

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**Subject: Theory of Computation**

**Topic: Barcode Scanner for Library**

|  |  |
| --- | --- |
| **Name** | **Roll Number** |
| Meenakshi S | CB.EN.U4CSE21035 |
| Picheri Likitha | CB.EN.U4CSE21044 |
| Ragala Tejdeep | CB.EN.U4CSE21046 |
| Sowrabh kumar | CB.EN.U4CSE21058 |
| Siddarth varma U | CB.EN.U4CSE21064 |

Case Study - TOC

Problem statement:

The problem at hand is to develop an automated barcode scanning system for library books using a pushdown automaton that simulates a Turing machine. The purpose of this system is to enhance book tracking and inventory management, improve efficiency and accuracy. By incorporating a barcode scanner, the system captures the book's information. The pushdown automaton processes scanned data, updating the library's inventory and transaction records. A user-friendly interface enables librarians to interact with the system, displaying book details and transaction status. Seamless integration with existing library management system ensures data synchronization and minimizes errors, optimizes the overall book transaction process

Justification:

The pushdown automaton (PDA) emulating a Turing machine in a library barcode scanning system offers several benefits, including flexibility in data processing, computational power, modularity and scalability, standardization and compatibility, and theoretical foundation. A PDA can store and process data using a stack, enabling efficient storage and retrieval of scanned barcode data, facilitating book inventory updates and transaction recording. It also provides computational power, enabling advanced data processing, complex book tracking algorithms, and handling diverse transaction scenarios within the library environment. Modularity and scalability are also achieved through the design of the system, allowing for the addition of additional functionalities and complex book tracking rules. Theoretical foundations of computer science and automata theory align with the use of a PDA for accurate and reliable book tracking and inventory management. Overall, the PDA's flexibility, computational power, modularity, compatibility, and theoretical foundation make it an ideal choice for efficient book tracking and inventory management in libraries.

Pushdown Automata:

A pushdown automaton (PDA) is a computational model that incorporates a stack, a finite set of states, input alphabet, stack alphabet, transition rules, and acceptance conditions. It can read input symbols, modify the stack, and transition between states based on the current symbol and the stack. The stack allows for last-in-first-out (LIFO) information management, recognizing and processing context-free languages.

**Push Down Automata is designed as:**

M= (Q, Σ, Γ, δ, q0, z, F)

Q symbolizes for finite set of states

Σ symbolizes for finite set of input alphabets

Γ symbolizes for finite set of stack elements

δ symbolizes for state transitions

Q0 represents the initial state

Z represents the initial stack symbol

F represents the Final state

Multi Stack Push Down Automata:

A multi-stack pushdown automaton is a computational model with multiple stacks, enabling more complex computations and enhanced expressive power compared to single-stack automatons. This stack functionality stores and processes more information during computations.

X – it is used to get input from user and it stores it to stack

Y – this is used to get the user input which is the search element

K – to start checking

# - TO sperate the user input

{0,1 } – is the binary form for the book Unique id

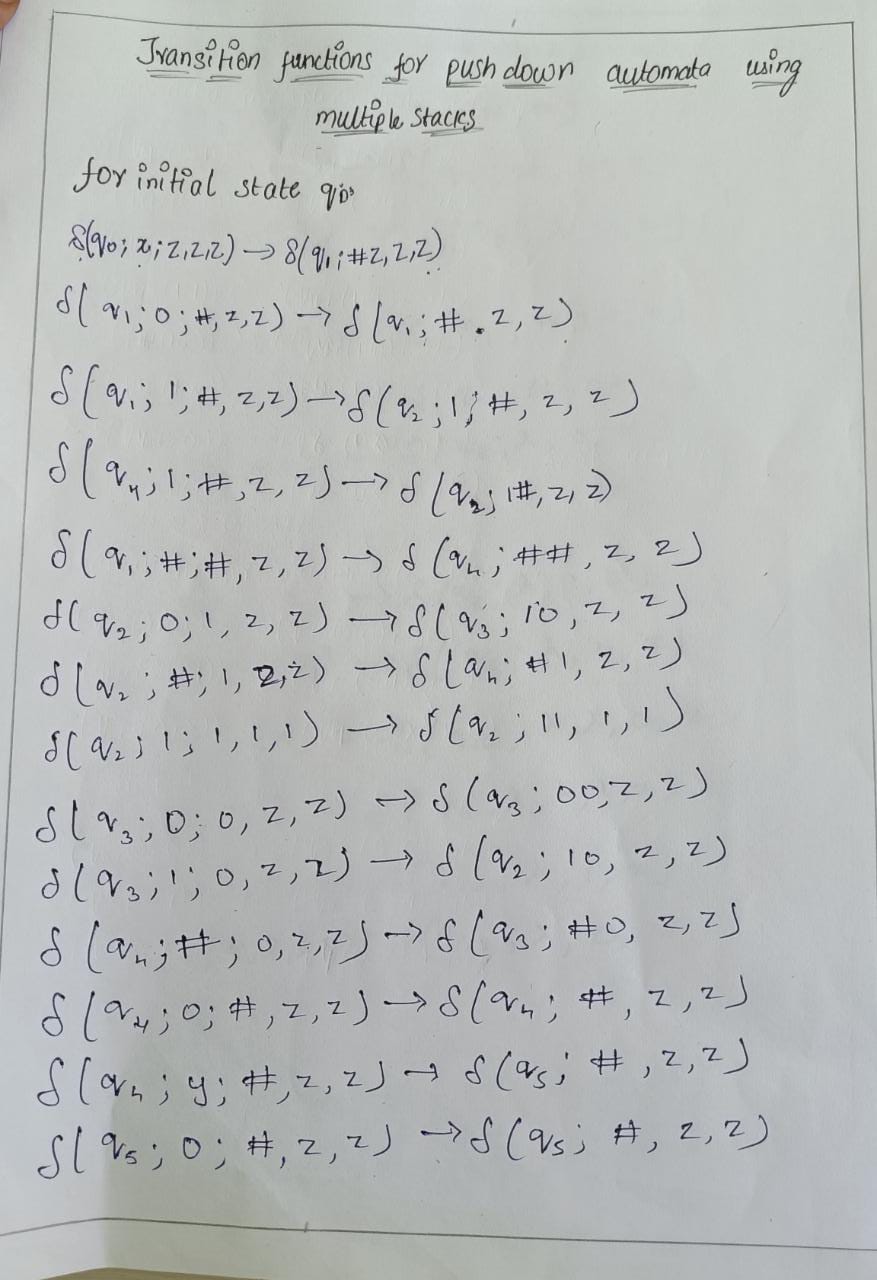
Alphabet set :

