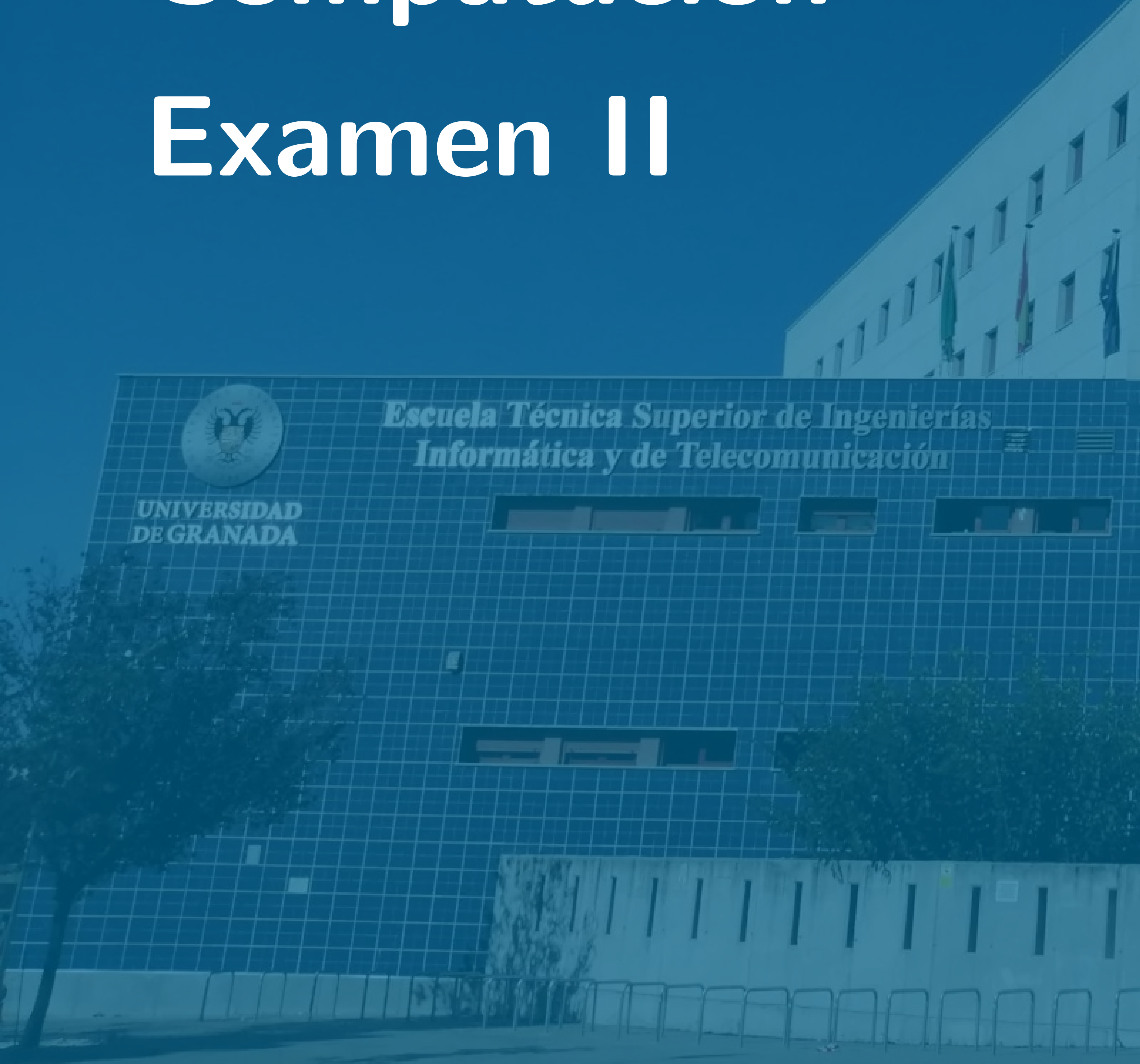
# Modelos de Computación Examen II





**Los Del DGIIM**, `losdeldgiim.github.io`

Doble Grado en Ingeniería Informática y Matemáticas
Universidad de Granada

# Modelos de Computación Examen II

Los Del DGIIM, `losdeldgiim.github.io`

Arturo Olivares Martos

Granada, 2024-2025

**Asignatura** Modelos de Computación

**Curso Académico** 2021-22.

**Grado** Doble Grado en Ingeniería Informática y Matemáticas.

**Descripción** Parcial Tema 2.

**Ejercicio 1.** Dar un AFD que acepte el siguiente lenguaje:

$$L = \{0^n 1^m \mid n - m \text{ es múltiplo de } 3\}.$$

Usaremos la siguiente notación para los estados:

- $q_i$: $n \bmod 3 = i$, $m = 0$.
- $q_{ij}$: $n \bmod 3 = i$, $m \bmod 3 = j$.

