AMRITA VISHWA VIDYAPEETHAM

TOC CASE STUDY

GROUP NO: 7

TOPIC: X-ROAD TRAFFIC LIGHT SYSTEM

S.NO	NAME OF THE STUDENT	ROLL NUMBER
1	K GANESH	CB.EN.U4CSE21426
2	MITESH V	CB.EN.U4CSE21438
3	RAJKUMAR	CB.EN.U4CSE21446
4	VISWAA RAMSUBRAMANIAN	CB.EN.U4CSE21468
5	VIGNESH G	CB.EN.U4CSE21466

PROBLEM STATEMENT:

To think, read, analyse and implement a X-Road Traffic Light System using a Pushdown Automaton as well as a Turing Machine.

OVERVIEW:

The X Road Junction is a type of road intersection where 4 roads meet in an ‘X’ shape. It is formed when 2 roads meet at right angles to each other creating 4 different paths where each road is regulated by either a signal, yield signs, stop signs, sometimes even by a roundabout etc.. Each road represents a different path and approaching vehicles must navigate by choosing either of the 4 paths.

This type of configuration is commonly used in road networks in rural and urban areas of the state.

It accounts for the easier movement of traffic between intersecting roads.Here usually in India we follow a left-right lane system that is we drive on the left lane and the oncoming traffic is on the right lane, here in the X-road system when a car approaches a signal, if it wants to turn left it need not wait for the signal to turn green as most of the lefts are free lefts in India and in the US and Middle East there are free rights which account easier movement for vehicle which want to move left/right and also makes it easier for pedestrian to cross the road.
It is also easier for the drivers to choose because there are fewer paths which reduces the decision errors made by drivers.

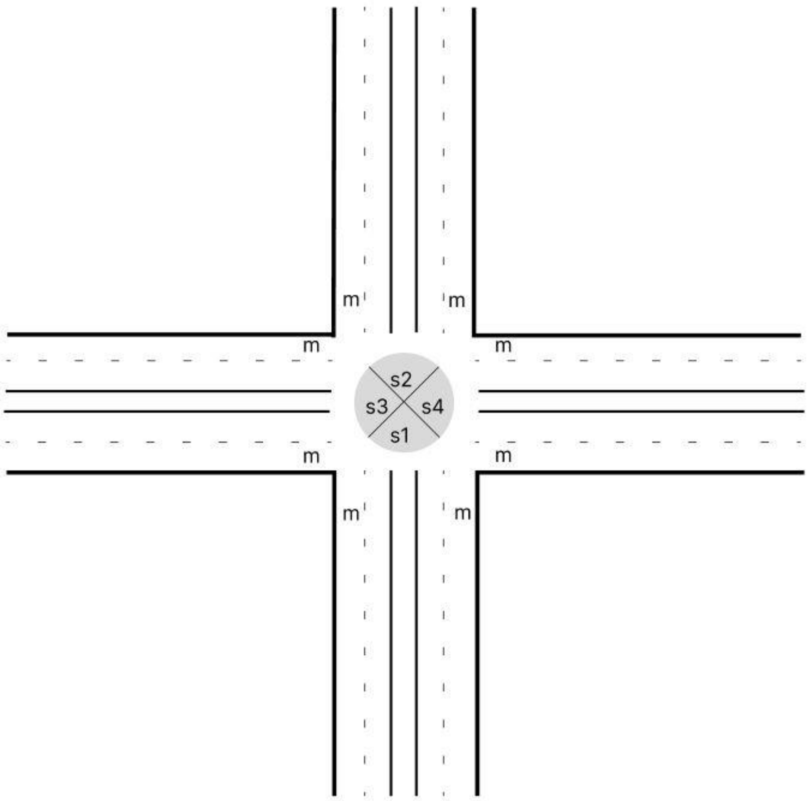
It is considered one of the best measures to control traffic be it either high level or low level because their nature is quite similar to round robin cpu scheduling where each road gets an equal priority to be given a green signal.
The visibility range for the driver is also quite high because the roads meet at right angles, the driver can see the traffic in the oncoming lane and make safer decisions while crossing the junction.
The cross-road pattern requires less space compared to other complex road networks which turns out to be one of the greatest advantages in densely populated zones/regions where the lying area is very small and X-road system proves to be the best to be laid there.
Since their design is quite simple and is easy to implement, they have low-lying as well as maintenance costs compared to other road networks.