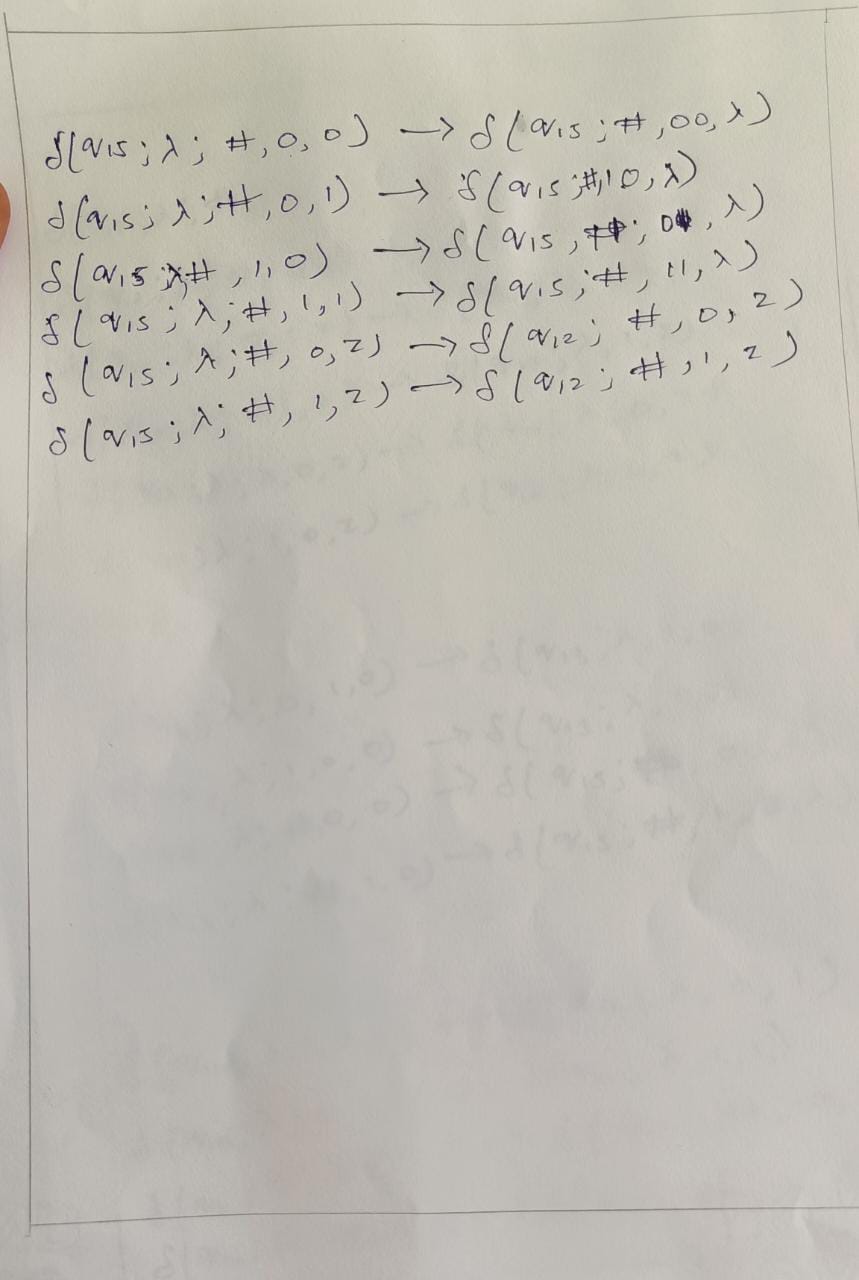
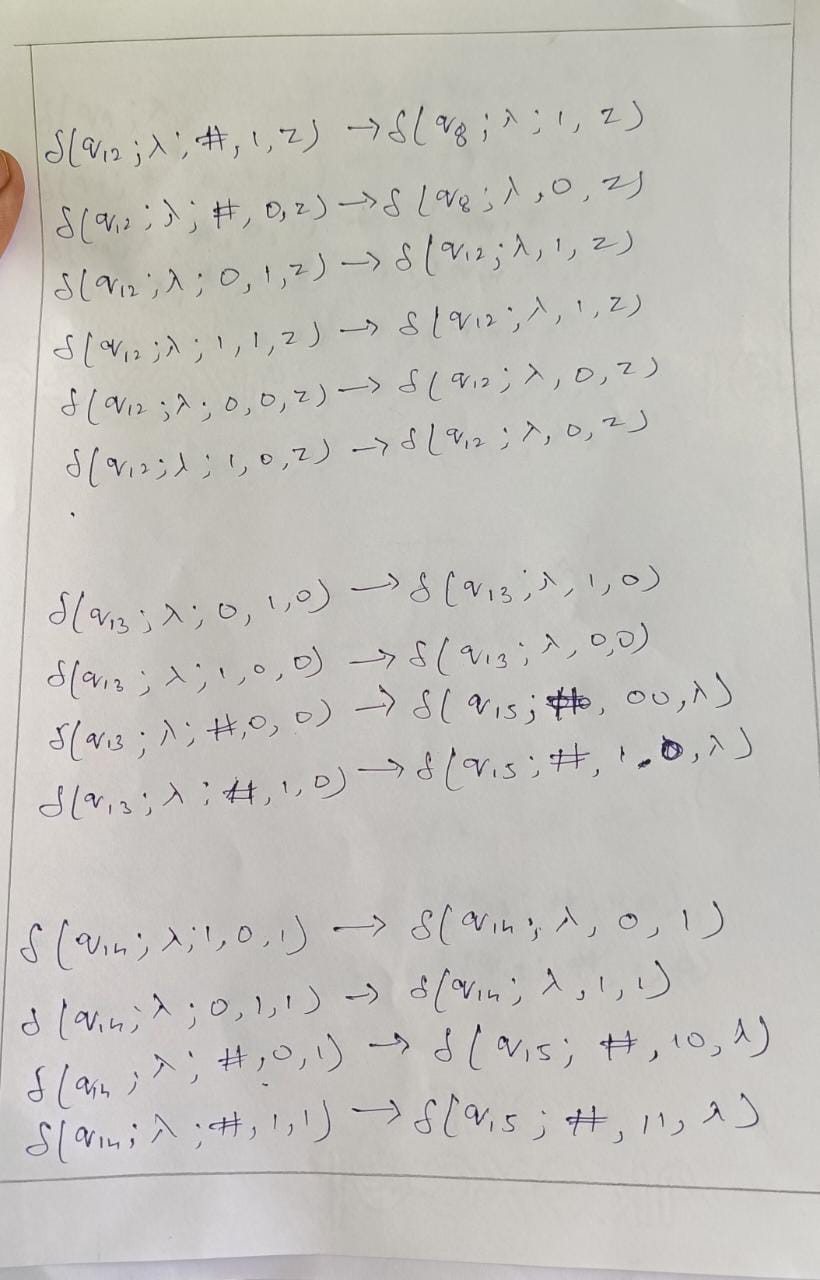
