por uma lista de regiões. Por exemplo, o mapa acima será representado pela lista

```
[regiao(a,C_a,[C_b,C_c,C_e]), regiao(b,C_b,[C_a,C_c,C_d,C_e]),
 regiao(c,C_c,[C_a,C_b,C_e]),regiao(d,C_d,[C_b,C_e]),
 regiao(e,C_e,[C_a,C_b,C_c,C_d,C_f]),regiao(f,C_f,[C_e])]
```

(a) Defina o predicado `colorir_regiao/2`, tal que `colorir_regiao(regiao(R,C_R,[C_R1,...,C_Rn]), Cores)`, em que `Cores` é uma lista de cores, significa que `C_R` é a primeira cor da lista `Cores` que não pertence a `[C_R1,...,C_Rn]`. Por exemplo,

```
?- colorir_regiao(regiao(a,C_a,[C_b,C_c,C_e]),
                  [vermelho, amarelo, azul, verde]).
```

```
C_a = vermelho

?- colorir_regiao(regiao(c, C_c, [vermelho, amarelo, C_e]),
                  [vermelho, amarelo, azul, verde]).
C_c = azul
```

(b) Defina agora o predicado `colorir_mapa/2`, tal que `colorir_mapa(Mapa,Cores)`, em que `Mapa` é uma lista de regiões e `Cores` é uma lista de cores, aplica o predicado `colorir_regiao/2` a cada uma das regiões de `Mapa`.

Resposta:

(a) Poderíamos pensar na seguinte solução óbvia:

```
colorir_regiao(regiao(_,Cor,Vizinhos), Cores) :-
      membro(Cor,Cores),
      nao_membro(Cor,Vizinhos).
```

usando os predicados `membro/2` e `nao_membro/2`, definidos no livro e no exercício 7.6.2, respetivamente. No entanto, obteríamos a seguinte interação:

```
?- colorir_regiao(regiao(a,C_a,[C_b,C_c,C_e]),
                  [vermelho, amarelo, azul, verde]).
false.
```

Este comportamento deve-se ao predicado `nao_membro/2`. Com efeito, `nao_membro(vermelho,[C_b,C_c,C_e])` é falso, porque `vermelho` *unifica* com qualquer dos elementos da lista `[C_b,C_c,C_e]`.
Assim, em vez do predicado `nao_membro/2`, deve ser usado o predicado `nao_pertence/2` definido no exercício 7.6.4:

```
colorir_regiao(regiao(_,Cor,Vizinhos), Cores) :-
      membro(Cor,Cores),
      nao_pertence(Cor,Vizinhos).
```

obtendo o comportamento desejado.

(b)
```
colorir_mapa([Reg | Regs], Cores) :-
      colorir_regiao(Reg, Cores),
      colorir_mapa(Regs, Cores).

colorir_mapa([], _).
```

7.6.31. Considere o algoritmo de ordenação por árvore, executado em dois passos sequenciais:

(a) Criar uma árvore binária com os elementos a ordenar: *a árvore binária de procura.*

(b) Percorrer a árvore criada de forma a visitar as raízes por ordem.

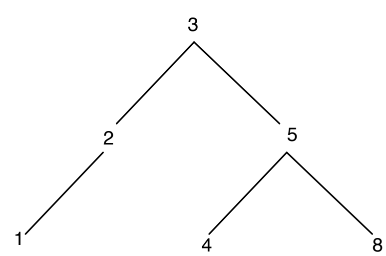
Criar a árvore binária de procura:

Começando com uma árvore vazia, inserem-se os elementos a ordenar, um a um, do seguinte modo:

(a) O resultado de inserir um elemento numa árvore vazia é uma árvore cuja raiz é o elemento e cujas árvores esquerda e direita são ambas vazias.

(b) Um elemento é inserido numa árvore não vazia comparando o elemento com a raiz da árvore. Se o elemento for maior do que a raiz da árvore, o elemento é inserido, utilizando o mesmo método, na árvore direita da árvore inicial, caso contrário, é inserido, pelo mesmo método, na árvore esquerda.

Por exemplo, se os elementos a ordenar fossem 3, 2, 5, 1, 4 e 8, a árvore binária de procura seria:



Percorrer a árvore binária de procura:

(a) Percorrer uma árvore vazia não causa nenhuma acção.