The case study presents the solution to the problem of a cross-road junction where 4 roads meet each other, which is regulated by 4 signals, 1 for each road. The main objective is to design different computational models which can simulate the behaviour of the junction including the control of traffic signals and devise other strategies for the efficient handling of traffic.
Some practical applications of the solutions devised by such computational models include:

- Optimising Traffic Flow: By modelling the behaviour of the junction, the models can be of great help in improving traffic flow by identifying highly effective signal control techniques, for minimising congestion, and reducing travelling times.

- Giving Priority To Emergency Vehicles such as ambulances: By dynamically modifying the signal timings, it ensures the smooth passage of emergency vehicles, such as ambulances.
- Analysis Of Safety Measures: They offer their assistance in making safety analysis by simulating various scenarios, identifying potential conflicts or dangerous circumstances, creating new safety protocols, and enhancing junction design.
- Formulating New Policies for The State: They provide a template to assess the traffic rules and regulations and for framing new traffic laws for the state.

HOW AN X ROAD JUNCTION LOOKS LIKE



Here the symbol ‘m’ denotes the emergency lanes for ambulances. This lane is for ambulances only and no other vehicle is permitted to travel in this lane.

IMPLEMENTATION OF X-ROAD SYSTEM USING PUSHDOWN AUTOMATA

We have designed a NPDA(Non-Deterministic Pushdown Automata) which models the X-Road traffic light system problem.

Since we have 4 signals, we associate the symbols s1,s2,s3,s4 which correspond to each of the four signals 1,2,3,4 respectively. Here s1 and s2 regulate 2 opposite straight roads while s3,s4 regulate the same for another pair of opposite roads.

At each signal, there are 2 possible options : to go straight or to go right. Since left turns are usually free in India, we don't consider the case where the vehicle turns left.
For example if a vehicle wants to go straight at signal s1, we associate the symbol ‘st1’ which ensures that the vehicle moves straight in the signal, whereas if it wants to go right, we associate the symbol ‘rt1’ which enables the right turn at the signal s1.
The symbol ‘w’ denotes a wait operation which pops the top most element from the stack and the symbol ‘s’ denotes a stop which causes to pop 2 elements from the stack and push back the element which got popped first from the stack. The symbol ‘s’ is designed in such a way that it regulates traffic for two opposite roads by ensuring that there is no additional wait time for the signal which is to be opened.
Both the functions associated with the symbols ‘w’ and ‘s’, work based on timers.

The clock-cycle time , that is the total time taken for the signal pattern to repeat again is 180 seconds

Initially the cycle starts when the timer reads 0 seconds and signal s1 is opened.
When the timer reads 30 seconds a wait operation ‘w’ is executed.
When the timer ticks to 60 seconds, a stop operation ‘s’ is called so that the signal s2 is opened now.
When it reaches 90 seconds ‘w’ is called twice to stop the current section of the X-road system which facilitates the opening of the other pair of opposite roads namely s3,s4.
First s3 is opened now and when the timer reaches 120 seconds, wait operation ‘w’ is called.
Lastly, to open the signal s4, a stop operation ‘s’ is called when the timer reads 150 seconds.
When the timer reaches 180 seconds, ‘w’ is called twice to where the timer resets to 0 seconds and this cycle repeats itself again starting from signal s1.

Stop operation here denotes 2 wait ‘w’ operations which are executed back to back.

During pop operation signal changes to red, whereas during push operation signal changes to green.

SPECIAL CASE:

The special case comes into play when an ambulance is approaching the cross-road junction. We have a special lane known as ambulance lane just like the ones in a railway track system in which there is at least one lane usually called the freight lane, which is operated for trains which carry goods/mails. No other type of train is allowed to operate on this lane, that is the signal on this lane is always red and when such a train approaches only then this track is released. In the same way, this lane is operated especially for ambulances and is opened only when the ambulance approaches the junction.

In such a case, the flow of the system is halted and the stack contents are cleared which ensures that the ambulance can reach its destination without any hindrance. The inclusion of a separate lane for ambulances proves to be beneficial and may even help in saving a poor victim’s life.

This can also be achieved by making the signal which leads to the destination of the ambulance to be green. This is also possible but there might be slight delay as the vehicles who wish to approach in that direction will remain ahead of the ambulance and cause some delay for the ambulance to sail past ahead. This can be an issue because we humans don't have the patience and also try to move even in the smallest gap we have in those signals which might become problematic for the ambulances.

India is slowly adapting european style for maintaining separate lanes for ambulances, and this system might be in full effect by 2030. It has already been implemented in some metro cities like Delhi, Bombay and the results are proven to be fruitful as well.