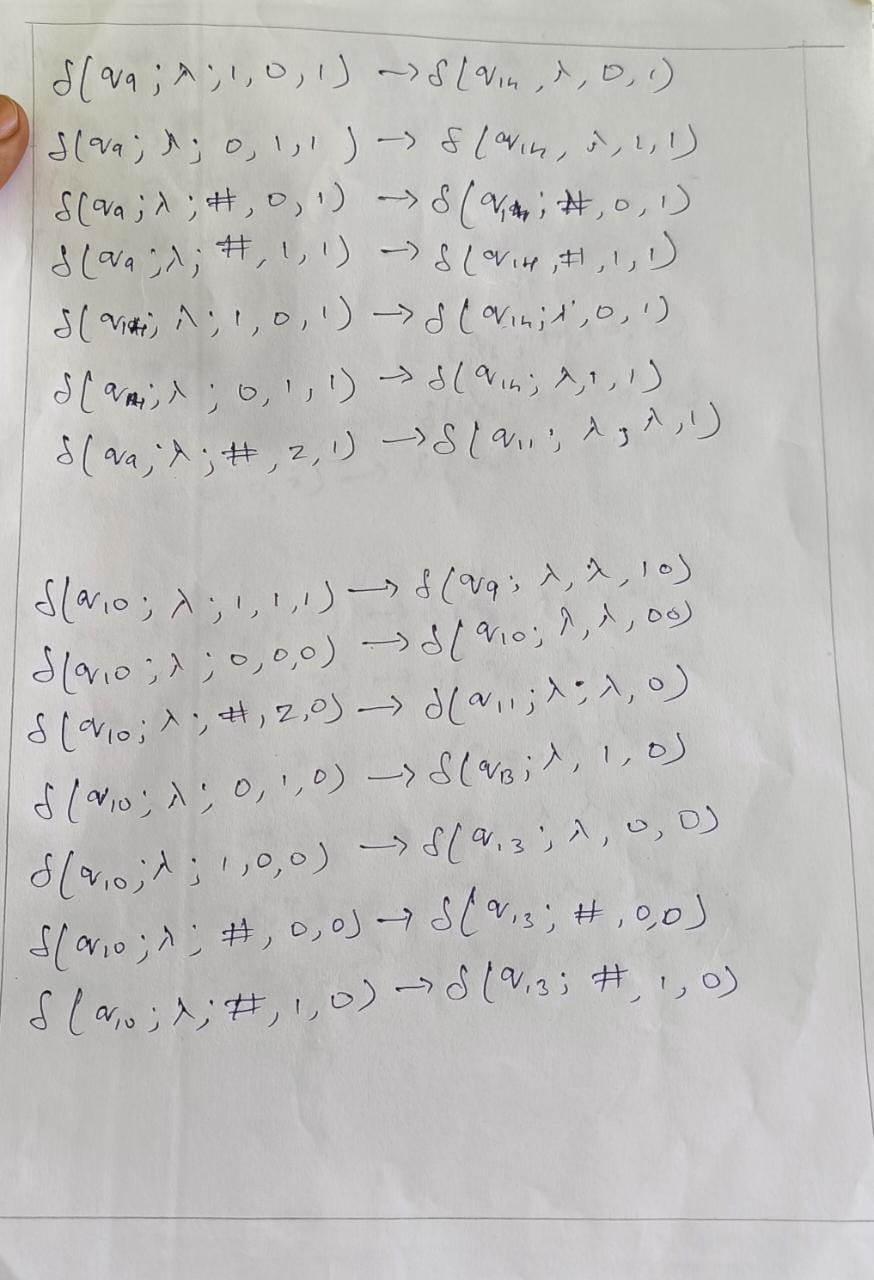
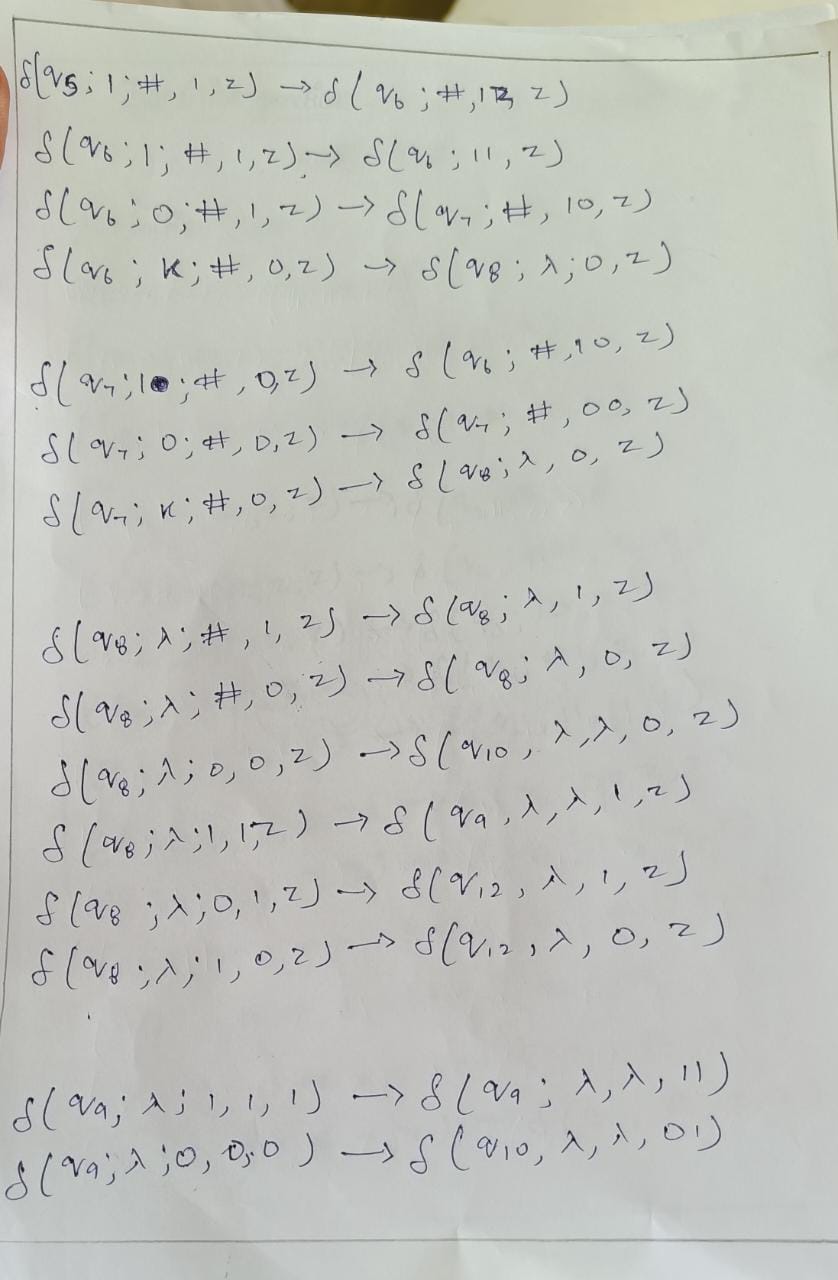
Σ = {0,1,#}