(b) Para percorrer uma árvore não vazia, primeiro percorremos a sua árvore esquerda, depois visitamos a raiz, depois percorremos a sua árvore direita.

Defina o predicado `ordena/2`, tal que `ordena(L,L_ord)`, em que `L` é uma lista de inteiros, significa que `L_ord` é o resultado de ordenar a lista `L` por ordem crescente. Por exemplo,

```
?- ordena([3,2,5,1,4,8],L).
L = [1, 2, 3, 4, 5, 8] ;
false.
```

Resposta:

```
% ordena(L,L_ord) significa que L_ord é o resultado de ordenar
% a lista L por ordem crescente
ordena(L, L_ord) :-
     cria_arv_procura(L, AP),
     percorre(AP, L_ord).

% cria_arv_procura(L, AP) significa que AP é a árvore binária
% de procura criada a partir dos elementos da lista L
cria_arv_procura(L, AP) :-
     nova_arv(Arv_aux),
     cria_arv_procura(L, Arv_aux, AP).

cria_arv_procura([], Arv_aux, Arv_aux).

cria_arv_procura([P | R], Arv_aux, AP) :-
     insere(P,Arv_aux,Nova_Arv_aux),
     cria_arv_procura(R, Nova_Arv_aux, AP).

% insere(X,A,A_mais_X) significa que A_mais_X é a árvore
% resultante de inserir o elemento X na árvore A

insere(X,A,A_mais_X) :-
     nova_arv(A),
     cria_arv(X,A,A,A_mais_X).

insere(X,A,A_mais_X) :-
     raiz(A,R),X < R,
     arv_esq(A,AE), arv_dir(A,AD),
     insere(X,AE,AE_mais_X),
     cria_arv(R,AE_mais_X,AD,A_mais_X).

insere(X,A,A_mais_X) :-
     raiz(A,R), X >= R,
     arv_esq(A,AE), arv_dir(A,AD),
     insere(X,AD,AD_mais_X),
     cria_arv(R,AE,AD_mais_X,A_mais_X).

% percorre(A, L) significa que L é a lista resultante
% de visitar as raizes da árvore A pela ordem adequada
```

```
percorre(A, []) :- nova_arv(A).

percorre(A, L) :-
     raiz(A,R),  arv_esq(A,AE), arv_dir(A,AD),
     percorre(AE, L_AE),
     junta(L_AE,[R], L_AE_R),
     percorre(AD, L_AD),
     junta(L_AE_R,L_AD, L).
```

7.7 Corte e negação

7.7.1. **Lista de permutações - lista_perm ficheiro listas.pl**

7.7.2. Considere o seguinte programa:

```
t(X) :- p(_,X).
t(4).
p(a, b).
p(X, Y) :- s(X, Y).
p(X, Y) :- q(X), !, r(Y).
s(3, 2).
q(1).
q(2).
r(a).
r(b).
r(c).
```

(a) Quais as respostas do PROLOG ao objetivo `?- t(X).`?

(b) Altere o programa, através da introdução de um operador de corte, de forma a obter apenas as 2 primeiras e a última respostas.

Resposta:

(a)
```
X = b ;
X = 2 ;
X = a ;
X = b ;
X = c ;
X = 4.
```

(b) `p(X, Y) :- s(X, Y), !.`

7.7.3. Considere o seguinte programa:

```
t(X) :- p(1,X).
t(4).
p(a, b).
p(X, Y) :- s(X, Y), !.
p(X, Y) :- q(X), r(Y).
s(3, 2).
q(1).
q(2).
r(a).
r(b).
r(c).
```

(a) Quais as respostas do PROLOG ao objetivo `?- t(X).`?

(b) Altere o programa, através da introdução de um operador de corte, de forma a obter apenas a primeira e a última respostas.

