Lógica e Computabilidade 2025-1

Hugo Nobrega

Lista de Exercícios 1

As entregas podem ser feitas em duplas, mas lembre-se que
não poderá haver repetição de duplas em listas diferentes!
Entregue todas as questões marcadas com * até
18 de maio às 20:00.

***Questão 1.** Seja X o conjunto das fórmulas da LC construídas apenas usando os conectivos $\wedge, \vee, \rightarrow, \leftrightarrow$ (além dos símbolos proposicional usuais), e seja φ uma fórmula em X. Prove que para todo $k \in \mathbb{N}$, se φ tem k ocorrências de símbolos proposicionais (contando todas as repetições), então o comprimento de φ (considerando φ como uma palavra em um alfabeto, i.e., contando todas as ocorrências de todos os símbolos em φ, incluindo parênteses, etc) é $4k - 3$.

Questão 2.

Definição 1.

1. Uma *árvore ternária* é uma árvore (não vazia) na qual cada nó tem nó máximo 3 filhos;
2. Uma *árvore estritamente ternária* é uma árvore ternária na qual cada nó tem 0 ou 3 filhos.

a. Dê definições recursivas desses dois conceitos.

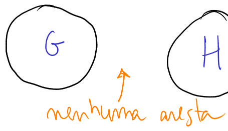
b. Prove que para toda árvore estritamente ternária, a sua quantidade de nós deixa resto 1 quando dividida por 3.

Questão 3.

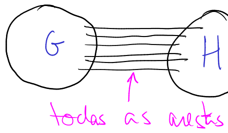
Definição 2. O conceito de *cografo* é definido recursivamente pelas seguintes regras:

B) O grafo com 1 vértice e 0 arestas é um cografo;

R1) Se G e H são cografos, então sua união disjunta também é;

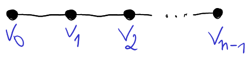


R2) Se G e H são cografos, seu *join* (i.e., o grafo obtido da união disjunta de G e H adicionando *todas* as arestas entre vértices de G e H) também é.



a.

Definição 3. Dado $n \geqslant 1$, denotamos por P_n o *grafo caminho* com n vértices (e portanto $n-1$ arestas):



Dê uma definição recursiva para esses grafos.

b. Prove que P_3 é um cografo.

*** c.**

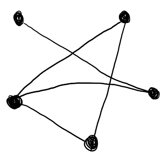
Definição 4. Sejam G e H grafos. Dizemos que H é um *subgrafo induzido* de G se:

- todo vértice de H é um vértice de G, e
- para toda aresta de G, se ambos os vértices dessa aresta estão em H, então a aresta também está em H.

Em outras palavras, H é subgrafo induzido de G sse podemos obter H a partir de G removendo apenas vértices (e todas as arestas incidentes aos vértices removidos).

Prove por indução que nenhum cografo tem P_4 como subgrafo induzido.

*** d.** Prove que o grafo abaixo não é um cografo.



Questão 4. Considere o alfabeto Σ cujos elementos são apenas os símbolos `)` e `(`. Assim, `)()()((` e `()()(())` são palavras nesse alfabeto.

Definição 5. Dizemos que uma palavra deste alfabeto é *balanceada* se, ao começarmos uma contagem com 0 no início da palavra e a percorrermos,

- somando 1 a cada `(` encontrado e
- subtraindo 1 a cada `)` encontrado,

chegaremos ao fim da palavra novamente com a contagem 0, sem nunca termos passado por um valor negativo ao longo do processo.

Intuitivamente, essas são exatamente as palavras obtidas a partir de expressões aritméticas bem formadas, apagando tudo o que não for parêntese.[a] Assim, `()()(())` é balanceada pois `(1+1)+(1+1)+(1+(1+1))` é uma expressão aritmética bem formada.

Dê uma definição recursiva para o conjunto das palavras balanceadas e prove que todas as palavras que são balanceadas de acordo com a sua definição são balanceadas também de acordo com a definição acima. Você não precisa provar a recíproca.