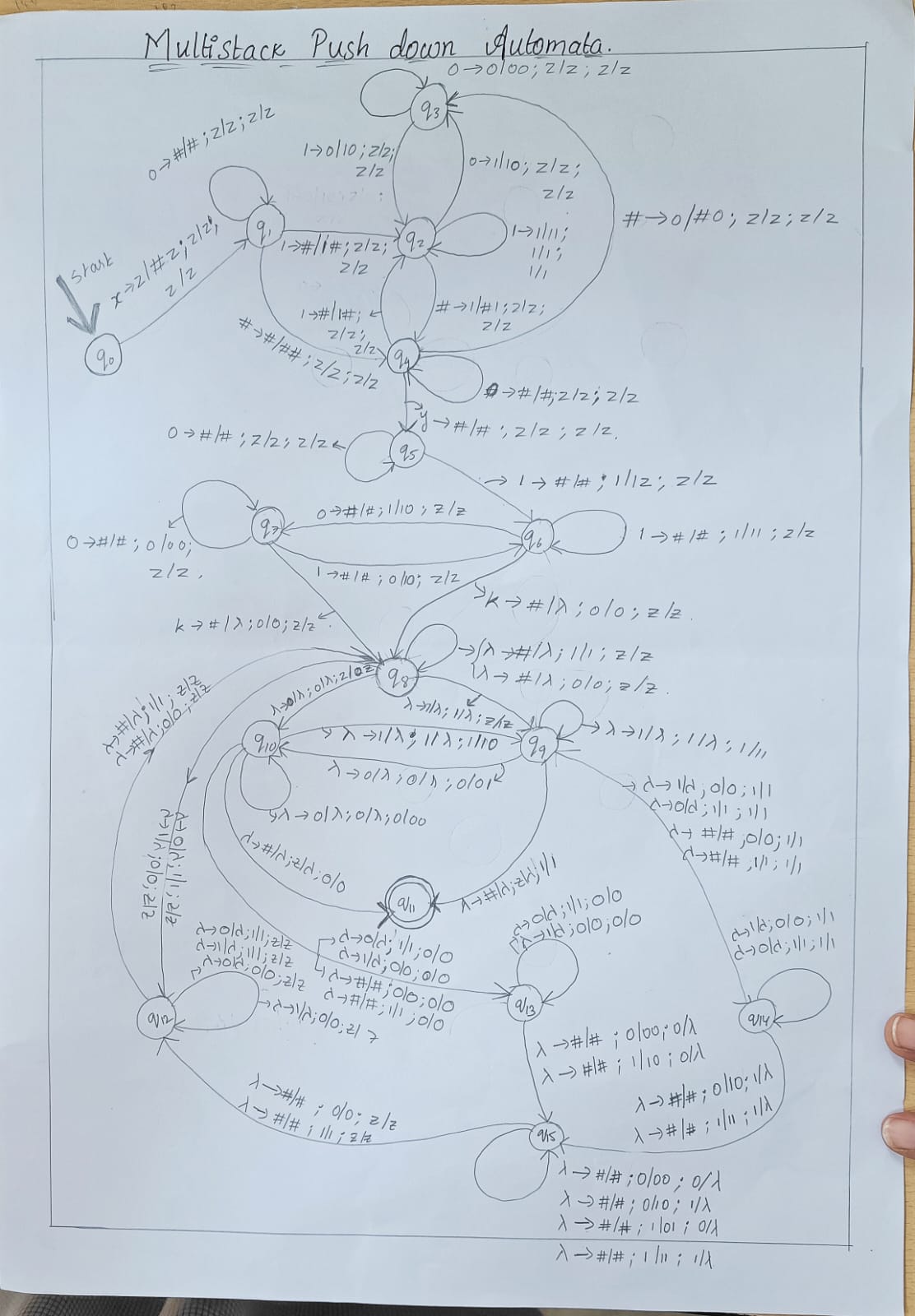
Language :