Necesitamos que $n - m$ sea múltiplo de 3. Para ello, neesitamos que el resto de dividir cada uno de ellos entre tres coincida, ya que se va a compensar haciendo que la resta sea múltiplo de tres. Tenemos por tanto el AFD de la Figura 1.

Notemos que todas las conexiones de error, para evitar que una vez se han introducido 1's se pueda volver a introducir un 0, se han hecho dibujado con menor opacidad para no saturar el diseño. Tal solo se ha establecido $\delta(q_{ij}, 0) = E$ para todo $i, j$.

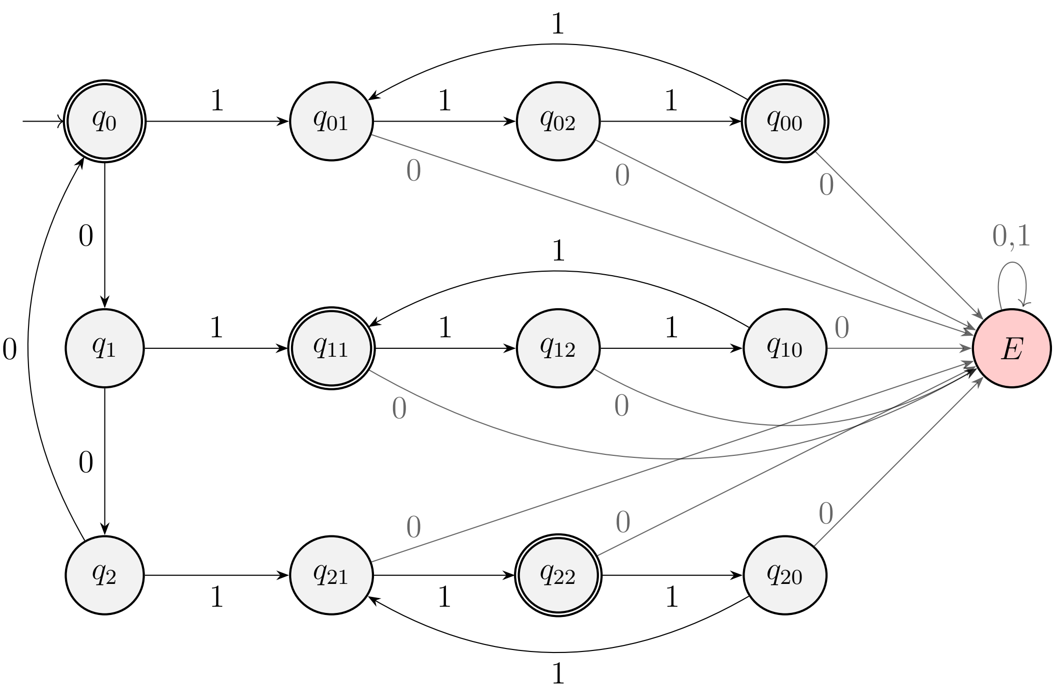



Figura 1: AFD que acepta el lenguaje $L$ del ejercicio 1.

Este diagrama tiene gran cantidad de estados. Una alternativa es hacer uso de que, una vez introducido algún 1, no se van a introducir más 0's. Por tanto, los estados son:

- $q_i$: $n \bmod 3 = i$, $m = 0$.
- $q_i'$: $n - m \bmod 3 = i$, $m \neq 0$.

El AFD resultante es el de la Figura 2.

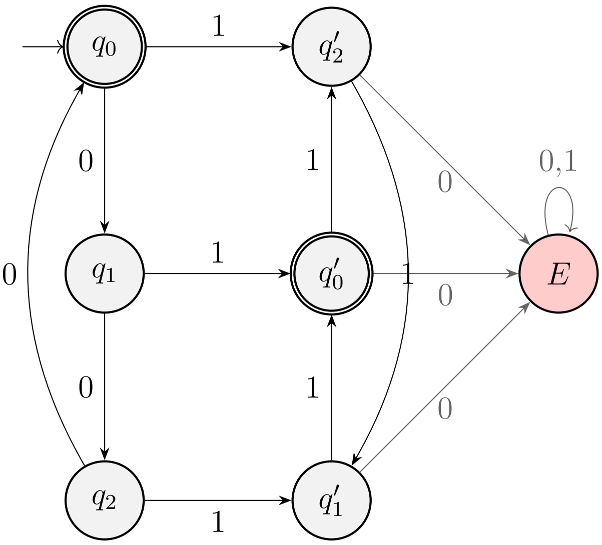



Figura 2: AFD alternativo que acepta el lenguaje $L$ del ejercicio 1.