[a] Atenção, essa é apenas uma intuição! A definição de fato é dada pela regra acima.

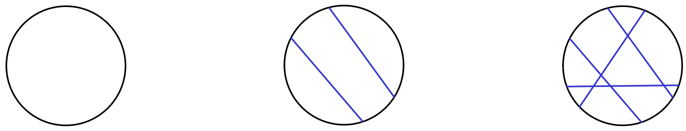
Questão 5.

Definição 6. Vamos definir o conceito de um *mapa de cordas* recursivamente da seguinte forma:

B) Um círculo é um mapa de cordas;

R) Dado um mapa de cordas X, o resultado de adicionar uma nova *corda* à circunferência externa de X (i.e., um segmento de reta ligando dois pontos distintos da circunferência) é também um mapa de cordas.

Veja três exemplos de mapas de linhas na figura abaixo:



Prove que todo mapa de cordas é 2-colorível, i.e., que sempre é possível colorir qualquer mapa de cordas usando apenas duas cores, de forma que regiões vizinhas do mapa recebam cores diferentes. Considere como vizinhas apenas regiões que se encontrem ao longo de algum segmento de reta, i.e., não considere como vizinhas regiões que se encontrem em apenas um ponto (essas podem receber a mesma cor sem problemas).

Questão 6.

Definição 7.

1. Dizemos que um polígono é *convexo* se nenhum segmento de reta ligando lados distintos do polígono encontra algum outro lado (i.e., se todo segmento de reta entre lados distintos do polígono fica completamente no interior do polígono).

2. Uma *diagonal* de um polígono convexo é um segmento de reta que liga dois vértices não-consecutivos do polígono (i.e., é um segmento de reta que liga dois vértices do polígono, desde que esse segmento não seja um dos lados do polígono).

3. Uma *triangulação* de um polígono convexo é um conjunto de triângulos com as propriedades:

   (a) Os vértices dos triângulos são vértices do polígono original;

   (b) As arestas dos triângulos não se cruzam, i.e., só podem se encontrar em vértices ou coincidirem exatamente;

   (c) Os triângulos cobrem exatamente o polígono todo (sem sobra nem falta).

   Em outras palavras, uma triangulação de um polígono é a adição de segmentos de retas entre seus vértices, de forma que esses segmentos não se cruzem, e tal que na nova figura todas as regiões formadas sejam triângulos.

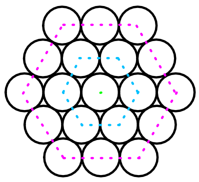
a. Prove por indução que todo polígono convexo com $n \geqslant 3$ vértices possui exatamente $\dfrac{n(n-3)}{2}$ diagonais.

b. Prove por indução que toda triangulação de polígono convexo com $n \geqslant 3$ vértices possui exatamente $n - 2$ triângulos.

c. Prove por indução que, para todo polígono convexo P com $n \geqslant 4$ vértices e toda diagonal d de P, há uma triangulação de P na qual d é a aresta de algum dos triângulos.

Questão 7.

Definição 8. Vamos dizer que um número natural n é *compacto* se, ao organizarmos n moedas de mesmo tamanho formando um "padrão hexagonal", como na figura abaixo, o padrão pode ser completado formando um hexágono externo.[a]



Assim, 1, 7 e 19 são compactos (pela figura acima), mas 9 não é. O nome "compacto" vem do fato desta ser a forma mais eficiente de se cobrir o plano com círculos de mesmo tamanho, fato provado por Lagrange no século XVIII. Esse é um caso particular do problema de *empacotamento de esferas*.

a. Dê uma definição recursiva para o conjunto dos números compactos.

b. 129 é um número compacto?

[a]Vamos forçar um pouco a barra e dizer que uma única moeda também forma um hexágono.