L=W1=wk | w2=wk | w3=wk ……..wn=wk , Wi € {0,1}+ i=1,2,3….n,Wk € {0,1}+

TRANSACTION FUNCTIONS







Turing Machine:

A Turing machine is a theoretical computational model introduced by Alan Turing. It consists of a tape divided into cells that can store symbols, a head that can read and write symbols on the tape, and a control unit that determines the machine's behavior based on the current state and the symbol being read. It can perform an infinite number of operations, making it a powerful model for solving algorithmic problems and representing general-purpose computation.

Multitape Turing Machine:

Multi-tape Turing Machines have multiple tapes where each tape is accessed with a separate head. Each head can move independently of the other heads. Initially the input is on tape 1 and others are blank. At first, the first tape is occupied by the input and the other tapes are kept blank. Next, the machine reads consecutive symbols under its heads and the TM prints a symbol on each tape and moves its heads.

A Multi-tape Turing machine can be formally described as a 6-tuple (Q, X, B, δ, q0, F) where −

* **Q** is a finite set of states
* **X** is the tape alphabet
* **[]** is the blank symbol
* **δ** is a relation on states and symbols where

δ: Q × Xk → Q × (X × {Left\_shift, Right\_shift, No\_shift })k

where there is **k** number of tapes

* **q1** is the initial state
* **F** is the set of final states

Working of multi tape turing machine:

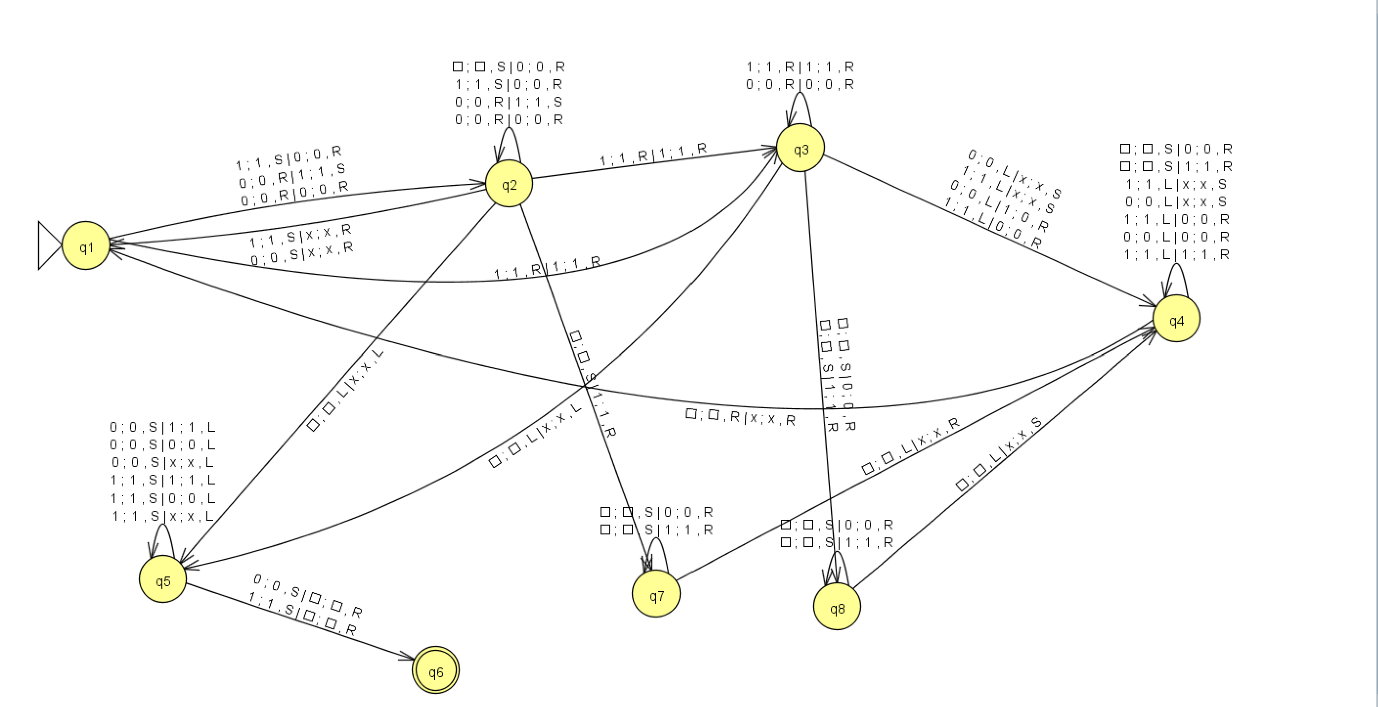
The search element inout will be tape1, and all the unique ids will be in the form of binary (0, 1) in tape 2. The elements are separated by the delimiter x. First, on tape 1, it will search for the element in tape 2, if not found, it will return to the starting head point of tape 1 and start searching for the next element in tape 2, and it will continue in tape until the element is found. If the element is found, it reaches the final state, or else the machine will be in a halt in the intermediate state.

