Programación y Algorítmica Avanzadas

Curso 2025-26, cuatrimestre de primavera

José Luis Balcázar, Jordi Delgado

Departament de Ciències de la Computació, UPC

Recopilación de problemas para el segundo parcial

Este texto recopila enunciados de problemas sobre gramáticas incontextuales, funciones recursivas parciales, decidibilidad, complejidad y λ-cálculo; muchos se han ido proponiendo en las sucesivas clases. Otros han aparecido en exámenes de años anteriores: una mención a un curso entre corchetes indica que ese problema apareció en un examen de ese curso.

Al resolver cada ejercicio, está permitido (y alentado explícitamente) el uso de los anteriores siempre que sea procedente, así como, cuando sea factible, comprobar la solución propia programándola en `PReFScript`.

Representa booleanos con los valores 0 y 1 para operar con ellos o como resultados de comparaciones; pero procura que todas las operaciones booleanas se puedan aplicar sobre funciones que representen `cierto` con cualquier valor no nulo.

1. Consideramos la gramática con un único símbolo no terminal `S` y símbolos terminales `+`, `*`, `1` y `2`, y reglas `S` $\to$ `1`, `S` $\to$ `2`, `S` $\to$ `S + S`, `S` $\to$ `S * S`. Explica cómo derivar las palabras: `1 * 2`, `1 + 2 * 2` y `1 + 2 * 2 + 1`. Encuentra más ejemplos de palabras generadas.

2. Algunas de esas palabras se pueden derivar de más de una manera. Encuentra derivaciones alternativas.

3. ¿Cómo puedes argumentar que la palabra `1 + * 2` no se puede generar con esa gramática?

4. Con dos símbolos no terminales `S` y `A` y los símbolos terminales `6` y `7`, consideramos la gramática con dos reglas `S` $\to$ `6 S A A`, `S` $\to$ `7` y `A` $\to$ `7`. ¿Cómo puedes describir las palabras generadas por esta gramática?

5. [2023-24] Escriu una gramàtica incontextual per generar el següent llenguatge L sobre l'alfabet $\Sigma = \{a, b\}$, tot raonant acuradament que la gramàtica que proposes genera efectivament el llenguatge L:

   $L = \{a^i b^j | i \geq j\}$.

6. Desarrollando ejemplos y con barra libre sobre el uso de `cantorpairs`, asegúrate de entender *todas* las funciones recursivas parciales básicas.

7. Construye, a partir de las funciones básicas y mediante composición y formación de pares, la función que hace corresponder a x el valor <x.1> (la podríamos llamar piggyback_1).
8. Igualmente con piggyfront_1 que hace corresponder a x el valor <1.x>.
9. E igualmente con piggyback_0 y piggyfront_0.
10. Construye, a partir de las funciones básicas y mediante composición y formación de pares, la función *anterior* (modificado): x - 1 para x > 0 y 0 para x = 0.
11. Explica por qué cualquier función constante está entre las funciones recursivas parciales.
12. Explica por qué está entre las funciones recursivas parciales, para cada natural i, la función que hace corresponder a x el valor <x.i>; idem para <i.x>.
13. ¿Cómo se construyen las dos proyecciones π^L y π^R a partir de las funciones básicas?
14. ¿Y la función que a x le hace corresponder <x.x>?
15. ¿Y la función que a x le hace corresponder x^2?
16. ¿Y la que a <x.y> le hace corresponder <y.x>?
17. ¿Y la que a <x.y> le hace corresponder <x+1.y>?
18. Construye, a partir de las funciones básicas y mediante composición y formación de pares, la función signo: sg(x) = 0 if x == 0 else 1 (será útil, entre muchos otros casos, para una de las posibles maneras de construir la operación de disyunción).
19. Construye, a partir de las funciones básicas y mediante composición y formación de pares, las operaciones de negación, conjunción y disyunción.
20. Construye, a partir de las funciones básicas y mediante composición y formación de pares, las funciones que, dado <x.y>, comparan x con y según criterios de orden o de igualdad y desigualdad.
21. [2022-23] La funció binària nor entre predicats és certa exactament quan tots dos arguments son falsos. Argumenta que aquesta funció és recursiva parcial (i, de fet, total).
22. [2023-24] Construeix com a funció recursiva parcial la funció que, per l'argument x, retorna el par <x.x-1> (on cal recordar que la diferència és modificada i, per tant, 0 - 1 es considera 0 doncs no treballem amb negatius).
23. [2023-24] Construeix com a funció recursiva parcial la funció que retorna el valor 2*x+1 per l'argument x.

24. [2024-25] Construye la función que, dado `x`, encuentra el primer valor `z` que sea mayor que `x`. (Obviamente, se trata de `x+1`, pero se desea una solución por minimización.)

