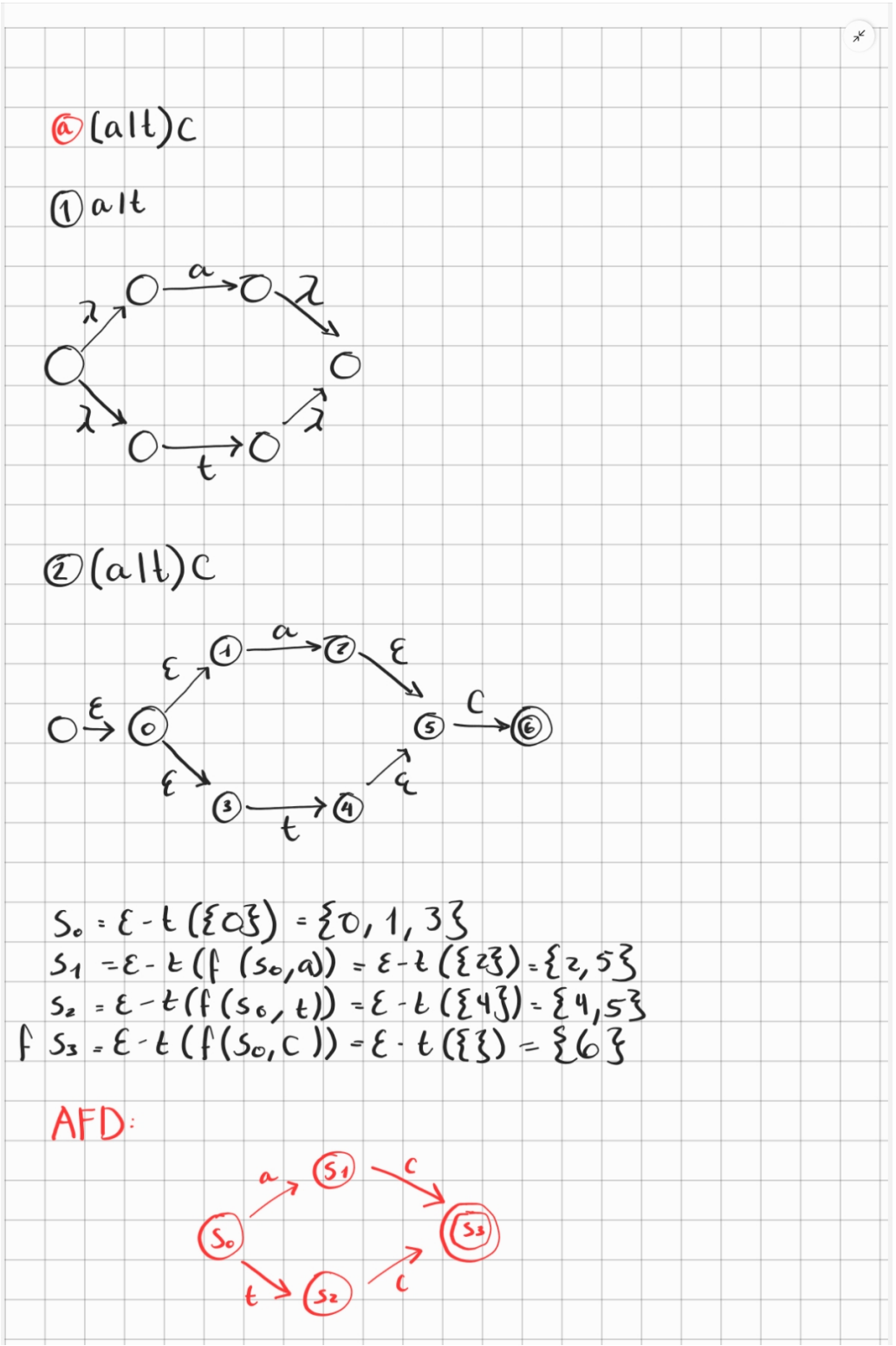
UNIVERSIDAD DEL VALLE DE GUATEMALA
Teoría de la Computación
Sección 20
Ing. Gabriel Brolo Tobal

