FUNDAMENTOS DE TEORÍA DE LA COMPUTACIÓN
Ejemplo de Examen (con las respuestas)

Comentario: Antes de leer las respuestas, tratar de resolver las preguntas. No detenerse tanto en cada pregunta, para tratar de resolver la mayor cantidad posible de ellas en 2 horas.

Ejercicio 1. ¿Qué postula la Tesis de Church-Turing?
Respuesta. Todo lo computable puede ser resuelto por una máquina de Turing.

Ejercicio 2. El Problema de la Partición consiste en determinar, dado un conjunto de números naturales $K = \{x_1, x_2, \ldots, x_n\}$, si existe una partición de K en dos subconjuntos K_1 y K_2 tal que la suma de los números de K_1 es igual a la suma de los números de K_2. Especificar el lenguaje que representa el problema.
Respuesta. PARTICION = { $K = \{x_1, \ldots, x_n$, con x_i número natural para todo i} | $\exists$ una partición de K de la forma $K_1 = \{y_1, \ldots, y_a\}$ y $K_2 = \{z_1, \ldots, z_b\}$ que cumple $\Sigma_{i=1,a}\ y_i = \Sigma_{i=1,b}\ z_i$ }.

Ejercicio 3. Se prueba que el modelo de máquinas de Turing (o MT) con cintas semi-infinitas (finitas a izquierda e infinitas a derecha) es equivalente al modelo de MT tratado en clase. ¿Qué significa que ambos modelos de MT son equivalentes?
Respuesta. Dos MT son equivalentes si aceptan el mismo lenguaje. Dos modelos de MT son equivalentes si para toda MT de un modelo existe una MT equivalente en el otro modelo. En este caso, dada una MT M_1 con cintas semi-infinitas, existe una MT equivalente M_2 con cintas infinitas, y dada una MT M_3 con cintas infinitas, existe una MT equivalente M_4 con cintas semi-infinitas.

Ejercicio 4. Construir una MT que acepta el lenguaje de las cadenas de 1 y 0 que empiezan con la subcadena 10. *Comentario: Hay que definir Q, Σ, Γ, δ y q_0.*
Respuesta.
Idea general.
La MT M debe rechazar si el 1er símbolo no es un 1, luego debe rechazar si el 2do símbolo no es un 0, y luego hasta el final debe rechazar si los símbolos siguientes no son 1 ni 0.
Construcción de la MT M.
$M = (Q, \Sigma, \Gamma, \delta, q_0, q_A, q_R)$ tal que:
- $Q = \{q_0, q_1, q_2\}$. q_0 es el estado inicial, que espera un 1. q_1 es el estado siguiente, que espera un 0. q_2 es el estado siguiente al estado q_1, que espera un 1, un 0 o un B.
- $\Sigma = \{1, 0\}$.
- $\Gamma = \{1, 0\}$.
- $q_0 = q_0$.
- La función de transición δ se define de la siguiente manera:
  1. $\delta(q_0, 1) = (q_1, 1, R)$
  2. $\delta(q_1, 0) = (q_2, 0, R)$
  3. $\delta(q_2, 1) = (q_2, 1, R)$
  4. $\delta(q_2, 0) = (q_2, 0, R)$
  5. $\delta(q_2, B) = (q_A, B, S)$

Ejercicio 5. Responder:
a) ¿Por qué R $\subseteq$ RE?
b) ¿Por qué RE $\subseteq \mathfrak{L}$?
c) ¿Por qué R $\subseteq$ CO-RE?
d) Dar un ejemplo de lenguaje en RE – R y un ejemplo de lenguaje en CO-RE – R.

Respuesta.
a) Que $L \in R$ significa que existe una MT que lo acepta y para siempre. Que $L \in RE$ significa que existe una MT que lo acepta (parando siempre o no). Por lo tanto, por definición, pertenecer a R es un caso particular de pertenecer a RE.
b) Todo lenguaje pertenece a $\mathfrak{L}$, en particular los de RE.
c) $L \in R \longrightarrow L^C \in R$ (probamos en clase que R es cerrada con respecto al complemento) $\longrightarrow L^C \in RE$ (porque $R \subseteq RE$) $\longrightarrow L \in$ CO-RE (por definición de CO-RE).
d) HP = {(<M>,w) | M para a partir de w} $\in$ RE – R, y $HP^C \in$ CO-RE – R.

