Actividad Evaluable 1

Abraham Álvarez Cruz

5 de abril de 2023

1. Parte 1

1.1. Paso 1

1.1.1. Creación de expresion regular

El primer paso que tuvimos que llevar a cabo fue la elaboración de la expresión regular que más tarde convertiremos al autómata finito determinista que implementaremos en un lenguaje de programación (en nuestro caso, Python).

Tras un rato, llegué a la siguiente expresión regular:

$$(A|C|G|T)*(TGGGCGTTT)+(A|C|G|T)*$$

La primera parte de la expresión nos permitirá reconocer cualquier cadena que contenga (o no) varios caracteres de los que están incluídos en el paréntesis. La segunda parte reconoce aquellas cadenas que tienen la secuencia $TGGGCGTTT$. Por último, la tercera y última parte reconoce lo mismo que la primera pero después de la secuencia de la segund parte.

1.2. Paso 2

1.2.1. Conversión de expresión regular a $\mathcal{E}$-NFA

Para la conversión de expresión regular a $\mathcal{E}$-NFA aplicamos el método de Thompson. Así, tras aplicar las distintas traducciones para cada bloque de la expresión regular, obtuvimos una tabla de transiciones de nuestro $\mathcal{E}$-NFA con 52 estados.

1.3. Paso 3

1.3.1. Conversión de $\mathcal{E}$-NFA a NFA

1.3.2. Cálculo de cierres

Lo primero que tenemos que hacer es calcular los cierres de cada estado, esto es en resumidas formas, *hasta dónde puede llegar cada estado mediante $\mathcal{E}$-transiciones única y exclusivamente*. Es importante remarcar que en esa lista de estados que conforman el cierre de cada estado, aparezca él mismo.

	A	C	G	T	eps	CL
q0	.	.	.	.	q1,q6	q0,q1,q2,q3,q6,q7,q8
q1	.	.	.	.	q2,q3	q1,q2,q3
q2	q4	.	.	.	.	q2
q3	.	q5	.	.	.	q3
q4	.	.	.	.	q11	q4,q11,q13,q0,q1,q2,q3,q6,q7,q8,q14,q16,q17
q5	.	.	.	.	q11	q5,q11,q13,q0,q1,q2,q3,q6,q7,q8,q14,q16,q17
q6	.	.	.	.	q7,q8	q6,q7,q8
q7	.	.	q9	.	.	q7
q8	.	.	.	q10	.	q8
q9	.	.	.	.	q12	q9,q12,q13,q0,q1,q2,q3,q6,q7,q8,q14,q16,q17
q10	.	.	.	.	q12	q10,q12,q13,q0,q1,q2,q3,q6,q7,q8,q14,q16,q17
q11	.	.	.	.	q13	q11,q13,q0,q1,q2,q3,q6,q7,q8,q14,q16,q17
q12	.	.	.	.	q13	q12,q13,q0,q1,q2,q3,q6,q7,q8,q14,q16,q17
q13	.	.	.	.	q14,q0	q13,q0,q1,q2,q3,q6,q7,q8,q14,q16,q17

Figura 1: Ejemplo de cierre de los primeros 14 estados.

1.3.3. Creación del NFA con los cierres y el $\mathcal{E}$-NFA

Ahora que tenemos el cierre de cada estado, podemos crear nuestro NFA. Para hacerlo, tenemos que seguir el siguiente algoritmo:

1. Recorremos cada estado de la tabla de transiciones.
2. Para cada estado, cogemos todos los estados que conforman su cierre.
3. Unimos las transiciones de todos los estados que conforman el cierre **para cada símbolo por separado**, esto es, la unión de las transiciones para cada símbolo (columnas).
4. Esa unión de estados conformarán las transiciones del estado que estamos iterando para un símbolo en particular (la columna que hayamos mirado).

	A	C	G	T
q0	q4	q5	q9	q10
q1	q4	q5	.	.
q2	q4	.	.	.
q3	.	q5	.	.
q4	q4	q5	q9	q10,q18
q5	q4	q5	q9	q10,q18
q6	.	.	q9	q10
q7	.	.	q9	.
q8	.	.	.	q10
q9	q4	q5	q9	q10,q18
q10	q4	q5	q9	q10,q18

Figura 2: Ejemplo de creación de NFA de los primeros 11 estados.