Tarea de resumen 1
Costos de Proyecto de SW

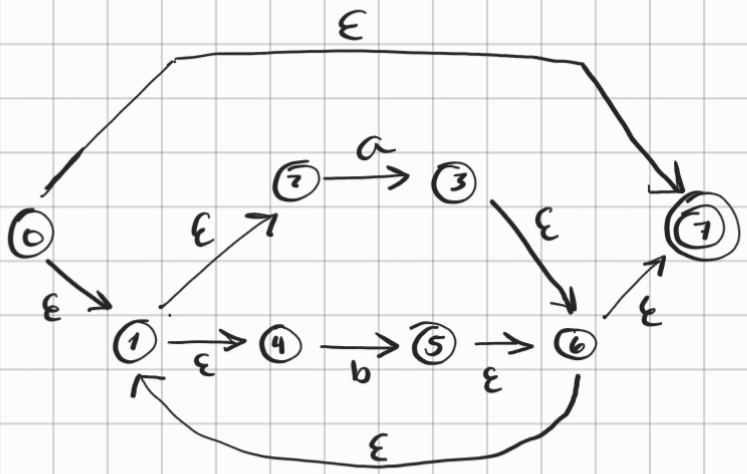
Maria Ramirez 21342
Gustavo Gonzalez 21438

Guatemala 26 de julio del 2,023

Ejercicio No.1



(b) (a|b)*

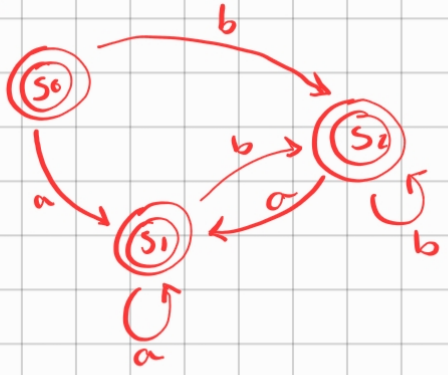


f $S_0 = \varepsilon\text{-}t(\{0\}) = \{0, 1, 2, 4, 7\}$

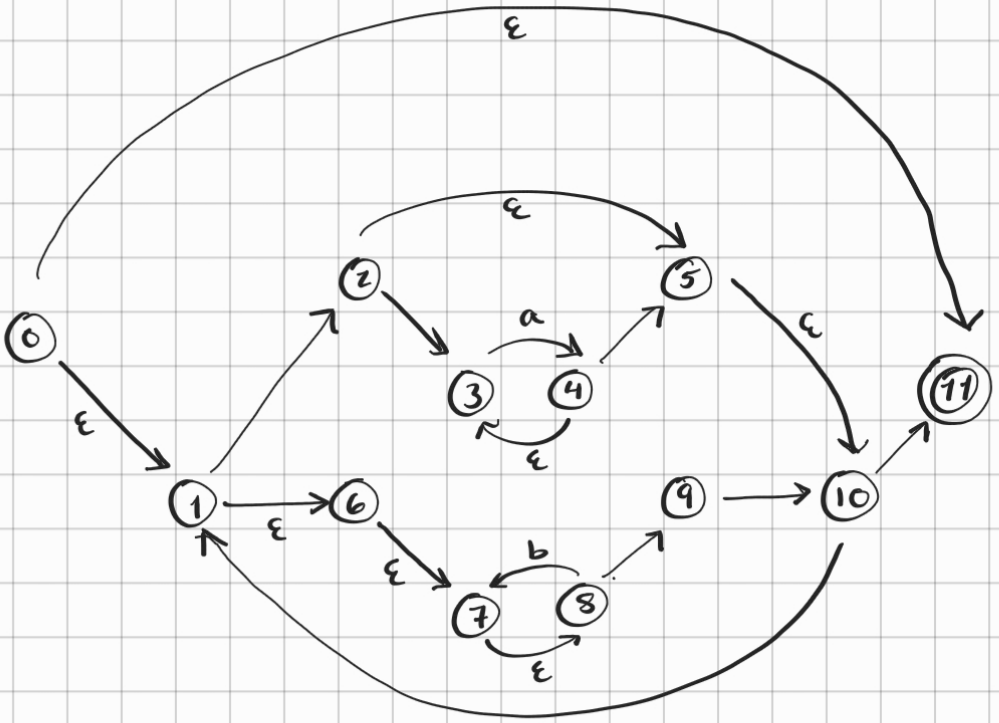
f $S_1 = \varepsilon\text{-}t(f(S_0, a)) = \varepsilon\text{-}t(\{3\}) = \{1, 2, 3, 4, 6, 7\}$