SYMBOLS ASSOCIATED WITH THE NPDA:

Since there are 4 signals s1,s2,s3 and s4 and each signal has 2 directions which are being controlled by the timer, we have associated the following symbols for each signal

Signal S1: a->denotes straight signal at s1 is green
p->denotes right signal at s1 is green

Signal S2: b->denotes straight signal at s1 is green
q->denotes right signal at s1 is green

Signal S3: c->denotes straight signal at s1 is green r->denotes right signal at s1 is green

Signal S4: d->denotes straight signal at s1 is green
s->denotes right signal at s1 is green

The symbol ‘m’ denotes the situation where an ambulance is approaching the cross-road junction.
In this case, we have to clear the contents of the stack, that is we have to make all the signals red and make sure that the ambulance sails ahead without any hindrance.
If you encounter any input after m which is not the symbol ‘a’, we do not push or pop anything and stay in the same state. After the ambulance has left the junction, the execution starts from first, that is it starts from signal s1The NPDA accepts a signal direction starting from signal s1, that is a straight signal followed by a right turn for the same signal which is pushed onto the stack.
When the next signal is opened, we pop
The NPDA accepts both the cases, one is handling an ambulance and the other when no ambulance signal is involved.

When the symbol ‘m’ is encountered in the input string, the NPDA clears the contents of the stack and the execution starts from first, that is the clock timer is reset to zero seconds, which is in accordance with the real world as well. This ensures that the system does not suffer unnecessary wait time.