1.4. Paso 4

1.4.1. Creación del DFA a partir del NFA

Para la creación del DFA a partir del NFA tenemos que eliminar las transiciones múltiples a partir de un estado y un símbolo. Esto lo lograremos siguiendo el siguiente algoritmo:

1. Crearemos una nueva tabla de transiciones en el que iremos añadiendo las nuevas transiciones.
2. Comenzamos mirando el estado inicial.
3. Vamos iterando cada uno de los símbolos del alfabeto
   a) **Si hay transición a múltiples estados**, los renombramos para que sean uno sólo. Ejemplo: q1,q3,q5 será q1_3_5 (no tiene por qué ser renombrados así pero es importante mantener un mismo esquema).
   b) **Si NO hay transición a múltiples estados**, no realizamos ningún renombrado.
   c) Añadimos el estado a la tabla de transiciones nueva para ser procesado posteriormente.
   d) Por último, al estado que estamos iterando, para el símbolo que estamos iterando, añadimos en la nueva tabla de transiciones una transición al estado que acabamos (o no) de renombrar. En el caso de haber sido renombrado, hemos pasado de tener una transición múltiple a una 'transición simple'.
4. Miramos el siguiente estado de nuestra nueva tabla de transiciones para el que no tengamos transiciones y volvemos al paso 3.

5. Una vez procesados todos los estados de nuestra tabla de transiciones 'antigua', marcaremos como inicial el estado que también era inicial en la tabla de transiciones 'antigua' y marcaremos como estados finales aquellos en los que, al ser renombrados, al menos uno de los estados era final. Ejemplo: q1 no es final, q2 no es final, q3 sí es final luego, q1_2_3 será final.

6. Para poder trabajar más cómodos, es conveniente renombrar los estados para reducir la complejidad de los nombres. Este paso es sencillo, allá donde hayamos puesto un determinado estado, lo modificamos por el nuevo nombre. Ejemplo: q1_2_3 pasará a llamarse q1 luego, en cualquier lugar de la nueva tabla donde esté q1_2_3 lo cambiaremos por q1.

q3	.	q5	.	.
q4	q4	q5	q9	q10,q18
q5	q4	q5	q9	q10,q18
q6	.	.	q9	q10
q7	.	.	q9	.
q8	.	.	.	q10
q9	q4	q5	q9	q10,q18
q10	q4	q5	q9	q10,q18
q11	q4	q5	q9	q10,q18
q12	q4	q5	q9	q10,q18
q13	q4	q5	q9	q10,q18
q14	.	.	q18	.
q15	q4	q5	q9	q10,q18

Figura 3: Ejemplo de tabla de transición NFA que convertiremos a DFA.

	A	**C**	**G**	**T**
q15	q4	q5	q9	q10_18
q4	q4	q5	q9	q10_18
q5	q4	q5	q9	q10_18
q9	q4	q5	q9	q10_18
q10_18	q4	q5	q9_20	q10_18

Figura 4: Ejemplo de tabla de transición DFA generada a partir de NFA.

1.5. Paso 5

1.5.1. Minimización del DFA

Este paso es un poco más difícil de explicar con palabras pero es muy sistemático.

1. Construiremos una tabla con todos los pares de estados, esto es, una tabla con todos los estados tanto en columna como en fila.

2. Marcaremos todos los pares de estados (fila x, columna y) que sean distinguibles. Esto es:

   a) La posición x,y pertenece a un estado final y otro no.

   b) Hay un símbolo cuya transición para el estado x y el estado y nos da un nuevo par de estados z,k que sí están distinguidos en nuestra tabla.

   c) Repetir el paso anterior (b) hasta que no haya nada más que marcar.

3. Todos los pares de estados nos marcados son equivalentes. En caso de haber una transitividad, se resumirán en un único estado. Ejemplo: el estado x es equivalente al estado y. Además, el estado y es equivalente al estado z luego, x,y,z son equivalentes.

Para ponernos en contexto, tras aplicar la minimización al DFA obtenido en el paso 4, he obtenido un nuevo DFA con 15 estados menos.