f $S_2 = \varepsilon\text{-}t(f(S_0, b)) = \varepsilon\text{-}t(\{5\}) = \{1, 2, 4, 5, 6, 7\}$

AFD:



© $(a^*|b^*)^*$

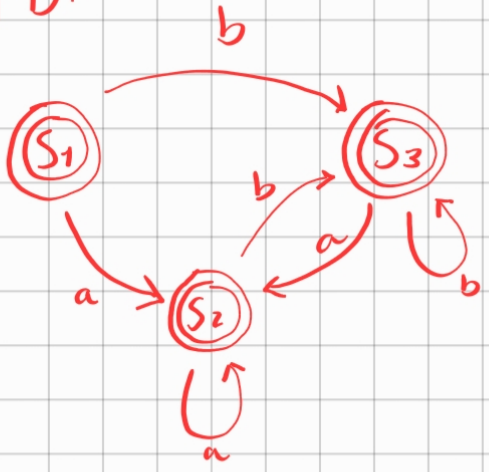


f $S_0 = \varepsilon\text{-}t(\{0\}) = \{0,1,2,3,5,6,7,9,10,11\}$

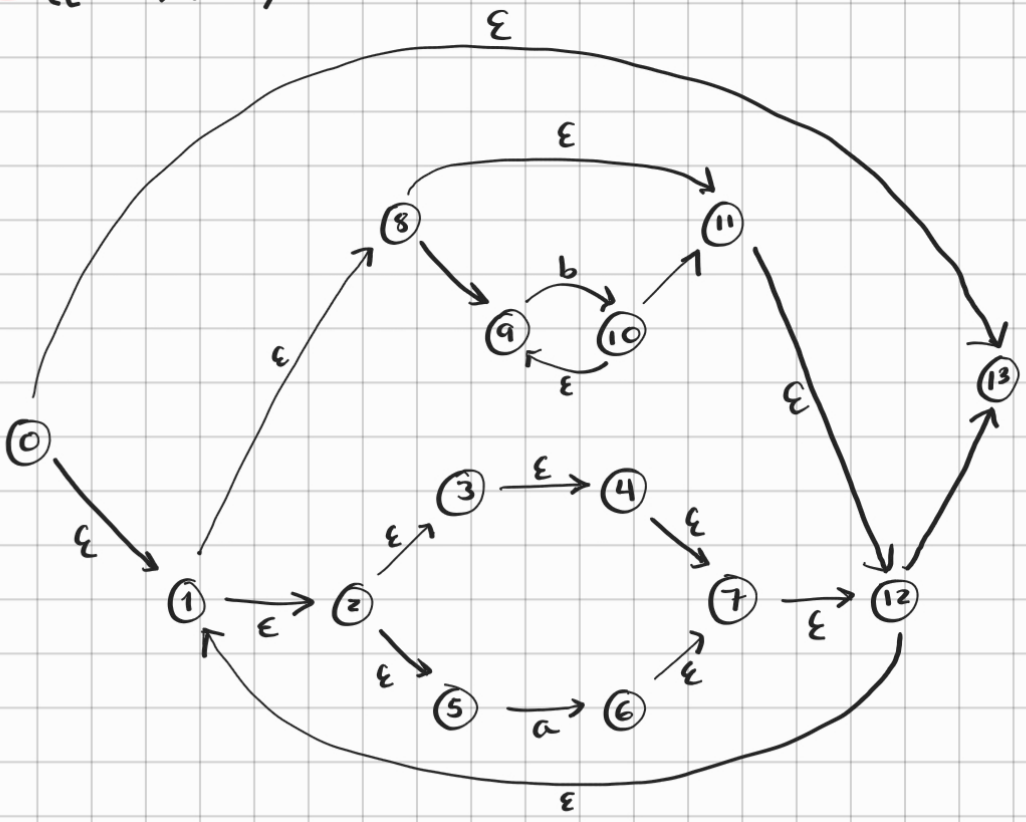
f $S_1 = \varepsilon\text{-}t(f(S_0,a)) = \varepsilon\text{-}t\{(4)\} = \{1,2,3,4,5,6,7,9,10,11\}$