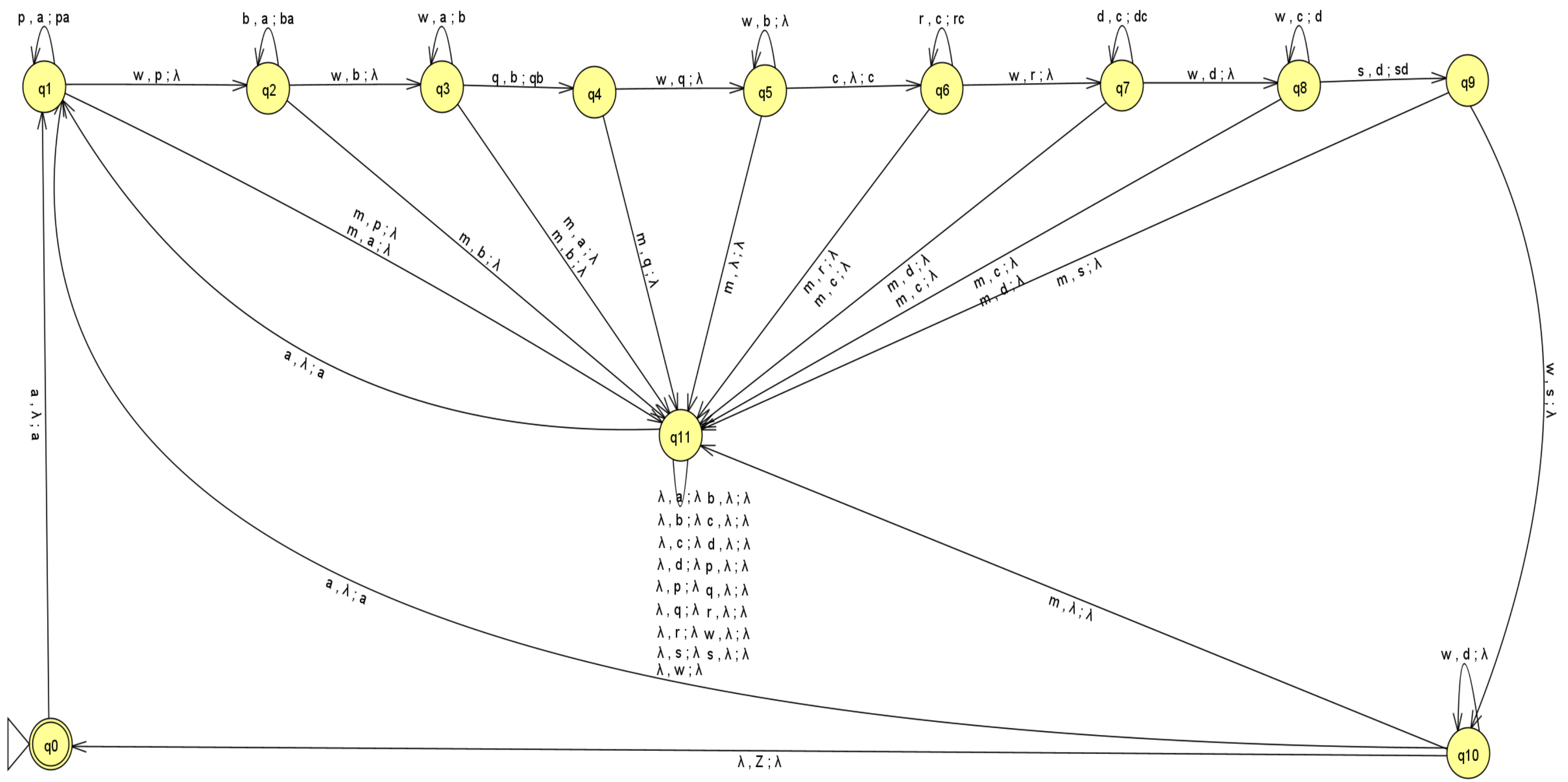
LANGUAGE

$$L=\{(am^*pm^*wm^*bm^*wm^*wm^*qm^*wm^*wm^*cm^*rm^*wm^*dm^*wm^*wm^*sm^*wm^*wm^*)^n : n \geq 0\}$$

TRANSITIONS FOR NPDA:

Current State	Input Symbol	Top of the stack	operation	Next state	Current top of the stack
q0	a	Z	push	q1	aZ
q1	p	a	push	q1	pa
q1	w	p	pop	q2	a
q2	b	a	push	q2	ba
q2	w	b	pop	q3	a
q3	w	a	Pop and push	q3	b
q3	q	b	push	q4	qb
q4	w	q	pop	q5	b
q5	w	b	pop	q5	Z
q5	c	Z	push	q6	cZ
q6	r	c	push	q6	rc
q6	w	r	pop	q7	c
q7	d	c	push	q7	dc
q7	w	d	pop	q8	c
q8	w	c	Push and pop	q8	d
q8	s	d	push	q9	sd
q9	w	s	pop	q10	d
q10	w	d	pop	q10	Z
q10	λ	Z	No push/pop	q11	Z

NPDA DESIGN



q0
q1
q2
q3
q4
q5
q6
q7
q8
q9
q10
q11
p , a ; pa
w , p ; λ
b , a ; ba
w , b ; λ
w , a ; b
q , b ; qb
w , q ; λ
w , b ; λ
c , λ ; c
r , c ; rc
w , r ; λ
d , c ; dc
w , d ; λ
w , c ; d
s , d ; sd
w , s ; λ
w , d ; λ
λ , Z ; λ
a , λ ; a
a , λ ; a
a , λ ; a
m , p ; λ
m , a ; λ
m , b ; λ
m , a ; λ
m , b ; λ
m , q ; λ
m , λ ; λ
m , r ; λ
m , c ; λ
m , d ; λ
m , c ; λ
m , c ; λ
m , d ; λ
m , s ; λ
m , λ ; λ
λ , a ; λ b , λ ; λ
λ , b ; λ c , λ ; λ
λ , c ; λ d , λ ; λ
λ , d ; λ p , λ ; λ
λ , p ; λ q , λ ; λ
λ , q ; λ r , λ ; λ
λ , r ; λ w , λ ; λ
λ , s ; λ s , λ ; λ
λ , w ; λ