Questão 8. No jogo de xadrez, o cavalo se move em "`L`", i.e., andando 2 casas na horizontal/vertical e 1 casa na vertical/horizontal (respectivamente).

a. Prove que, para qualquer $n \geqslant 4$, um cavalo começando na casa do canto inferior esquerdo de um tabuleiro $n \times n$ consegue chegar a qualquer outra casa do tabuleiro em uma quantidade finita de movimentos.

b. Prove que o mesmo vale mesmo sem assumirmos que o cavalo começa no canto inferior esquerdo do tabuleiro, i.e., prove que para todo $n \geqslant 4$ o cavalo consegue chegar de qualquer casa a qualquer casa de um tabuleiro $n \times n$.

***Questão 9.** Em um torneio de *Winning Eleven* com $n \geqslant 2$ jogadores, cada jogador jogou exatamente 1 vez contra cada outro jogador, e nenhum jogo terminou em empate. Prove que existe um jogador A tal que, para todo outro jogador B:

- B perdeu seu jogo contra A, ou
- B perdeu o jogo contra alguém que por sua vez perdeu seu jogo contra A.

Questão 10. Dê uma definição para a noção "a fórmula resultante de substituir a n-ésima ocorrência de p em φ por ψ", sendo $n \in \mathbb{N}$ com $n \geqslant 1$, p símbolo proposicional, e φ, ψ fórmulas.

Questão 11. Classifique cada uma das fórmulas abaixo em *válida*, *contradição* ou *contingência*. Justifique cada resposta.

a.

```
(p ↔ [¬r]) ∧ (r ∧ p)
∧
├ ↔
│ ├ p
│ └ ¬
│   └ r
└ ∧
  ├ r
  └ p
```

b.

```
(p ↔ [¬r]) ∧ (r ∨ p)
∧
├ ↔
│ ├ p
│ └ ¬
│   └ r
└ ∨
```

```
  ├ r
  └ p
```

c.

```
([{r ∧ q} → {q ↔ q}] ∧ p) ∨ p
∨
├ ∧
│ ├ →
│ │ ├ ∧
│ │ │ ├ r
│ │ │ └ q
│ │ └ ↔
│ │   ├ q
│ │   └ q
│ └ p
└ p
```

d.

```
(¬r) ∨ (q ↔ [q ∧ r])
∨
├ ¬
│ └ r
└ ↔
  ├ q
  └ ∧
    ├ q
    └ r
```

Questão 12. A seguinte definição será usada em alguns dos itens abaixo.

Definição 9. Um *literal* é um símbolo proposicional, ou a negação de um símbolo proposicional (i.e., uma fórmula $(\neg p)$ onde p é um símbolo proposicional).

Em notação Backus–Naur:

$$\langle\text{literal}\rangle := \langle\text{símbolo proposicional}\rangle \mid (\neg\langle\text{símbolo proposicional}\rangle)$$

a. Considere os seguintes conjuntos de fórmulas, definidos por recursão:

Definição 10.

- Uma *cláusula para FND* é um literal, ou a conjunção entre um literal e uma cláusula para FND.
- Uma *fórmula em FND* é uma cláusula para FND, ou a disjunção entre uma cláusula para FND e uma fórmula em FND.

Em notação Backus–Naur:

$$\begin{aligned} \langle\text{cláusula para FND}\rangle &:= \langle\text{literal}\rangle \mid (\langle\text{literal}\rangle \wedge \langle\text{cláusula para FND}\rangle) \\ \langle\text{fórmula em FND}\rangle &:= \langle\text{cláusula para FND}\rangle \\ &\qquad\qquad \mid (\langle\text{cláusula para FND}\rangle \vee \langle\text{fórmula em FND}\rangle) \end{aligned}$$

Prove que toda fórmula da LC é semanticamente equivalente a alguma fórmula em FND.

b. Faça (em pseudocódigo ou em sua linguagem favorita) um programa de computador que converta uma fórmula qualquer da LC para uma equivalente em FND.

c. Para essa questão, primeiramente vamos…

- expandir o alfabeto da lógica proposicional LC, adicionando um novo símbolo: $\perp$ (usualmente chamado "falso" ou "bottom")
- expandir a definição das fórmulas, permitindo um caso base adicional: "$\perp$ é uma fórmula"
- expandir a definição de semântica das fórmulas, com um caso base adicional: "o valor-verdade de $\perp$ é sempre F."