Ejercicio 6. Sean dos lenguajes L_1 y L_2 de RE, aceptados por MT M_1 y M_2, respectivamente, y sea la siguiente MT M, que dado un input w, hace:

1. Ejecuta M_1, y si M_1 rechaza (parando o no), entonces rechaza (parando o no).
2. Ejecuta M_2, y responde como M_2.

Se cumple que la MT M acepta la intersección entre L_1 y L_2., es decir: $L_1 \cap L_2 = L(M)$. Justificar.
Respuesta.
La igualdad $L_1 \cap L_2 = L(M)$ se prueba por doble inclusión de conjuntos, que a su vez puede resolverse técnicamente usando el $\leftrightarrow$ de la lógica:
$w \in L_1 \cap L_2 \leftrightarrow w \in L_1$ y $w \in L_2 \leftrightarrow M_1$ acepta w y M_2 acepta $w \leftrightarrow M$ acepta $w \leftrightarrow w \in L(M)$.

Ejercicio 7. Sea el lenguaje $D = \{w_i \mid$ la MT M_i para desde $w_i\}$, considerando el orden canónico. Probar que $D \in$ RE. *Ayuda: Construir una MT que acepta el lenguaje D.*
Respuesta. Dado un input w, la MT M que acepta el lenguaje D hace lo siguiente:
1. Obtiene el subíndice i tal que w = wi según el orden canónico.
2. Obtiene la MT M_i según el orden canónico.
3. Ejecuta M_i a partir de w y acepta sii M_i para.

Ejercicio 8. Sea f una función de reducción de un lenguaje L_1 a un lenguaje L_2. Responder:
a) ¿Qué significa que f es una función de reducción de L_1 a un lenguaje L_2?
b) ¿A qué clase pertenece L_1 si $L_2 \in$ R? Justificar.
c) ¿A qué clase no pertenece L_2 si $L_1 \notin$ RE? Justificar.

Respuesta.
a) f es una función con las siguientes dos características: (1) es total computable, (2) para toda cadena w se cumple que $w \in L_1 \leftrightarrow f(w) \in L_2$.
b) A la clase R, por lo siguiente: probamos en clase que si existe una reducción de L_1 a L_2, es decir si L_1 α L_2, entonces se cumple: $L_2 \in R \rightarrow L_1 \in R$.
c) A la clase RE, por lo siguiente: probamos en clase que si L_1 α L_2, entonces: $L_2 \in RE \rightarrow L_1 \in RE$, o lo que es lo mismo, empleando la forma contra-recíproca: $L_1 \notin RE \rightarrow L_2 \notin RE$.

Ejercicio 9. ¿Qué diferencias existen entre las MT generales, los autómatas finitos y los autómatas con pila, en los siguientes aspectos?:
a) Movimientos del cabezal sobre la cinta de input.
b) Escritura sobre la cinta de input y sobre las otras cintas.
c) Cuándo paran.
d) Cuándo aceptan y cuándo rechazan.

Respuesta.
a) Las MT se pueden mover sobre la cinta de input a izquierda, derecha, o quedarse en el mismo lugar. En cambio los AF y los AP sólo se pueden mover a derecha.
b) Las MT pueden escribir sobre cualquier cinta. Los AF no pueden escribir sobre su única cinta. Los AP no pueden escribir sobre su cinta de input pero sí sobre su cinta que se comporta como una pila.
c) Las MT paran cuando alcanzan el estado q_A o el estado q_R. Los AF y los AP para cuando alcanzan el blanco en la cinta de input.
d) Las MT aceptan cuando alcanzan el estado q_A y rechazan cuando alcanzan el estado q_R o no paran. Los AF aceptan (rechazan) cuando paran en un estado perteneciente (no perteneciente) al conjunto de estados finales F. Los AP aceptan (rechazan) cuando paran y la pila está (no está) vacía.

Ejercicio 10. Sea la clase temporal TIME(T(n)). ¿Qué significa que un lenguaje L pertenece a TIME(T(n))?
Respuesta. Que existe una MT determinística M que acepta L y para siempre, tal que para todo input w de tamaño |w| = n, M ejecutada a partir de w hace a lo sumo O(T(n)) pasos.

Ejercicio 11. ¿Qué postula la Tesis Fuerte de Church-Turing?
Respuesta. El *overhead* o retardo temporal entre dos modelos computacionales razonables es a lo sumo polinomial. P.ej., vimos en clase que si una MT determinística con K cintas trabaja en tiempo O(T(n)), entonces una MT determinística equivalente con 1 cinta trabaja en tiempo $O(T^2(n))$. Otros ejemplos clásicos de modelos razonables tales que el pasaje de unos a otros es polinomial en términos de tiempo son: MT y máquinas RAM, MT y funciones recursivas parciales, MT y lambda cálculo, MT y autómatas celulares, etc. Un modelo no razonable de MT lo

constituyen las MT no determinísticas. Lo mismo, una representación numérica no razonable para los inputs es la notación unaria.