25. Argumenta que la funció següent és recursiva: la funció rep $x = \langle i.j \rangle$ i retorna com a imatge el quadrat de *la diferència usual dels enters* entre ells, $(i-j)^2$. *Ull viu*: el quadrat fa que el resultat sigui no negatiu encara que $i-j$ ho sigui; *no disposem* d'aquesta operació de diferència directament ja que $i-j$ sí pot ser negatiu.

26. [2023-24] Construeix com a funció recursiva parcial el predicat que indica si l'argument p és parell.

27. Construye la función que, dados `x` e `y`, retorna el cociente de dividir `x` entre `y`; también para el resto; y, de nuevo, para obtener ambos a la vez, formando un par, pero usando *una única* minimización.

28. [2023-24] Construeix com a funció recursiva parcial la funció que, per l'argument x, retorna el menor valor y tal que el seu quadrat $y * y$ es més gran o igual que x: $y * y \geq x$.

29. [2023-24] Construeix com a funció recursiva parcial el predicat que indica si l'argument `x` és un quadrat perfecte (és a dir, coincideix amb el quadrat d'un altre número natural).

30. [2022-23] Argumenta que la funció que per cada número natural retorna la aproximació entera per defecte de la seva arrel quadrada és recursiva parcial.

31. [2022-23] Argumenta que la funció que a cada número natural retorna la aproximació entera per defecte de la seva arrel cúbica és recursiva parcial.

32. [2023-24] Construeix com a funció recursiva parcial la funció que, donat l'argument $\langle p.q.r \rangle$, retorna el número de solucions de l'equació de segon grau $px^2 + qx + r$.

33. Sea `A` un conjunto de naturales, y sean `f` y `g` funciones totales. Combinándolas con las básicas, y con la función característica de `A`, mediante composición y formación de pares, explica cómo obtener la función `h` donde `h(x)`vale:

    `f(x) si x` $\in$ `A`

    `g(x) si x` $\notin$ `A`

34. Sea P un predicado recursivo. Justifica que también lo son los predicados que se obtienen a partir de él por cuantificación existencial acotada y por cuantificación universal acotada:

    $P'(\langle x.t \rangle) = 1 \iff \exists y (y \leq t \wedge P(\langle x.y \rangle) = 1)$,

    $P''(\langle x.t \rangle) = 1 \iff \forall y (y \leq t \Rightarrow P(\langle x.y \rangle) = 1)$.

35. Si f es (total y) biyectiva, su inversa está bien definida y es total. Si f es recursiva, ¿lo es f^{-1}?

36. Si f es (total e) inyectiva, su inversa está bien definida cuando está definida, pero no necesariamente es total. Si f es además recursiva, ¿es f^{-1} recursiva parcial? ¿Cómo podemos enunciar algo similar si f no es inyectiva (pero es total)?

37. [2023-24] Argumenta que el conjunt $Z = \{i \mid \forall p(\phi_i(p) > p)\}$ és indecidible.

38. [2023-24] Argumenta que el conjunt $\{i \mid$ algun dels valors 0, 1, 2 és a Dom $\phi_i\}$ no és decidible.

39. [2023-24] Construeix una reducció de K al conjunt $\{i \mid$ Dom ϕ_i no és buit$\}$.

40. Argumenteu (o demostreu!) que $A \leq_m B \Leftrightarrow \overline{A} \leq_m \overline{B}$

41. Demostra la transitivitat de $\leq_m$. És a dir, si $A \leq_m B$ i $B \leq_m C$, aleshores $A \leq_m C$.

42. [2024-25] Sigui $f : \mathbb{N} \to \mathbb{N}$ una funció qualsevol (no necessàriament una funció recursiva parcial). Definim el conjunt $A_f = \{i \mid \phi_i = f\}$. Argumenteu que, o bé $A_f = \emptyset$ o bé A_f és infinit.

43. [2024-25] Sigui el conjunt $A = \{x \mid \exists y \geq x \;\; \phi_y$ és constant$\}$. Què podem dir d'aquest conjunt? És decidible?

44. [2024-25] Sigui el conjunt $A = \{x \mid \phi_x = id\}$. És A decidible?

45. [2022-23] Considerem el conjunt $A = \{i \,|\, \phi_i(42) \downarrow\}$, és a dir, els índexos i tals que el valor 42 és al domini de la funció ϕ_i. Argumenta que és indecidible.

46. [2024-25] Sigui el conjunt $A = \{i|\phi_i(y) = y$, per a infinits $y\}$. Argumenta que A és indecidible.

47. [2024-25] Siguin L i M conjunts a co-NP. Argumenta (o, si pots, demostra) que la seva intersecció també està a co-NP: $L \cap M \in$ co-NP.