**Ejercicio 2.** Considerar el lenguaje siguiente:

$$L = \{w \in \{0,1\}^* \mid \text{El tecer símbolo empezando por la derecha es 1}\}.$$

1. Dar un AFD que acepte $L$.

   Podríamos optar con construir el AFD de forma directa, pero construiremos un AFND que acepte el lenguaje y luego lo convertiremos a AFD. El AFND se muestra en la Figura 3, cuyos estados son:

   - $q_0$: No estamos en la cadena final, por lo que podemos leer 0's y 1's.
   - $q_1$: Acabo de empezar la cadena final. He leído un 1.
   - $q_2$: Estoy en la cadena final. El leído el 1 y el segundo símbolo.
   - $q_3$: Hemos terminado la cadena final.

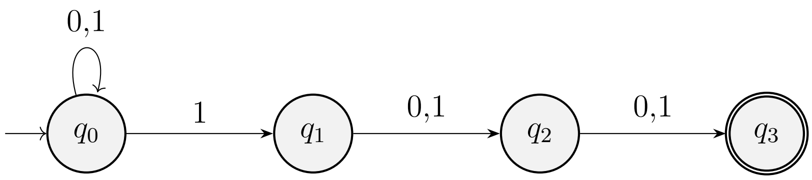



Figura 3: AFND que acepta el lenguaje $L$ del ejercicio 2.

Convertimos ahora el AFND de la Figura 3 en un AFD, representado en la Figura 4.

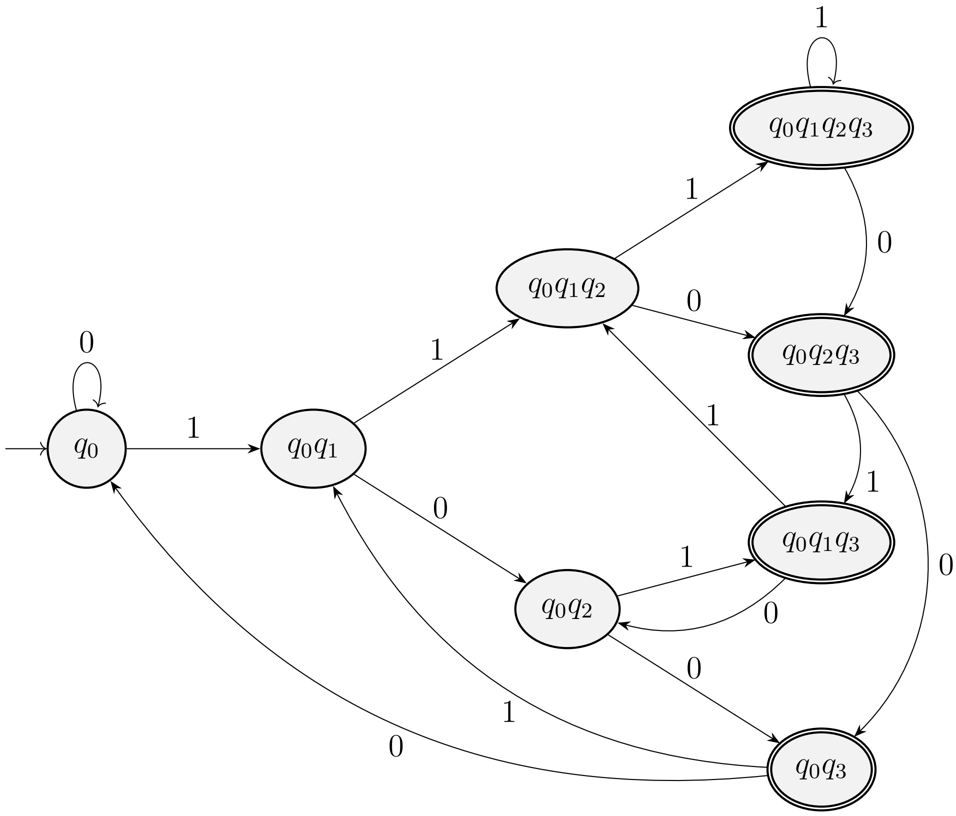



Figura 4: AFD que acepta el lenguaje $L$ del ejercicio 2.