Resposta:

(a)
```
X = a ;
X = b ;
X = c ;
X = 4.
```

(b) `p(X, Y) :- q(X), r(Y), !.`

7.7.4. Considere o seguinte programa:

```
s(3, 2).
q(1).
q(2).
r(a).
r(b).
r(c).
t(X) :- p(X, _).
t(4).
p(X, Y) :- s(X, Y).
```

Indique todas as respostas do PROLOG ao objetivo `?- t(X).` supondo que se adiciona no fim do programa cada uma das regras:

(a) `p(X, Y) :- q(X), r(Y).`

(b) `p(X, Y) :- q(X), !, r(Y).`

(c) `p(X, Y) :- q(X), r(Y), !.`

Resposta:

(a)
```
X = 3 ;
X = 1 ;
X = 1 ;
X = 1 ;
X = 2 ;
X = 2 ;
X = 2 ;
X = 4.
```

(b)
```
X = 3 ;
X = 1 ;
X = 1 ;
X = 1 ;
X = 4.
```

(c)
```
X = 3 ;
X = 1 ;
X = 4.
```

7.7.5. Considere o seguinte programa:

```
s(3, 2).
q(1).
q(2).
r(a).
r(b).
r(c).
t(X) :- p(X, _).
t(4).
p(X, Y) :- !, q(X), r(Y).
p(X, Y) :- s(X, Y).
```

(a) Indique todas as respostas do PROLOG ao objetivo `?- t(X).` .

(b) Suponha agora que a penúltima regra é substituída por `p(X, Y) :- q(X), !, r(Y).` Indique todas as respostas do PROLOG ao objetivo `?- t(X).` .

(c) Suponha agora que a penúltima regra é substituída por `p(X, Y) :- q(X), r(Y), !.` Indique todas as respostas do PROLOG ao objetivo `?- t(X).` .

Resposta:

(a)
```
X = 1 ;
X = 1 ;
X = 1 ;
X = 2 ;
X = 2 ;
X = 2 ;
X = 4.
```
(b)
```
X = 1 ;
X = 1 ;
X = 1 ;
X = 4.
```
(c)
```
X = 1 ;
X = 4.
```

7.7.6. Considere o seguinte programa:

```
s(X) :- p(1,X).
s(4).
p(a, b).
p(X, Y) :- t(X, Y), !.
p(X, Y) :- q(X), r(Y).
t(3, 2).
q(1).
q(2).
r(a).
r(b).
r(c).
```

(a) Quais as respostas do PROLOG ao objetivo `?- s(X).`?

(b) Altere o programa, através da introdução de um operador de corte, de forma a obter apenas a primeira e a última respostas.

Resposta:

(a)
```
X = a ;
X = b ;
```

```
X = c ;
X = 4.
```

(b) `p(X, Y) :- q(X), r(Y), !.`

7.7.7. Considere o seguinte programa:

```
p(X,Y) :- q(X),r(Y).
p(5,z).
q(1).
q(2).
r(a).
r(b).
```

(a) Indique todas as respostas do PROLOG ao objetivo `p(X,Y)`.
Resposta:

```
X = 1,
Y = a ;
X = 1,
Y = b ;
X = 2,
Y = a ;
X = 2,
Y = b ;
X = 5,
Y = z.
```

(b) Suponha que a primeira cláusula é substituída por `p(X,Y) :- !,q(X),r(Y).` Indique todas as respostas do PROLOG ao objetivo `p(X,Y)`.
Resposta:

```
X = 1,
Y = a ;
X = 1,
Y = b ;
X = 2,
Y = a ;
X = 2,
Y = b.
```

(c) Suponha que a primeira cláusula é substituída por `p(X,Y) :- q(X),!,r(Y).` Indique todas as respostas do PROLOG ao objetivo `p(X,Y)`.
Resposta:

```
X = 1,
Y = a ;
X = 1,
Y = b.
```

(d) Suponha que a primeira cláusula é substituída por `p(X,Y) :- q(X),r(Y),!.` Indique todas as respostas do PROLOG ao objetivo `p(X,Y)`.
Resposta:

```
X = 1,
Y = a.
```

7.7.8. Considere o seguinte programa:

```
p(2, 3).
p(X,Y) :- q(X), r(Y).
q(1).
q(2).
q(3).
r(1).
r(3).
```

(a) Indique todas as respostas do PROLOG ao objetivo `p(X,Y)`.
Resposta:

```
X = 2,
Y = 3 ;
X = 1,
Y = 1 ;
X = 1,
Y = 3 ;
X = 2,
Y = 1 ;
X = 2,
Y = 3 ;
X = 3,
Y = 1 ;
X = 3,
Y = 3.
```

(b) Introduza um operador de corte, de modo a só obter a primeira resposta.
Resposta:

```
p(2, 3) :- !.
```

(c) Introduza um operador de corte, de modo a só obter as 3 primeiras respostas.
Resposta:

```
p(X,Y) :- q(X), !, r(Y).
```