f $S_2 = \varepsilon\text{-}t(f(S_0,b)) = \varepsilon\text{-}t\{(8)\} = \{1,2,3,5,6,7,8,9,10,11\}$

AFD:



ⓓ $((\varepsilon|a)|b^*)^*$

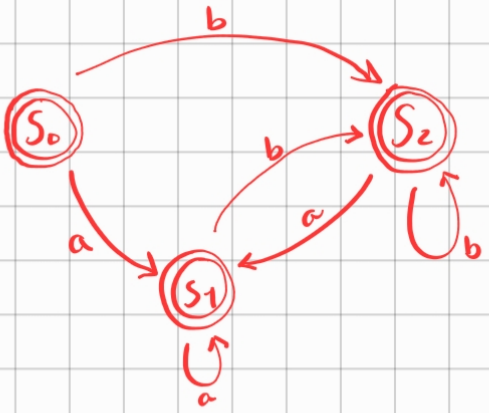


f $S_0 = \varepsilon\text{-}t\{(0)\} = \{0,1,2,3,4,5,7,8,9,11,12,13\}$

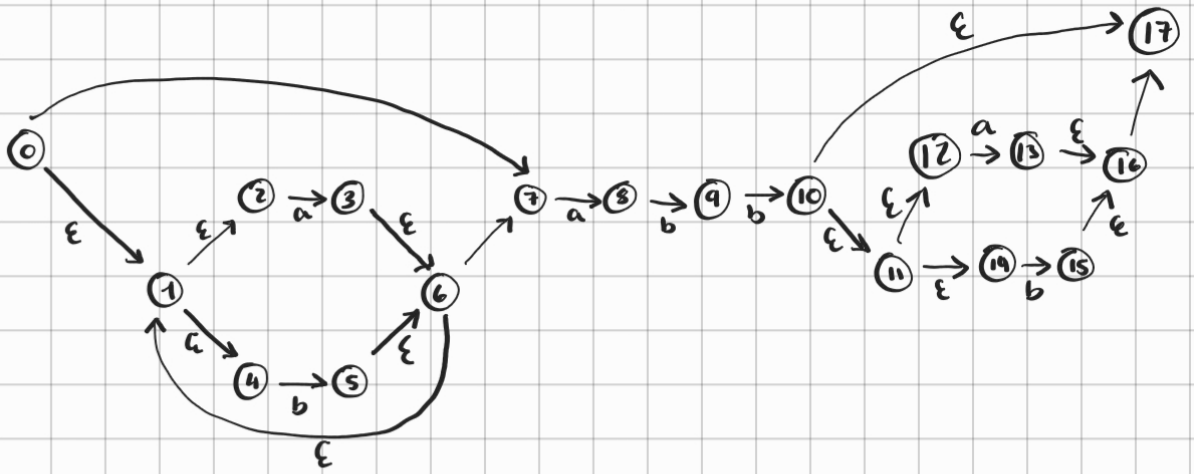
f $S_1 = \varepsilon\text{-}t(f(S_0,a)) = \varepsilon\text{-}t\{(6)\} = \{1,2,3,4,5,6,7,8,9,11,12,13\}$

f $S_2 = \varepsilon\text{-}t(f(S_0,b)) = \varepsilon\text{-}t\{(10)\} = \{1,2,3,4,5,7,8,9,10,11,12,13\}$

AFD:



(a|b)* abb(a|b)*



$S_0 = \varepsilon\text{-t}\{(0)\} = \{0,1,2,4,7\}$ A

$S_1 = \varepsilon\text{-t}(f(S_0,a)) = \varepsilon\text{-t}\{(3)\} = \{1,2,3,4,6,7,8\}$ B