El AFD de la Figura 4 acepta el lenguaje $L$, y es idéntico al que podríamos haber razonado de forma directa. Veamos qué representa cada estado:

- $q_0$: No estamos en un candidado a ser cadena final. Si leemos un 1, empezaremos la que puede ser la cadena final.
- $q_0q_1$: Hemos leído un 1, por lo que hemos empezado la posible cadena final. Llevamos 1.
- $q_0q_1q_2$: Hemos leído un 1 y un 1. Llevamos 11 de cadena final.
- $q_0q_2$: Hemos leído un 1 y un 0. Llevamos 10 de cadena final.
- $q_0q_1q_2q_3$: Hemos leído un 1, un 1 y un 1. Llevamos 111 de cadena final.
- $q_0q_2q_3$: Hemos leído un 1, un 1 y un 0. Llevamos 110 de cadena final.
- $q_0q_1q_3$: Hemos leído un 1, un 0 y un 1. Llevamos 101 de cadena final.
- $q_0q_3$: Hemos leído un 1, un 0 y un 0. Llevamos 100 de cadena final.

Lo complejo de hacerlo de forma directa sería ver las transiciones desde los estados finales. Razonando cuál es la cadena final leída, odríamos haberlo hecho de forma directa, pero el AFND nos ha ayudado a hacerlo de forma algorítmica.

2. Dar una expresión regular que genere $L$.

   Tenemos dos opciones:

**De forma directa:** La expresión regular que genera $L$ es:

$$(0+1)^*1(0+1)(0+1)$$

**Pasando de Autómata a Expresión Regular:** Debido al gran número de estados del AFD, vamos a pasar el AFND sin transiciones nulas a una expresión regular. Para ello, establecemos una ecuación por cada estado y resolvemos el sistema:

$$\begin{cases} q_0 = & 0q_0 + 1q_0 + 1q_1 = (0+1)q_0 + 1q_1 \\ q_1 = & 0q_2 + 1q_2 = (0+1)q_2 \\ q_2 = & 0q_3 + 1q_3 = (0+1)q_3 \\ q_3 = & \varepsilon \end{cases}$$

Sustituyendo $q_3$ en la ecuación de $q_2$, llegamos a que:

$$q_2 = (0+1)\varepsilon = (0+1)$$

Sustituyendo $q_2$ en la ecuación de $q_1$, llegamos a que:

$$q_1 = (0+1)(0+1)$$

Sustituyendo $q_1$ en la ecuación de $q_0$, llegamos a que:

$$q_0 = (0+1)q_0 + 1(0+1)(0+1) = (0+1)^*1(0+1)(0+1)$$

Por tanto, la expresión regular que genera $L$ es:

$$(0+1)^*1(0+1)(0+1)$$

Como podemos ver, hemos llegado a la misma expresión regular que de forma directa.

3. Dar una gramática regular por la izquierda que genere $L$.

En primer lugar, invertimos el AFND de la Figura 3 para obtener el AFND de la Figura 5.

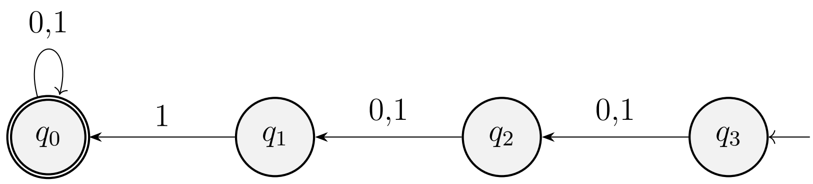



Figura 5: AFND que acepta el lenguaje $L^{-1}$, siendo $L$ el lenguaje del ejercicio 2.

Obtenemos ahora la gramática lineal por la derecha $G' = \{\{q_0, q_1, q_2, q_3\}, \{0,1\}, P', q_3\}$ que genera el lenguaje $L^{-1}$, siendo $P$ el conjunto de reglas:

$$P' = \begin{cases} q_3 \to & 0q_2 \mid 1q_2 \\ q_2 \to & 0q_1 \mid 1q_1 \\ q_1 \to & 1q_0 \\ q_0 \to & 0q_0 \mid 1q_0 \mid \varepsilon \end{cases}$$

Ahora, a partir de $G'$, obtenemos la gramática lineal por la izquierda, $G$, que genera el lenguaje $L$, donde $G = \{\{q_0, q_1, q_2, q_3\}, \{0, 1\}, P, q_3\}$ siendo $P$ el conjunto de reglas:

$$P = \begin{cases} q_3 \to & q_2 0 \mid q_2 1 \\ q_2 \to & q_1 0 \mid q_1 1 \\ q_1 \to & q_0 1 \\ q_0 \to & q_0 0 \mid q_0 1 \mid \varepsilon \end{cases}$$