(d) Introduza um operador de corte, de modo a só obter as 2 primeiras respostas.
Resposta:

```
p(X,Y) :- q(X), r(Y), !.
```

(e) Introduza um operador de corte, de modo a só obter as 5 primeiras respostas.
Resposta:

```
q(2) :- !.
```

7.7.9. Defina o predicado `classe/2`, tal que `classe(N, C)`, em que `N` é um inteiro, significa que a classe de `N` é `C`, sendo o domínio de `C` o conjunto {`zero`, `positivo`, `negativo`}. Use o operador de corte sempre que tal melhorar o seu programa. Por exemplo,

```
?- classe(10, C).
C = positivo.

?- classe(0, C).
C = zero.

?- classe(-20, C).
C = negativo.
```

Resposta:

```
classe(0,zero) :- !.
classe(N,positivo) :- N > 0, !.
classe(N, negativo) :- N < 0.
```

7.7.10. Defina o predicado `separa/3`, tal que `separa(L, L_pos, L_neg)`, em que `L` é uma lista de inteiros, significa que `L_pos` é a lista com os elementos positivos ou nulos de `L`, e `L_neg` é a lista com os elementos negativos de `L`. Use o operador de corte sempre que tal melhorar o seu programa. Por exemplo,

```
?- separa([1,-2,3,4,5,0,-6], L_pos, L_neg).
L_pos = [1, 3, 4, 5, 0],
L_neg = [-2, -6].
```

Resposta:

```
separa([], [], []) :- !.
separa([P | R], [P | R_pos], L_neg) :-
     P >= 0,
     !,
     separa(R, R_pos, L_neg).
separa([P | R], L_pos, [P | R_neg]) :-
     P < 0,
     !,
     separa(R, L_pos, R_neg).
```

7.7.11. Defina o predicado `intersecao/3`, tal que `intersecao(L1, L2, I)`, em que `L1` e `L2` são listas de inteiros, significa que `I` é a lista com os elementos comuns a `L1` e `L2`. Use o operador de corte sempre que tal melhorar o seu programa. O seu programa não deve permitir que o pedido de múltiplas respostas origine respostas erradas. Por exemplo,

```
?- intersecao([1,2,3,4,5], [1,3,5], I).
I = [1, 3, 5].
```

Resposta:

```
intersecao([], _, []) :- !.
intersecao(_, [], []) :- !.
intersecao([P | R], L, [P | IRL]) :-
     membro(P, L),
     !,
     intersecao(R, L, IRL).
intersecao([P | R], L, IRL) :-
     \+ membro(P, L),
     intersecao(R, L, IRL).
```

A segunda cláusula, embora não necessária, evita que o programa percorra a primeira lista quando a segunda lista é vazia.

7.7.12. Defina o predicado `uniao/3`, tal que `uniao(L1, L2, U)`, em que `L1` e `L2` são listas de inteiros representando conjuntos, isto é, que não têm elementos repetidos, significa que `U` é a lista união de `L1` e `L2`. Use o operador de corte sempre que tal melhorar o seu programa. O seu programa não deve permitir que o pedido de múltiplas respostas origine respostas erradas. Por exemplo,

```
?- uniao([1,3,5], [2,4,0],U).
U = [1, 3, 5, 2, 4, 0].

?- uniao([1,3,5], [2,4,0,3],U).
U = [1, 5, 2, 4, 0, 3].
```

Resposta:

```
uniao([], L, L) :- !.
uniao(L, [], L) :- !.
uniao([P | R], L, U) :-
      member(P, L),
      !,
      uniao(R, L, U).
uniao([P | R], L, [P | U]) :-
      uniao(R, L, U).
```

A segunda cláusula, embora não necessária, evita que o programa percorra a primeira lista quando a segunda lista é vazia.