Ejercicio 12. Ya definimos antes el Problema de la Partición: consiste en determinar, dado un conjunto de números naturales $K = \{x_1, x_2, \ldots, x_n\}$, si existe una partición de K en dos subconjuntos K_1 y K_2 tal que la suma de los números de K_1 es igual a la suma de los números de K_2. Probar que el problema está en NP. *Comentario: la suma de dos números tarda tiempo poly(n)*.
Respuesta. Hay que construir una MT no determinística M que acepte el lenguaje PARTICION (ver el ejercicio 2) en tiempo polinomial. Dado un input w, M hace:
1. Si w no tiene la forma $K = \{x_1, x_2, \ldots, x_n\}$, siendo los x_i nros naturales, rechaza.
2. Genera no determinísticamente a partir de K una partición K_1 y K_2.
3. Acepta sii la suma de los números de K_1 es igual a la suma de los números de K_2.

Claramente M acepta PARTICION. Veamos que el tiempo de ejecución de cada uno de los pasos es polinomial, y entonces también el tiempo de ejecución de M:
Paso 1. Puede consistir en recorrer primero el input para detectar si hay símbolos distintos a los esperados (dígitos y separadores), y luego volver a recorrerlo para validar si hay un nro que se repite. Tiempo poly(n).
Paso 2. Puede consistir en generar no determinísticamente una permutación de K, con un separador entre un par de nros consecutivos, para así obtener una partición particular K_1 y K_2. Tiempo poly(n).
Paso 3. La longitud de todo nro es a lo sumo n y la suma de 2 nros de longitud a lo sumo n tarda tiempo poly(n), por lo que también tarda tiempo poly(n) la suma de los nros de K_1 y la suma de los nros de K_2. Por otro lado, comparar 2 nros de tamaño a lo sumo n tarda tiempo O(n). Tiempo total: poly(n).

Ejercicio 13. Probar que si se cumple $L_1 \alpha_P L_2$, $L_2 \alpha_P L_1$ y $L_1 \in$ NPC, entonces $L_2 \in$ NPC. *Ayuda: usar las definiciones de reducción polinomial y de problema NP-completo*.
Respuesta. Hay que probar que $L_2 \in$ NPC, es decir: (1) $L_2 \in$ NP, y (2) Todo lenguaje de NP se reduce polinomialmente a L_2:
(1) $L_2 \in$ NP porque $L_2 \alpha_P L_1$ y $L_1 \in$ NP (ya que L_1 es NP-completo).
(2) Sea L algún lenguaje de NP. L se reduce polinomialmente a L_1 porque L_1 es NP-completo. Y como $L_1 \alpha_P L_2$, entonces por transitividad de las reducciones polinomiales se cumple $L \alpha_P L_2$.

Ejercicio 14. Se indicó en clase que si existe un lenguaje NP-completo en CO-NP, entonces NP = CO-NP, y se probó sólo la inclusión NP $\subseteq$ CO-NP. La prueba de la inclusión inversa, es decir CO-NP $\subseteq$ NP, podría hacerse de la siguiente manera:
i. Existe por hipótesis un lenguaje L_1 NP-completo que está en CO-NP.
ii. Sea L_2 algún lenguaje de CO-NP. Hay que probar que L_2 está en NP.
iii. Se cumple que $L_2{}^C \in$ NP.
iv. Entonces $L_2{}^C \alpha_P L_1$.
v. Entonces $L_2 \alpha_P L_1{}^C$.
vi. Entonces $L_2 \in$ NP.

Explicar cómo se obtienen los pasos iii a vi, uno a uno.
Respuesta.
iii. Porque $L_2 \in$ CO-NP.
iv. Porque $L_2{}^C \in$ NP y L_1 es NP-completo.
v. Por propiedad de las reducciones polinomiales (y no polinomiales): $L_1 \alpha_P L_2 \leftrightarrow L_1{}^C \alpha_P L_2{}^C$. Por lo tanto, si $L_2{}^C \alpha_P L_1$, entonces $L_2{}^{CC} \alpha_P L_1{}^C$, o lo que es lo mismo: $L_2 \alpha_P L_1{}^C$.
vi. Porque $L_2 \alpha_P L_1{}^C$ y $L_1{}^C \in$ NP (ya que $L_1 \in$ CO-NP).

Ejercicio 15. Explicar por qué si una MT M_1 trabaja en espacio SPACE(S(n)), entonces existe una MT M_2 equivalente que trabaja en el mismo espacio y para siempre.
Respuesta. Si una MT determinística M_1 trabaja en espacio SPACE(S(n)), es decir en espacio acotado, entonces se puede detectar que loopea dado que se puede calcular la cantidad de configuraciones distintas N por las que puede pasar, y así construir una MT M_2 equivalente pero que pare siempre: M_2 simula M_1 y además en cada paso suma 1 a un contador inicializado en 0 para chequear si llegó a N, en cuyo caso rechaza porque significa que M_1 repitió una configuración, es decir que entró en loop. M_2 trabaja en espacio O(S(n)) porque simula M_1, y además $N = c^{S(n)}$, con c constante, representado en base 2 ocupa también O(S(n)) celdas.