$S_2 = \varepsilon\text{-t}(f(S_0,b)) = \varepsilon\text{-t}\{(5)\} = \{1,2,4,5,6,7\}$ C

$S_3 = \varepsilon\text{-t}(f(S_1,a)) = \varepsilon\text{-t}\{(9)\} = \{1,2,4,5,6,7,9\}$ D

$S_4 = \varepsilon\text{-t}(f(S_1,b)) = \varepsilon\text{-t}\{(5)\} = \{1,2,4,5,6,7,10,11,12,14,17\}$ E

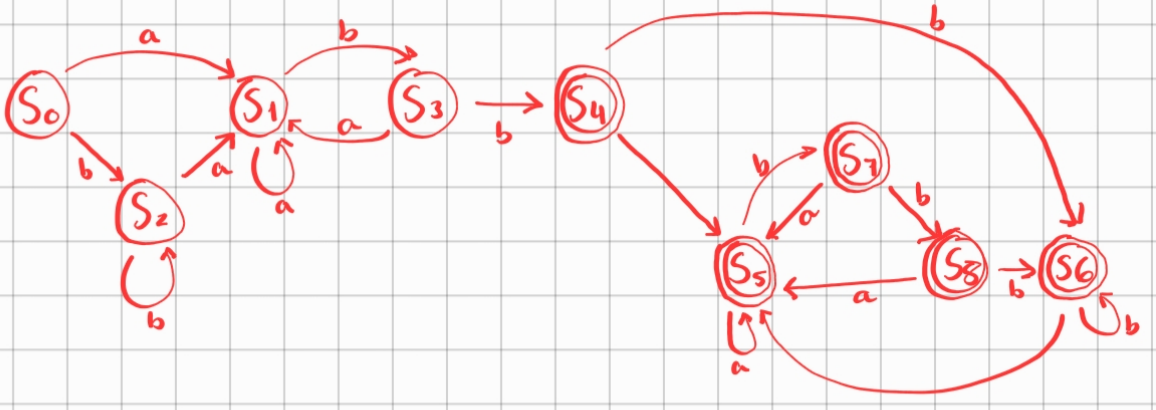
$S_5 = \varepsilon\text{-t}(f(S_2,a)) = \varepsilon\text{-t}\{(3)\} = \{1,2,3,4,6,7,8,11,12,13,14,16,17\}$ F

$S_6 = \varepsilon\text{-t}(f(S_2,b)) = \varepsilon\text{-t}\{(5)\} = \{1,2,4,5,6,7,11,12,14,15,16,17\}$ G

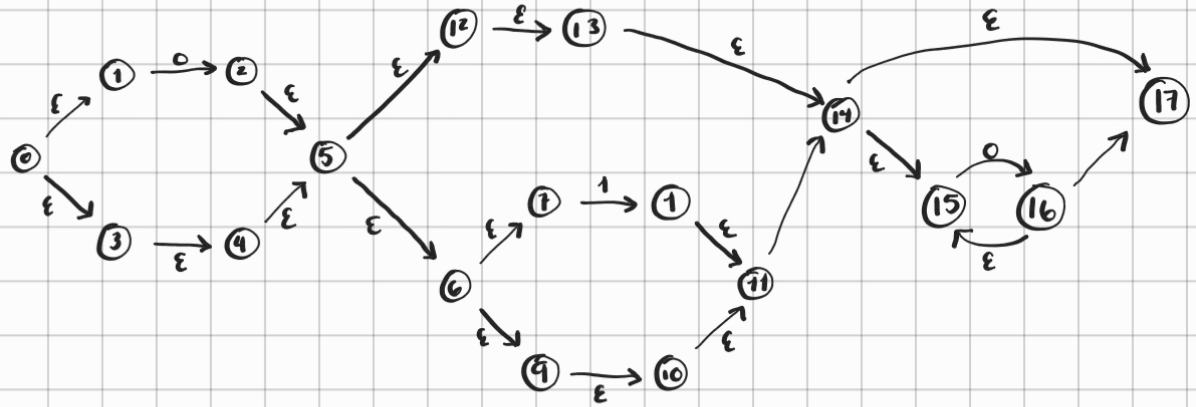
$S_7 = \varepsilon\text{-t}(f(S_3,a)) = \varepsilon\text{-t}\{\} = \{1,2,4,5,6,7,9,11,12,14,15,16,17\}$ H