Agora considere a seguinte definição:

Definição 11.

- Uma *fórmula em FNI* é um símbolo proposicional, ou uma fórmula do tipo $(\varphi \to \bot)$, onde φ é uma fórmula em FNI, ou uma fórmula do tipo $(\varphi \to \psi)$, onde φ, ψ são fórmulas em FNI.

Em notação Backus–Naur:

$$\begin{aligned}\langle \text{fórmula em FNI} \rangle := {} & \langle \text{símbolo proposicional} \rangle \\ & \mid (\langle \text{fórmula em FNI} \rangle \to \bot) \\ & \mid (\langle \text{fórmula em FNI} \rangle \to \langle \text{fórmula em FNI} \rangle)\end{aligned}$$

Prove que toda fórmula da LC é semanticamente equivalente a alguma fórmula em FNI.

d. Faça (em pseudocódigo ou em sua linguagem favorita) um programa de computador que converta uma fórmula qualquer da LC para uma equivalente em FNI.

e (Desafio!)**.** Para essa questão, primeiramente vamos...

- expandir o alfabeto da lógica proposicional LC, adicionando um novo símbolo: $\bot$ (usualmente chamado “falso” ou “bottom”)
- expandir a definição das fórmulas, permitindo um caso base adicional: “$\bot$ é uma fórmula”
- expandir a definição de semântica das fórmulas, com um caso base adicional: “o valor-verdade de $\bot$ é sempre F.”

Agora considere a seguinte definição:

Definição 12.

- Um *literal para FNI2* é um símbolo proposicional ou uma fórmula do tipo $(p \to \bot)$, onde p é um símbolo proposicional.

Em notação Backus–Naur:

$$\begin{aligned}\langle\text{literal para FNI2}\rangle := {} & \langle\text{símbolo proposicional}\rangle \\ & \mid (\langle\text{símbolo proposicional}\rangle \to \bot)\end{aligned}$$

- Uma *cláusula para FNI2* é uma fórmula do tipo $(\ell \to \bot)$ onde ℓ é um literal para FNI2, ou uma fórmula do tipo $(\ell \to C)$ onde ℓ é um literal para FNI2 e C é uma cláusula para FNI2.

Em notação Backus–Naur:

$$\begin{aligned}\langle\text{cláusula para FNI2}\rangle := {} & (\langle\text{literal para FNI2}\rangle \to \bot) \\ & \mid (\langle\text{literal para FNI2}\rangle \to \langle\text{cláusula para FNI2}\rangle)\end{aligned}$$

- Uma *fórmula em FNI2* é uma fórmula do tipo $(C \to \bot)$ onde C é uma cláusula para FNI2, ou uma fórmula do tipo $(C \to \varphi)$ onde C é uma cláusula para FNI2 e φ é uma fórmula em FNI2.

Em notação Backus–Naur:

$$\begin{aligned}\langle\text{fórmula em FNI2}\rangle := {} & (\langle\text{cláusula para FNI2}\rangle \to \bot) \\ & \mid (\langle\text{cláusula para FNI2}\rangle \to \langle\text{fórmula em FNI2}\rangle)\end{aligned}$$

Prove que toda fórmula da LC é semanticamente equivalente a alguma fórmula em FNI2.