Language

L=W1=wk | w2=wk | w3=wk ……..wn=wk , Wi € {0,1}+ i=1,2,3….n,Wk € {0,1}+

Alphabet Set

Σ = {0,1,x}



TRANSACTION FUNCTIONS

Initial state q1:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Current state | Input symbol | | Next state | Write symbol | | move | |
|  | Tape 1 | Tape 2 |  | Tape 1 | Tape 2 | Tape 1 | Tape 2 |
| Q1 | 0 | 0 | Q2 | 0 | 0 | R | R |
| Q1 | 0 | 1 | Q2 | 0 | 1 | R | S |
| Q1 | 1 | 0 | Q2 | 1 | 0 | S | R |
| Q1 | 1 | 1 | Q2 | 1 | 1 | R | R |

Intermediate state q2:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Current state | Input symbol | | Next state | Write symbol | | move | |
|  | Tape 1 | Tape 2 |  | Tape 1 | Tape 2 | Tape 1 | Tape 2 |
| Q2 | [] | 0 | Q2 | [] | 0 | S | R |
| Q2 | 1 | 0 | Q2 | 1 | 0 | S | R |
| Q2 | 0 | 1 | Q2 | 0 | 1 | R | S |
| Q2 | 0 | 0 | Q2 | 0 | 0 | R | R |
| Q2 | 1 | 1 | Q2 | 1 | 1 | R | R |
| Q2 | [] | X | Q5 | [] | X | L | L |
| Q2 | 1 | X | Q1 | 1 | X | S | R |
| Q2 | 0 | X | Q1 | 0 | X | S | R |
| Q2 | [] | 1 | Q7 | [] | 1 | S | R |

Intermediate state q3:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Current state | Input symbol | | Next state | Write symbol | | move | |
|  | Tape 1 | Tape 2 |  | Tape 1 | Tape 2 | Tape 1 | Tape 2 |
| Q3 | 1 | 1 | Q3 | 1 | 1 | R | R |
| Q3 | 0 | 0 | Q3 | 0 | 0 | R | R |
| Q3 | 0 | 1 | Q4 | 0 | 1 | R | R |
| Q3 | 1 | 0 | Q4 | 1 | 0 | L | R |
| Q3 | 0 | X | Q4 | 0 | X | L | S |
| Q3 | 1 | X | Q4 | 1 | X | L | S |
| Q3 | [] | X | Q5 | [] | X | L | L |
| Q3 | [] | 0 | Q8 | [] | 0 | S | R |
| Q3 | [] | 1 | Q8 | [] | 1 | S | R |

Intermediate state q4:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Current state | Input symbol | | Next state | Write symbol | | move | |
|  | Tape 1 | Tape 2 |  | Tape 1 | Tape 2 | Tape 1 | Tape 2 |
| Q4 | [] | 0 | Q4 | [] | 0 | S | R |
| Q4 | [] | 1 | Q4 | [] | 1 | S | R |
| Q4 | 1 | X | Q4 | 1 | X | L | S |
| Q4 | 0 | X | Q4 | 0 | X | L | S |
| Q4 | 1 | 0 | Q4 | 1 | 0 | L | R |
| Q4 | 0 | 0 | Q4 | 0 | 0 | L | R |
| Q4 | 1 | 1 | Q1 | 1 | 1 | L | R |
| Q4 | [] | X | Q4 | [] | X | L | L |

Intermediate state q5:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Current state | Input symbol | | Next state | Write symbol | | move | |
|  | Tape 1 | Tape 2 |  | Tape 1 | Tape 2 | Tape 1 | Tape 2 |
| Q5 | 0 | 1 | Q5 | 0 | 1 | S | L |
| Q5 | 0 | 0 | Q5 | 0 | 0 | S | L |
| Q5 | 0 | X | Q5 | 0 | X | S | L |
| Q5 | 1 | 1 | Q5 | 1 | 1 | S | L |
| Q5 | 1 | 0 | Q5 | 1 | 0 | S | L |
| Q5 | 1 | X | Q5 | 1 | X | S | L |
| Q5 | 0 | [] | Q6 | 0 | [] | S | R |
| Q5 | 1 | [] | Q6 | 1 | [] | S | R |

Intermediate state q7:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Current state | Input symbol | | Next state | Write symbol | | move | |
|  | Tape 1 | Tape 2 |  | Tape 1 | Tape 2 | Tape 1 | Tape 2 |
| Q7 | [] | 0 | Q7 | [] | 0 | S | R |
| Q7 | [] | 1 | Q7 | [] | 1 | S | R |
| Q7 | [] | X | Q4 | [] | X | R | R |

Intermediate state q8:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Current state | Input symbol | | Next state | Write symbol | | move | |
|  | Tape 1 | Tape 2 |  | Tape 1 | Tape 2 | Tape 1 | Tape 2 |
| Q8 | [] | 0 | Q8 | [] | 0 | S | R |
| Q8 | [] | 1 | Q8 | [] | 1 | S | R |
| Q8 | [] | X | Q4 | [] | X | L | S |