$S_8 = \varepsilon\text{-t}(f(S_3,b)) = \varepsilon\text{-t}\{(5)\} = \{1,2,4,5,6,7,10,11,12,14,15,16,17\}$ I

AFD:



(f) $0?(1?)?0^*$



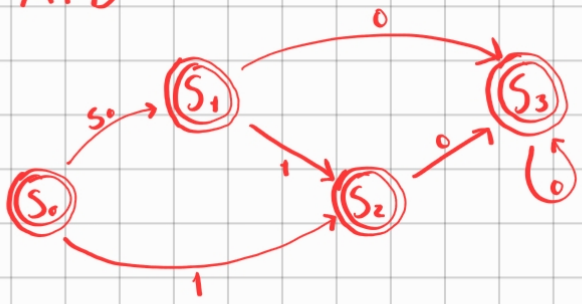
f $S_0 = \varepsilon\text{-}t\{(0)\} = \{0, 1, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 17\}$

f $S_1 = \varepsilon\text{-}t(f(S_0, 0)) = \varepsilon\text{-}t\{(2)\} = \{2, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17\}$

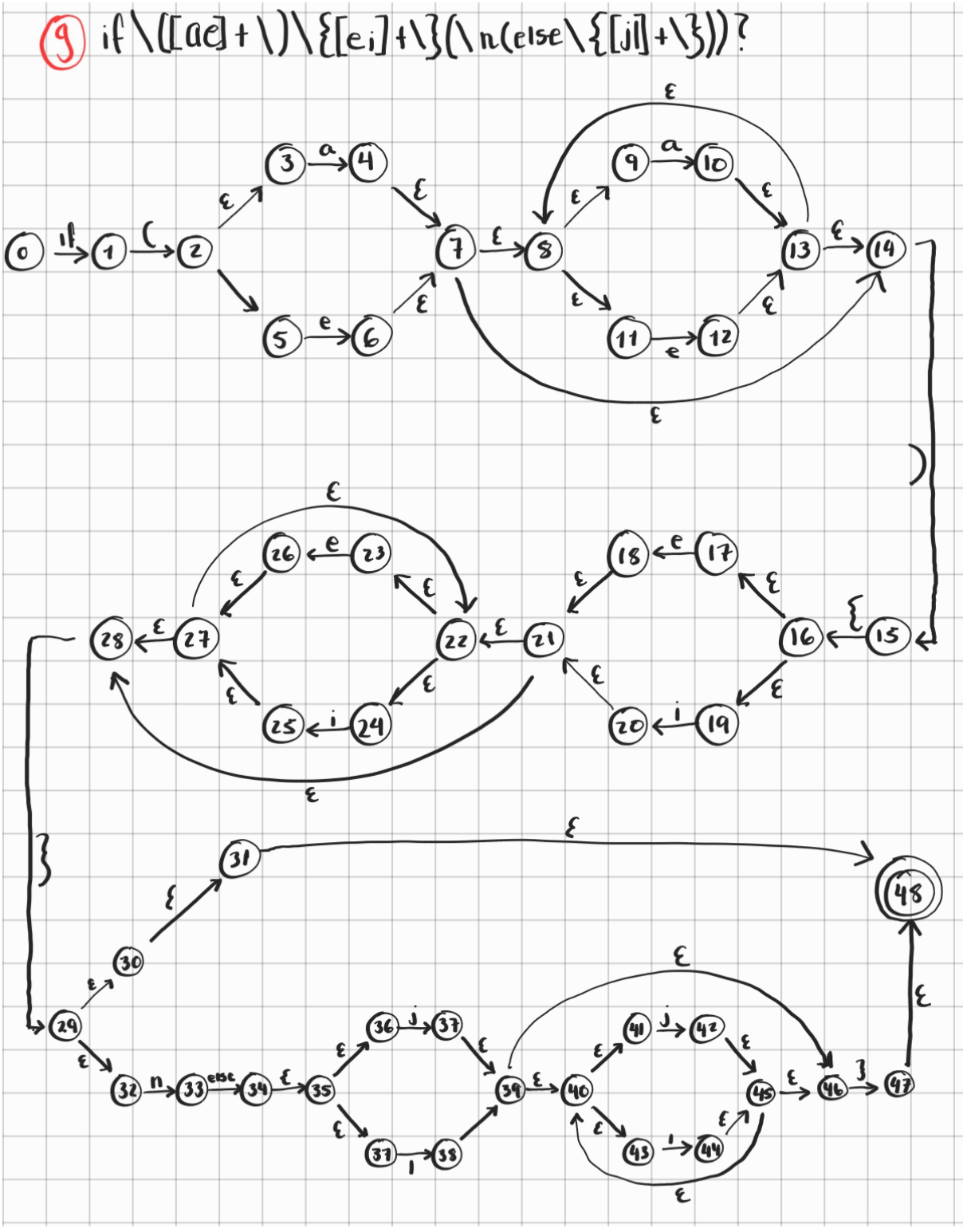
f $S_2 = \varepsilon\text{-}t(f(S_0, 1)) = \varepsilon\text{-}t\{(8)\} = \{8, 11, 14, 15, 17\}$