Questão 13. Mostre que os seguintes conjuntos de conectivos são completos:

a. $\{\neg, \wedge\}$

b. $\{\neg, \vee\}$

c. $\{\neg, \to\}$

d. $\{\neg, \leftrightarrow\}$

e. $\{\rightarrow, \bot\}$, com $\bot$ conforme descrito na Questão 12(c) acima (note que $\bot$ deve ser visto como um conectivo "nulário" ou "0-ário", i.e., um conectivo que se aplica a zero outras fórmulas, e **não** como um símbolo proposicional).

f. {NAND}, sendo NAND o conectivo binário "não ambos", i.e., com a semântica dada pela tabela abaixo:

φ	ψ	(φ NAND ψ)
V	V	F
V	F	V
F	V	V
F	F	V

Por motivos históricos, o conectivo NAND também é conhecido como "Sheffer stroke".

g. {NOR}, sendo NOR o conectivo binário "nenhum dos dois", i.e., com a semântica dada pela tabela abaixo:

φ	ψ	(φ NOR ψ)
V	V	F
V	F	F
F	V	F
F	F	V

*** h.** $\{\bot, \top,$ IF-THEN-ELSE$\}$, onde $\bot$ é o "conectivo 0-ário" sempre falso ("bottom"), $\top$ é o "conectivo 0-ário" sempre verdadeiro ("top"), e IF-THEN-ELSE é o conectivo "condicional ternário", presente em algumas linguagens de programação, cuja semântica é dada pela tabela abaixo.

φ	ψ	β	(IF φ THEN ψ ELSE β)
V	V	V	V
V	V	F	V
V	F	V	F
V	F	F	F
F	V	V	V
F	V	F	F
F	F	V	V
F	F	F	F

Em outras palavras: se φ é verdadeiro, copie ψ; senão, copie β.

Questão 14. Seja X o conjunto das fórmulas construídas apenas usando os conectivos $\wedge, \vee, \rightarrow, \leftrightarrow$ (além dos símbolos proposicional usuais).

a. Escreva a definição recursiva de X.

* **b.** Prove o seguinte teorema:

Teorema. *Seja $\varphi \in X$ e sejam $p_0, p_1, \ldots, p_n$ as subfórmulas atômicas de φ.*

Então em qualquer contexto no qual $p_0, p_1, \ldots, p_n$ são todos verdadeiros, temos que φ também é verdadeiro.

Em outras palavras, temos

$$p_0,\ p_1,\ \ldots p_{n-1},\ p_n \vDash \varphi$$

* **c.** Prove que o conjunto $C = \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ de conectivos não é completo.

Questão 15. Considere a seguinte sequência infinita de fórmulas definidas por recursão (aqui o parâmetro n é sempre natural):

$$\varphi_n = \begin{cases} (p \rightarrow q), & \text{se } n = 0 \\ (\varphi_{n-1} \rightarrow p), & \text{se } n > 0. \end{cases}$$

Para quais valores de $n \in \mathbb{N}$ temos que φ_n é uma tautologia (i.e., uma fórmula válida)? Justifique sua resposta.

Questão 16. Nesta questão você não pode usar os teoremas de corretude e completude vistos em sala.

Sejam Σ, Δ conjuntos de fórmulas da LC e sejam φ, ψ fórmulas da LC.

Prove cada um dos itens abaixo.

a. Se $\varphi \in \Sigma$, então $\Sigma \vdash \varphi$.

b. Se $\varphi : \star$ e $\varphi : \bar{\star}$ ambos estão em Σ para algum φ, então $\Sigma \vdash \psi$ para qualquer ψ.

c. Se $\Sigma \vdash \varphi$, então também $\Sigma \cup \Delta \vdash \varphi$.

d. $\Sigma, \varphi \vdash \psi$ se, e somente se, $\Sigma \vdash \varphi \rightarrow \psi$.

Questão 17. Prove os itens abaixo usando árvores de avaliação.