48. [2024-25] Argumenteu (o demostreu!) que $A \leq_m^p B \Leftrightarrow \overline{A} \leq_m^p \overline{B}$

49. [2024-25] Demostra la transitivitat de $\leq_m^p$. És a dir, si $A \leq_m^p B$ i $B \leq_m^p C$, aleshores $A \leq_m^p C$.

50. Inventa algunes fórmules en forma clausal, construeix el graf corresponent a la reducció de SAT a CLIQUE i identifica quina presència o absència de cliques és relevant per la reducció en cada cas.

51. Escriu fórmules en forma clausal que expressin $y = x_1 \wedge x_2$ y $y = x_1 \vee x_2$, on x_1, x_2 e y son variables proposicionals (és a dir, booleanes). Tot fent-les servir, troba com transformar una fórmula proposicional arbitraria en una altra, de manera que (a) la fórmula obtenida estigui en forma clausal i (b) si una de les dues és satisfactible, llavors l'altra també. Pista: per

De Morgan, podem suposar que totes las negacions s'aplican únicament a variables; afegeix una variable per cada subfórmula i fes servir les clàusules indicades al principi d'aquest exercici per relacionar les noves variables entre elles.

52. [2022-23] Considerem els següents conjunts:

$$A = \{(M, p_0, \ldots, p_{n-1}) \mid \exists J \subseteq \{0, \ldots, n-1\} \sum_{i \in J} p_i = M\}$$
$$B = \{(x_0, \ldots, x_{m-1}) \mid \exists J \subseteq \{0, \ldots, m-1\} \sum_{i \in J} x_i = \sum_{i \notin J} x_i\}$$

on M, els p_i i els x_i són tots enters no negatius. Podem veure que el primer és la variant de la Motxilla que vam anomenar a classe *Subset Sum*, només amb pesos i tot requerint emplenar exactament la motxilla. El segon es coneix com a *Partició*.
Argumenta que $A \leq_m^p B$: per fer això, indica com construir valors x_i per B, a partir de valors M i p_i per A, de manera que $(M, p_0, \ldots, p_{n-1})$ és a A si i només si $(x_0, \ldots, x_{m-1})$ és a B, sense oblidar argumentar que es poden construir en temps polinòmic. Qué en pots deduir immediatament sobre B? Argumenta que la reducció en el sentit contrari també existeix, però, *sense construir-la explícitament.* En dedueixes quelcom?

53. [2023-24] Recordem que hem argumentat a classe que el problema *Subset Sum* és NP-complet: donats sumands $s_i \in \mathbb{N}$ i objectiu $t \in \mathbb{N}$, podem trobar un subconjunt J tal que $\sum_{i \in J} s_i = t$? És, en efecte, semblant al que hem estudiat sota el nom de *Problema de la Motxilla.* Argumenta, el més convincentment que puguis, que el Problema de la Motxilla amb una fita superior pel pes total i una fita inferior pel valor total, que hem treballat a classe, és NP-complet.

54. [2024-25] Sabent que `pair` = $\lambda x.\lambda y.\lambda p.pxy$, i `first` = $\lambda p.p(\lambda x.\lambda y.x)$, calcula (la forma normal de) `first pair 2 1`. No cal saber la codificació en λ-càlcul del 2 ni de l'1 per fer aquest problema.

55. [2024-25] Recordem que, en el λ-càlcul, `TRUE` es representa per λxy.x i `FALSE` per λxy.y. Argumenta per què podem definir `OR` p q com p p q (o el que és el mateix, `OR` = λp.λq.p p q)

56. [2022-23] Sabent que en el λ-càlcul `TRUE` es representa per λxy.x i `FALSE` per λxy.y, argumenta en detall que λab.ab(λxy.y) és la implementació de la funció `AND`.

57. [2023-24] Siguin les definicions en λ-càlcul de `TRUE` λ`xy.x` i `FALSE` λ`xy.y`, i definim `XOR` $\equiv$ λ`xy.x(y FALSE TRUE )y`. Demostra que `XOR TRUE TRUE = FALSE` (recorda que l'aplicació associa a l'esquerra). Demostra que `XOR FALSE TRUE = TRUE`. Verifica també els dos casos restants.

58. [2024-25] A classe vam veure el combinador de punt fix `Y` descobert pel lògic Haskell B. Curry, i vam veure que per a qualsevol λ-expressió `R` es verifica que `Y R = R (Y R)`. Alan Turing també va descobrir un combinador de

punt fix $\Theta = (\lambda xy.y\ (x\ x\ y))(\lambda xy.y\ (x\ x\ y))$. Demostreu que per a qualsevol λ-expressió `R` es verifica que `Θ R = R (Θ R)`

59. [2023-24] A classe vam veure que la definició en Python del combinador de punt fix `Y`:

    ```
    Y = (lambda f: (lambda x: f(x(x))) (lambda x: f(x(x))))
    ```

    no anava bé, generava errors. Explica breument quin és el problema i com es resol.