7.7.13. Defina o predicado `contida/2`, tal que `contida(L1, L2)`, em que `L1` e `L2` são listas de inteiros, significa que todos os elementos de `L1` pertencem a `L2`. Use o operador de corte sempre que tal melhorar o seu programa. O seu programa não deve permitir que o pedido de múltiplas respostas origine respostas erradas. Sugestão: use o falhanço forçado. Por exemplo,

```
?- contida([1,2], [3,4,2]).
false.

?- contida([1,2], [3, 1, 4,2]).
true.
```

Resposta:

```
contida([], _) :- !.
contida([P | R], L2) :-
      member(P, L2),
```

```
      !,
      contida(R, L2).
contida(_, _) :-
      fail.
```

7.7.14. Defina o predicado `disjuntas/2`, tal que `disjuntas(L1,L2)`, em que `L1` e `L2` são listas de inteiros, significa que `L1` e `L2` são disjuntas, isto é, não têm elementos em comum. Por exemplo,

```
?- disjuntas([1,2], [3,4,2]).
false.

?- disjuntas([1,2], [3,4]).
true.
```

Sugestão: use o predicado `membro` definido no livro.

(a) Usando o corte e o falhanço forçado.
(b) Usando a negação.

Resposta:

(a)
```
disjuntas([], _) :- !.
disjuntas([P1 | _], L2) :- membro(P1,L2), !, fail.
disjuntas([_ | R1], L2) :- disjuntas(R1, L2).
```
(b)
```
disjuntas(L1,L2) :- \+ (membro(E,L1),membro(E,L2)).
```

7.7.15. Defina o predicado `lista_perm/2`, tal que `lista_perm(L, LP)`, em que `L` e `LP` são listas, significa que `LP` é uma lista cujos elementos são todas as permutações de `L1` (não interessando a ordem). Por exemplo,

```
?- lista_perm([1, 2, 3], LP).
LP = [[3, 2, 1], [3, 1, 2], [2, 3, 1], [2, 1, 3], [1, 3, 2],
      [1, 2, 3]].
```

Sugestão: use o predicado `membro` definido no livro.

Resposta:

```
lista_perm(L, LP) :- lista_perm(L, [], LP).

lista_perm(L, Acc, LP) :-
    perm(L, Perm),
```

```
    \+ member(Perm, Acc), !,
    lista_perm(L, [Perm|Acc], LP).
lista_perm(_, LP, LP).
```

7.7.16. Considere o seguinte programa:

```
trabalha(P, D) :-
    pessoa(P),
    diaUtil(D),
    \+ temJust(P, D).

temJust(P, D) :-
    doente(P, D),
    \+ constipado(P, D).

diaUtil(D) :- membro(D, [seg, ter, qua, qui, sex]).
pessoa(rui).
pessoa(jaime).
pessoa(joana).
doente(rui, seg).
doente(jaime, qua).
constipado(jaime, qua).
```

(a) Quais as respostas do PROLOG ao objetivo `?- trabalha(P, seg).`?

(b) Qual o objetivo que origina a seguinte resposta do PROLOG ?

```
trabalha(rui,qua)
trabalha(jaime,qua)
trabalha(joana,qua)
false.
```

(c) Quais as respostas do PROLOG ao objetivo
`?- \+ temJust(P, seg).`?

(d) Qual o objetivo que deveria ser dado ao PROLOG para responder à questão “Quem não tem justificação na segunda-feira?"?

Resposta:

(a) `P = jaime`
`P = joana`

(b) `?- trabalha(P, qua), writeln(trabalha(P, qua)), fail.`

(c) `false.`

(d) `?- pessoa(P), \+ temJust(P, seg).`

7.7.17. Considere o seguinte programa:

```
constipado(P) :-
      nariz_entupido(P),
      \+ febre_elevada(P).
engripado(P) :-
      nariz_entupido(P),
      febre_elevada(P).

nariz_entupido(jaime).
nariz_entupido(ana).
febre_elevada(jaime).
febre_elevada(pedro).
```

Quais as respostas do PROLOG aos seguintes objetivos?

(a) `?- constipado(P).`

(b) `?- engripado(P).`

(c) `?- \+ febre_elevada(P).`

(d) `?- nariz_entupido(P), \+ febre_elevada(P).`

Resposta:

(a) `P = ana`

(b) `P = jaime`

(c) `false`

(d) `P = ana`

7.7.18. Considere a seguinte representação em PROLOG para *fbfs* da Lógica Proposicional:

Fbf	Representação
P	`P`
$\alpha \wedge \beta$	`e(`$Rep(\alpha)$`,`$Rep(\beta)$`)`
$\alpha \vee \beta$	`ou(`$Rep(\alpha)$`,`$Rep(\beta)$`)`
$\alpha \rightarrow \beta$	`imp(`$Rep(\alpha)$`,`$Rep(\beta)$`)`
$\neg\alpha$	`nao(`$Rep(\alpha)$`)`

Por exemplo, a *fbf* $(P \land \neg P) \to Q$ é representada pelo termo `imp(e(P,nao(P)),Q))`.