* **a.** $p \vee q,\ p \to r,\ q \to r \vdash r$

b. $p \to (q \to r) \vdash (p \wedge q) \to r$

* **c.** $p \leftrightarrow (q \leftrightarrow r) \nvdash [p \wedge (q \wedge r)] \vee [(\neg p) \wedge \langle(\neg q) \wedge (\neg r)\rangle]$

d. $p \to (q \to r) \nvdash (p \to q) \to r$

Questão 18. Como vimos, além de $\{\neg, \wedge, \vee, \to, \leftrightarrow\}$ há outros conjuntos de conectivos completos para a LC. Em cada item abaixo, dê regras de manipulação de árvores de avaliação que correspondam aos conectivos listados e que sejam corretas, i.e., que preservem satisfabilidade das árvores.

a. $\{\text{NAND}\}$

b. $\{\text{NOR}\}$

* **c.** $\{\bot, \top, \text{IF-THEN-ELSE}\}$

Questão 19. Nesta questão vamos provar a completude do nosso sistema de provas para a LC, i.e., vamos provar que para qualquer conjunto Σ de fórmulas da LC e qualquer fórmula φ da LC temos

$$\Sigma \vDash \varphi \quad \Rightarrow \quad \Sigma \vdash \varphi.$$

Na verdade, provaremos a contrapositiva dessa implicação.

Definição. Seja A uma árvore de avaliação e seja r um ramo de A. Dizemos que r é *saturado* se, para qualquer julgamento composto $\varphi : \star$ que ocorra como rótulo de um nó de r, temos algum dos casos abaixo:

- Caso $\neg$. Se $\varphi = (\neg\psi)$, então em r há algum nó com rótulo $\psi : \bar{\star}$.
- Caso $\wedge : V$. Se $\varphi = (\psi \wedge \beta)$ e $\star = V$, então em r há nós com rótulos $\psi : V$ e $\beta : V$.
- Caso $\wedge : F$. Se $\varphi = (\psi \wedge \beta)$ e $\star = F$, então em r há algum nó com rótulo $\psi : F$ ou algum nó com rótulo $\beta : F$.
- Caso $\vee : V$. Se $\varphi = (\psi \vee \beta)$ e $\star = V$, então em r há algum nó com rótulo $\psi : V$ ou algum nó com rótulo $\beta : V$.
- Caso $\vee : F$. Se $\varphi = (\psi \vee \beta)$ e $\star = F$, então em r há nós com rótulos $\psi : F$ e $\beta : F$.
- Caso $\rightarrow: V$. Se $\varphi = (\psi \rightarrow \beta)$ e $\star = V$, então em r há algum nó com rótulo $\psi : F$ ou algum nó com rótulo $\beta : V$.
- Caso $\rightarrow: F$. Se $\varphi = (\psi \rightarrow \beta)$ e $\star = F$, então em r há nós com rótulos $\psi : V$ e $\beta : F$.
- Caso $\leftrightarrow: V$. Se $\varphi = (\psi \leftrightarrow \beta)$ e $\star = V$, então em r há nós com rótulos $\psi : V$ e $\beta : V$, ou nós com rótulos $\psi : F$ e $\beta : F$.
- Caso $\leftrightarrow: F$. Se $\varphi = (\psi \leftrightarrow \beta)$ e $\star = F$, então em r há nós com rótulos $\psi : V$ e $\beta : F$, ou nós com rótulos $\psi : F$ e $\beta : V$.

* **a.** Seja r um ramo aberto e saturado de uma árvore de avaliação A. Seja c_r o contexto definido por:

$$c_r(p) = \begin{cases} V, & \text{se } p : V \text{ é o rótulo de algum nó de } r \\ F, & \text{se } p : F \text{ é o rótulo de algum nó de } r. \end{cases}$$

Mostre que este contexto satisfaz r (i.e., satisfaz todos os julgamentos que aparecem como rótulos dos nós em r). (*Dica*: em “quem” você pode fazer indução?)

b. Por que precisamos exigir que o ramo r seja aberto e saturado, no item anterior?

* **c.** Prove o teorema da completude: se $\Sigma \nvdash \varphi$ (i.e., nenhuma árvore de avaliação para $\Gamma = \{\sigma : V \; ; \; \sigma \in \Sigma\} \cup \{\varphi : F\}$ é fechada), então $\Sigma \nvDash \varphi$ (i.e., algum contexto torna todas as fórmulas em Σ verdadeiras mas φ falsa). Para simplificar, você pode assumir que Σ é finito.