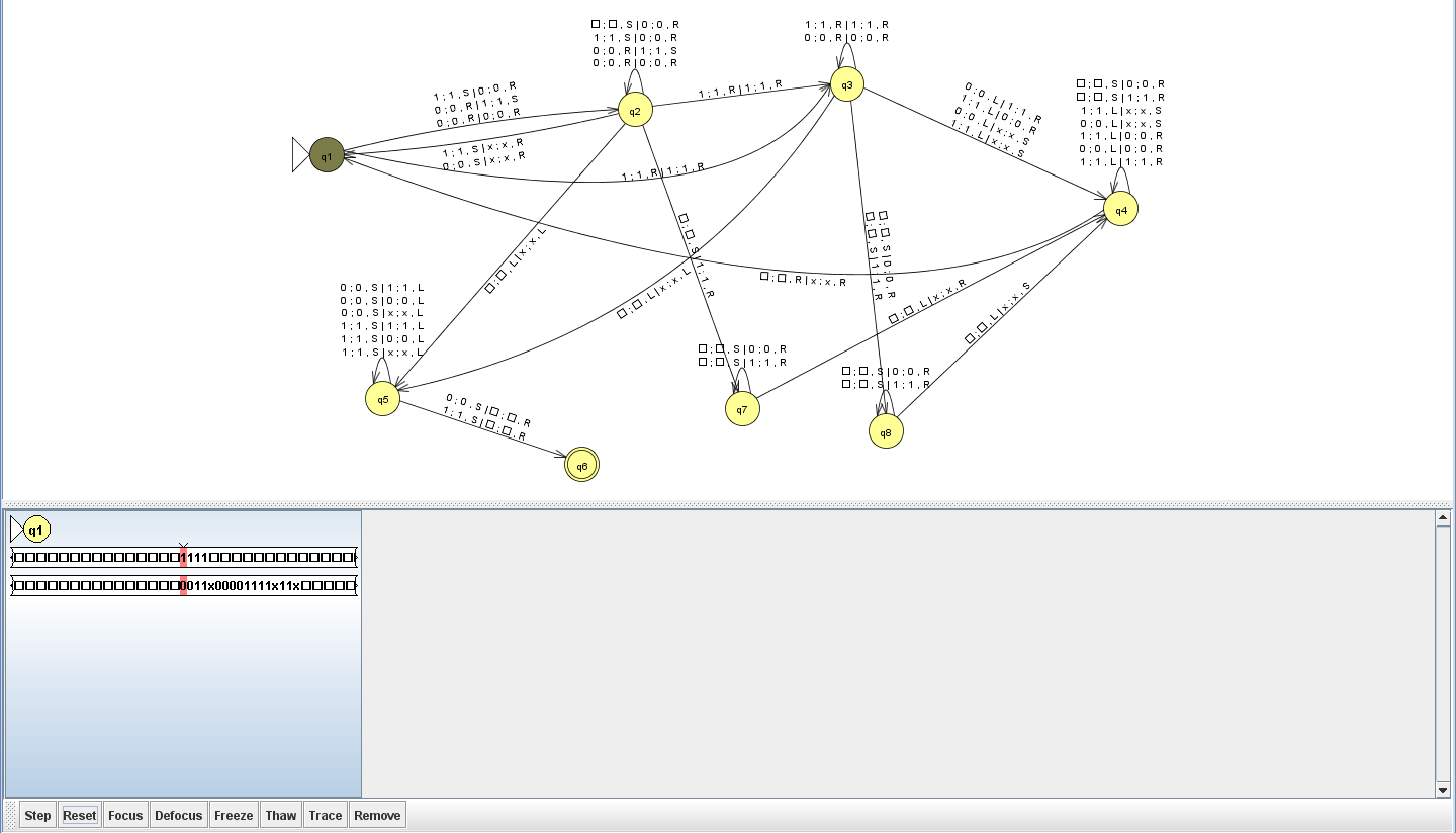
Input Tracing:

Tape 1 Input1: 1111

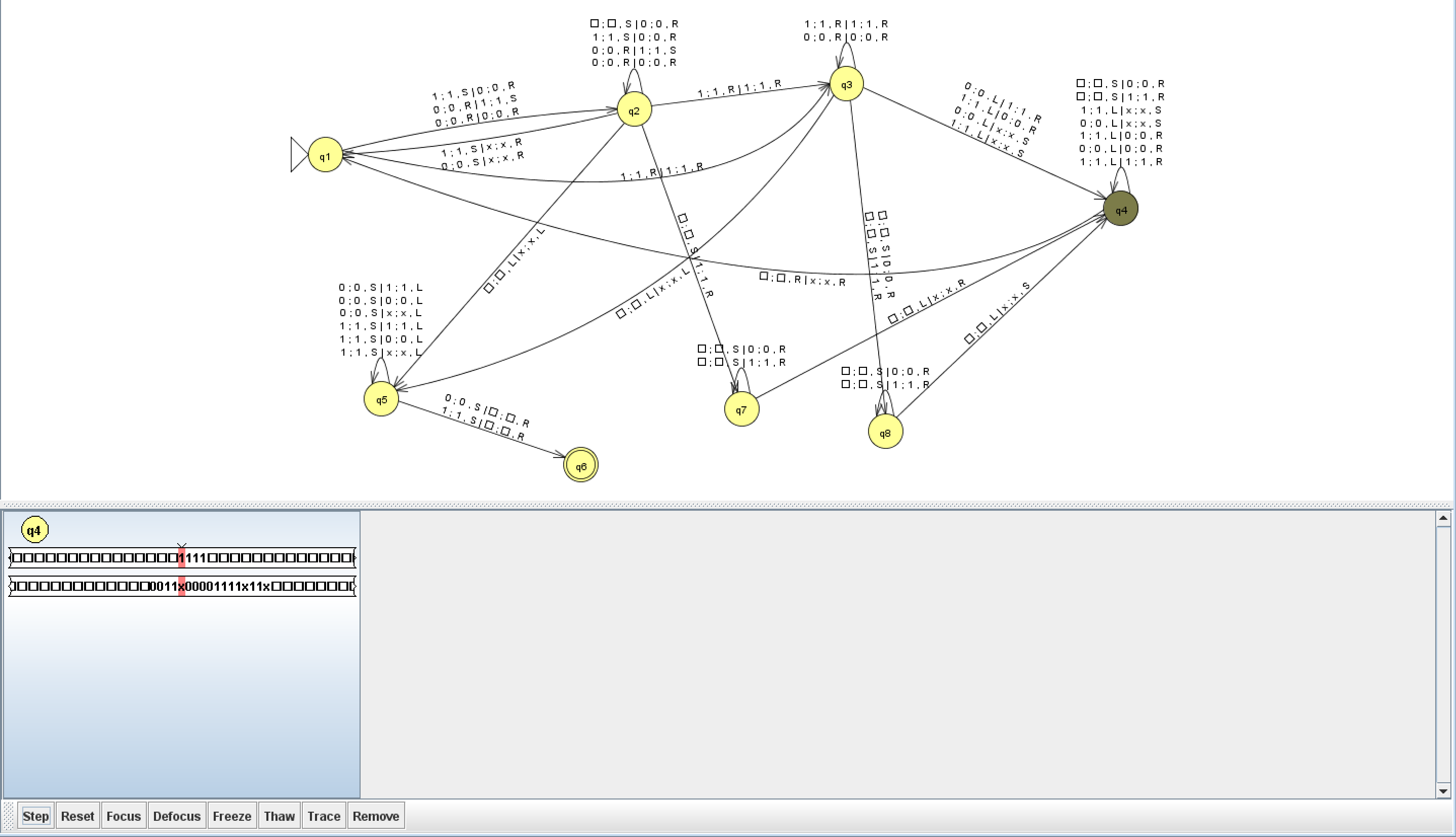
Tape 2 Input2:0011x00001111x11x

Tape1 contains the search element and Tape2 contains all the unique for the books

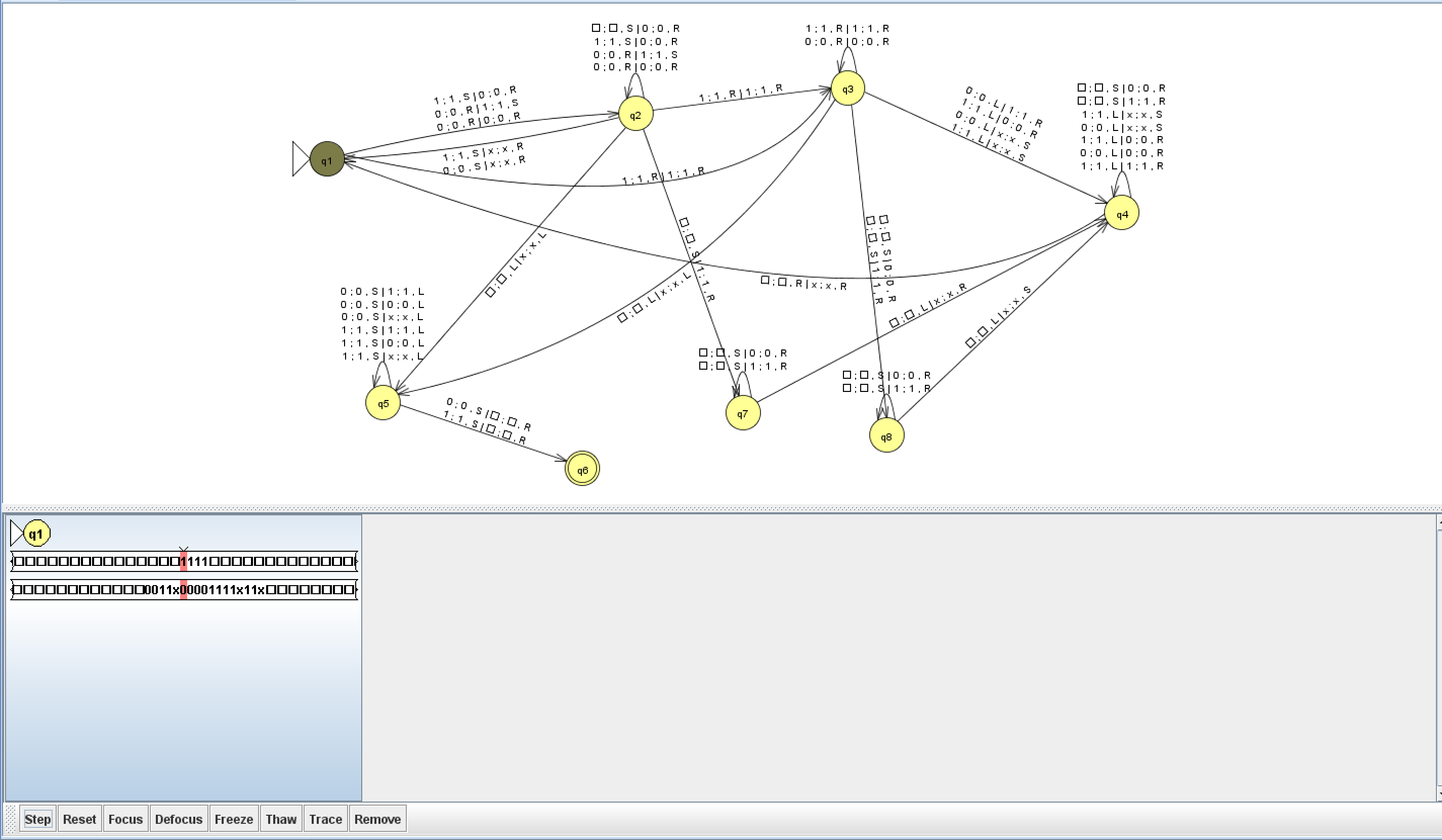
This image represents the initial state



A tape 2 is searched for an element, but the element is not found, thus the search is repeated back to the first state to see if the element is found in the other state.



When it returns to the original state after searching, tape 1 is shifted to the left until it receives the first digit of the search element.



Hence the search element is found and reached the final state and tape 2 pointer is bought to the left of the first tape head