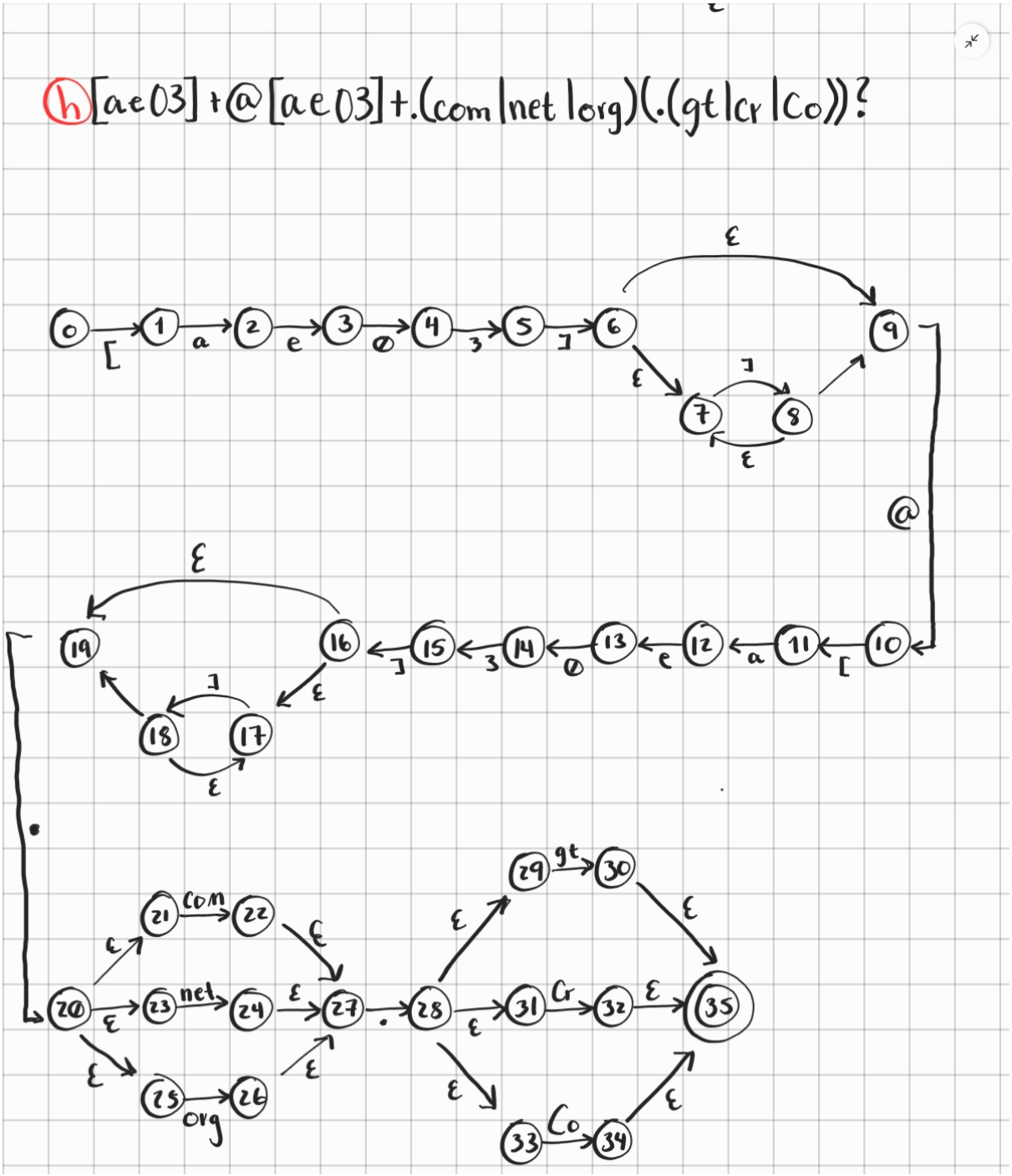
f $S_3 = \varepsilon\text{-}t(f(S_1, 0)) = \varepsilon\text{-}t\{(15)\} = \{15, 16, 17\}$