(a) Considere agora a seguinte definição para o predicado `satisfazivel/1`, tal que `satisfazivel(F)`, em que `F` é uma *fbf* , significa que `F` é satisfazível.

```
% o atomo v representa o valor logico verdadeiro
satisfazivel(v).
satisfazivel(e(X,Y)) :- satisfazivel(X),satisfazivel(Y).
satisfazivel(ou(X,_)) :- satisfazivel(X).
satisfazivel(ou(_,Y)) :- satisfazivel(Y).
satisfazivel(imp(X,Y)) :- satisfazivel(ou(nao(X),Y)).
satisfazivel(nao(X)) :- falsificavel(X).
```

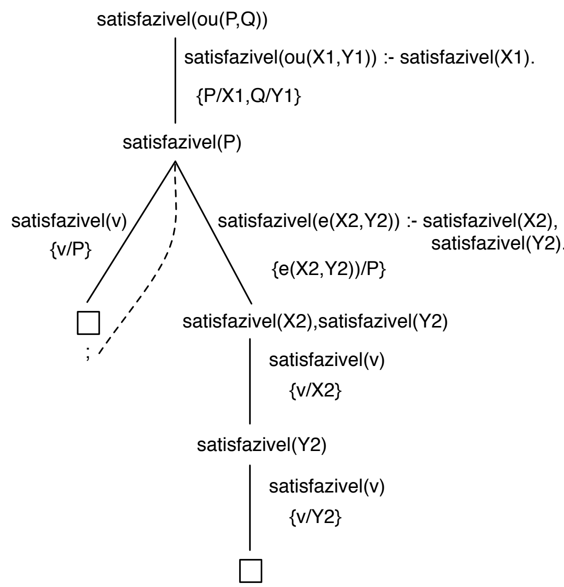
Defina o predicado `falsificavel/1` usado na última cláusula (o significado deste predicado está definido nesta cláusula).

Resposta:

```
% o atomo f representa o valor logico falso
falsificavel(f).
falsificavel(e(X,_)) :- falsificavel(X).
falsificavel(e(_,Y)) :- falsificavel(Y).
falsificavel(ou(X,Y)) :- falsificavel(X),falsificavel(Y).
falsificavel(imp(X,Y)) :- falsificavel(ou(nao(X),Y)).
falsificavel(nao(X)) :- satisfazivel(X).
```

(b) Desenhe a árvore SLD originada pelo objetivo `satisfazivel(ou(P,Q))`, até obter as duas primeiras respostas dadas pelo PROLOG . Diga quais são estas respostas.

Resposta:

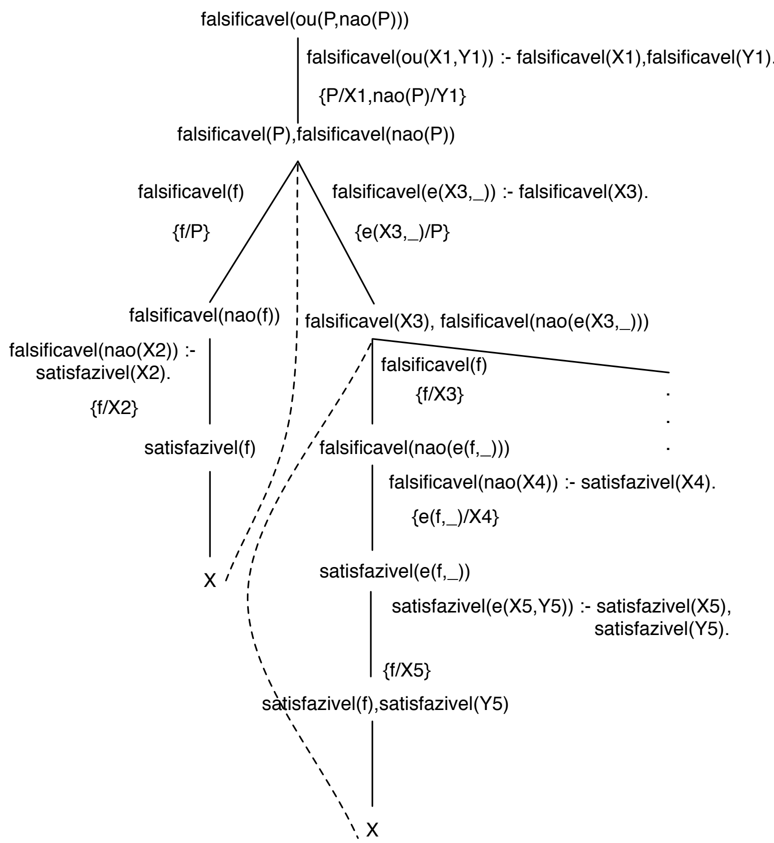


Respostas:

```
P = v ;
P = e(v, v)
```

(c) Desenhe a árvore SLD originada pelo objetivo `falsificavel(ou(P,nao(P)))`.

Resposta:



(d) Altere o programa, de forma a evitar o problema detetado na alínea anterior. Quais são agora as respostas do PROLOG ao objetivo `satisfazivel(ou(P,Q))`?

Resposta:

Para evitar o ciclo infinito gerado pelo objetivo `falsificavel(ou(P,nao(P)))`, a primeira cláusula da definição do predicado `falsificavel` deve ser alterada para `falsificavel(f) :- !.`

Para evitar outros ciclos infinitos, como, por exemplo, o ciclo gerado pelo objetivo `satisfazivel(e(P,nao(P)))`, a primeira cláusula da definição do predicado `satisfazivel` deve também ser alterada para `satisfazivel(v) :- !.`

As respostas do PROLOG ao objetivo `satisfazivel(ou(P,Q))` são agora

```
P = v ;
Q = v.
```

(e) Usando os predicados `satisfazivel` e `falsificavel`, defina os seguintes predicados:

- `tautologia/1` tal que `tautologia(F)`, em que `F` é uma *fbf* , significa que `F` é uma tautologia.
- `contradicao/1` tal que `contradicao(F)`, em que `F` é uma *fbf* , significa que `F` é uma contradição.

Resposta:

```
tautologia(F) :- \+ falsificavel(F).
contradicao(F) :- \+ satisfazivel(F).
```

(f) Defina o predicado `faz_conj/2`, tal que `faz_conj(Forms, Conj)`, em que `Forms` é uma lista de *fbfs* , significa que `Conj` é a conjunção de todos os elementos de `Forms`. Por exemplo,

```
?- faz_conj([P],Conj).
P = Conj.

?- faz_conj([P,Q,R],Conj).
Conj = e(P, e(Q, R)).

?- faz_conj([P,e(nao(R),Q)],Conj).
Conj = e(P, e(nao(R), Q)).
```

Resposta:

```
faz_conj([F],F) :- !.
faz_conj([F | RF],e(F,Conj_RF)) :- faz_conj(RF, Conj_RF).
```

(g) Usando os predicados anteriores, defina o predicado `valido/1`, tal que `valido(Argumento)` significa que o argumento `Argumento` é válido. Um argumento é representado por um par em que o 1º elemento é uma lista contendo as premissas, e o 2º é a conclusão. Por exemplo, o argumento $(\{\neg Q, P \rightarrow Q\}, \neg P)$ é representado por `([nao(Q),imp(P,Q)],nao(P))`. Sugestão: use o teorema da refutação:

Dado um conjunto de *fbfs* Δ e uma *fbf* α, $\Delta \models \alpha$, ou seja, o argumento (Δ, α) é válido se e só se $\Delta \cup \{\neg\alpha\}$ não é satisfazível. Por exemplo,

```
?- valido(([nao(Q),imp(P,Q)],nao(P))).
true.
```

```
?- valido(([nao(Q),imp(P,Q)],P)).
false.
```

Resposta:

```
valido((Prems,Conc)) :-
      faz_conj([nao(Conc) | Prems], Conj),
      contradicao(Conj).
```

Capítulo 8

Bibliografia

[1] João P. Martins *Lógica e Raciocínio*. College Publications / Série de Cadernos de Lógica e Computação, 2014.

[2] Albert Einstein. *Zur Elektrodynamik bewegter Körper*. (German) [*On the electrodynamics of moving bodies*]. Annalen der Physik, 322(10):891–921, 1905.

[3] Knuth: Computers and Typesetting,
`http://www-cs-faculty.stanford.edu/~uno/abcde.html`