	A	C	G	T
q0	q1	q2	q3	q4
q1	q1	q2	q3	q4
q2	q1	q2	q3	q4
q3	q1	q2	q3	q4
q4	q1	q2	q5	q4
q5	q1	q2	q6	q4
q6	q1	q2	q7	q4
q7	q1	q8	q3	q4
q8	q1	q2	q9	q4
q9	q1	q2	q3	q10
q10	q1	q2	q5	q11
q11	q1	q2	q5	q12
q12	q13	q14	q15	q16
q13	q13	q14	q17	q16
q14	q13	q14	q17	q16
q15	q13	q14	q18	q16
q16	q13	q14	q15	q16
q17	q13	q14	q17	q16
q18	q13	q14	q19	q16
q19	q13	q20	q17	q16
q20	q13	q14	q21	q16
q21	q13	q14	q17	q22
q22	q13	q14	q15	q23
q23	q13	q14	q15	q24
q24	q13	q14	q15	q16

Figura 5: Tabla de transiciones del DFA sin minimizar.

	A	C	G	T
q0	q0	q0	q0	q1
q1	q0	q0	q2	q1
q2	q0	q0	q3	q1
q3	q0	q0	q4	q1
q4	q0	q5	q0	q1
q5	q0	q0	q6	q1
q6	q0	q0	q0	q7
q7	q0	q0	q2	q8
q8	q0	q0	q2	q9
q9	q9	q9	q9	q9

Figura 6: Tabla de transiciones del DFA minimizado.

2. Parte 2

2.1. Paso 1

2.1.1. Creación del DFA

La creación del segundo DFA fué más directa que en la primera parte ya que es un autómata determinista finito sencillo de generar de manera directa. El resultado se puede observar en la figura 7 que se muestra a continuación.

	A	C	G	T
q0	q1	q4	q4	q3
q1	q4	q4	q4	q2
q2	q2	q2	q2	q2
q3	q2	q4	q4	q4
q4	q4	q4	q4	q4

Figura 7: Tabla de transiciones del segundo DFA.

2.2. Paso 2

2.2.1. Minimización del DFA

Al igual que en la primera parte, tenemos que obtener un DFA mínimo que nos permita trabajar con mayor comodidad en los siguientes pasos. En mi caso, creé un DFA que ya era mínimo así que no hay ningún cambio con respecto al paso anterior.

3. Parte 3

3.1. Paso 1

3.1.1. Producto de los DFA's

Para este último paso, partimos de los DFA's que hemos generado tanto en la parte 1 como en la 2. Ambos son mínimos así que podremos trabajar más cómodamente. El DFA que generaremos en este paso combinará la funcionalidad de los dos DFA, es decir, el resultado de este paso será la conjunción de ambos DFA's.

Para realizar el producto de ambos DFA's seguiremos el siguiente algoritmo:

1. Realizaremos el producto cartesiano de los estados de ambos DFA's, es decir, combinaremos todos los estados de un DFA con todos los estados del otro. Ejemplo: el primer DFA tiene los estados q1 y q2. El segundo DFA tiene los estados q3 y q4. El producto cartesiano será: q1_3, q1_q4, q2_q3, q2_q4.

2. Para las nuevas transiciones:

   a) Iteramos cada estado recién generado.

   b) Iteramos cada símbolo del alfabeto.

   c) La transición del estado que estamos iterando para el símbolo que estamos iterando será el resultado de combinar las transiciones para el símbolo actual de los estados que conforman el estado actual.

3. El estado inicial será aquel que combine los estados iniciales de ambos DFA's y los estados finales serán aquellos que combinen los estados finales de ambos DFA's.

4. Por último, renombramos los estados para obtener una solución más simples sin nombres complejos (similar a como hicimos en la primera parte).

3.2. Paso 2

3.2.1. Minimización del producto

Por último, para obtener una solución mínima, aplicamos nuevamente el algoritmo de minimización.

	A	C	G	T
q0	q5	q5	q3	q12
q1	q5	q5	q3	q1
q2	q5	q5	q11	q1
q3	q5	q5	q2	q1
q4	q5	q5	q5	q0
q5	q5	q5	q5	q1
q6	q8	q8	q8	q1
q7	q7	q7	q7	q7
q8	q8	q8	q8	q8
q9	q5	q8	q8	q8
q10	q6	q8	q8	q9
q11	q5	q13	q5	q1
q12	q5	q5	q3	q7
q13	q5	q5	q4	q1

Figura 8: Tabla de transiciones del DFA final minimizado.