AFD:



(9) if\([ae]+\)\{[ei]+\}(\n(else\{[jl]+\}))?





Ejercicio No.2

(Repositorio GitHub)

Ejercicio No.3

Investigación:

Este es un algoritmo inventado por el científico Edsger W. Dijkstra en 1961. Y su uso original era poder evaluar expresiones matemáticas en compiladores. Seguidamente se generalizó para trabajar con expresiones regulares y lenguajes formales.

Adentrándose más en su uso, este es un método para convertir expresiones matemáticas o lógicas en notación infija a notación postfija. Esta última es útil para evaluar expresiones aritméticas o lógicas de manera eficiente, ya que elimina la necesidad de utilizar paréntesis y reglas de precedencia.
La idea básica detrás de este algoritmo de *Shunting Yard* es utilizar dos estructuras de datos; una cola (*queue*) y una pila (*stack*). De manera que a medida que se procesa la expresión infija, los operandos se colocan en la cola mientras que los operadores se van a la pila. El algoritmo mantiene una prioridad de operadores para asegurarse de que se respeten las reglas de precedencia al convertir la expresión.

Es proceso de conversión se realiza así:

1. Se recorre la expresión infija de izquierda a derecha.
2. Si se encuentra un operando (número, variable, etc.), se agrega directamente a la cola de salida.
3. Si se encuentra un operador, se comparará con el operador en la cima de la pila.
    a. Si la pila se encuentra vacía o contiene un paréntesis de apertura ‘(‘, el operador se va a la pila.
    b. Si el operador en la cima de la pila tiene mayor prioridad que el operador actual, se saca el operador de la pila y se agrega a la cola de salida. Esto se repite hasta que la pila esté vacía o el operador en la cima tenga menor prioridad que el operador actual.
    c. Finalmente, se agrega el operador actual de la pila.
4. Si se encuentra un paréntesis de apertura ‘(‘, se va a la pila.
5. Si se encuentra un paréntesis de cierre ‘)’, se sacan todos los operadores de la pila y se van a la cola de salida hasta encontrar el paréntesis de apertura correspondiente ‘(‘.

La notación postfija resultante en la cola de salida representa la misma expresión, pero sin necesidad de utilizar los paréntesis, ya que la notación postfija establece el orden de evaluación de manera explícita.

Referencia:

Central European Researchers Journal. (2016). Obtenido de Central European Researchers Journal: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjdmazwqrWAAxUzsTEKHUtGDKoQFnoECA8QAQ&url=https%3A%2F%2Fceres-journal.eu%2Fdownload.php%3Ffile%3D2016_02_full.pdf&usg=AOvVaw3jcIcH5GtiW53su1FBpZQs&opi=89978449