INPUTS

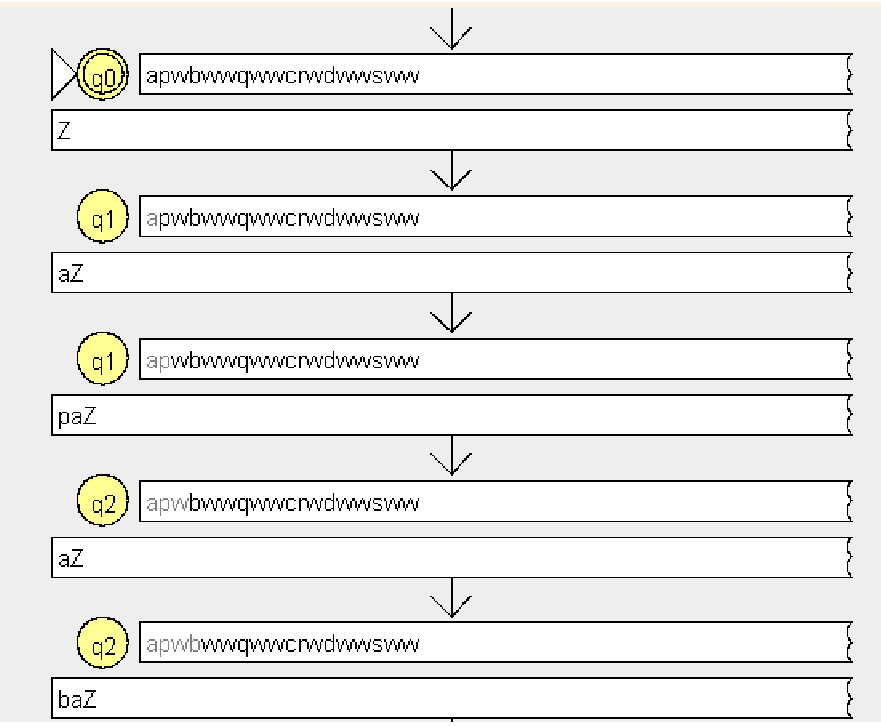
- Without Ambulance “m”
  - apwbwwqwwcrwdwwsww
  - apwbwwqwwcrwdwwswwapwbwwqwwcrwdwwsww
  - apwbwwwqwwcrwdwwsww
- With Ambulance “m”
  - apwbwwqwwmcrwdwwswwapwbwwqwwcrwdwwsww
  - apwbwwqwwmcrwdwwswwapwbmwwqwwcrwdwwswwapwbwwqwwcrwdwwsww

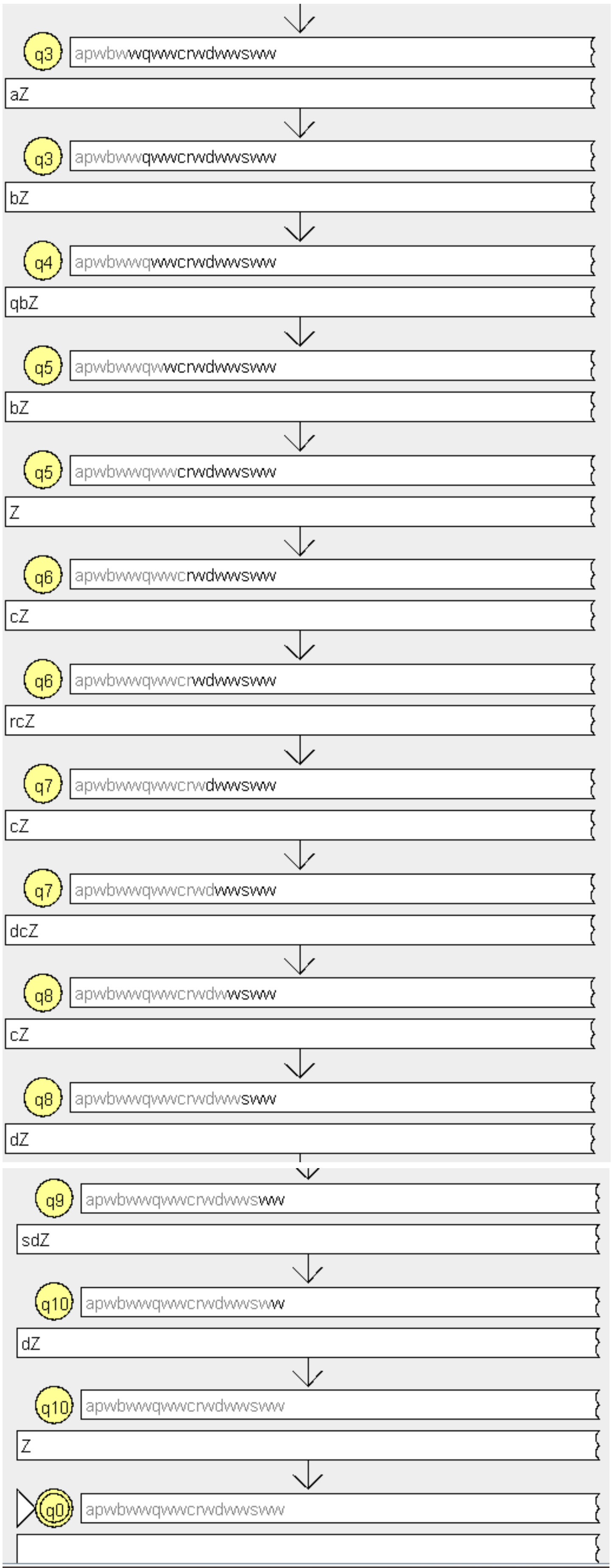
OUTPUT

Table Text Size

Input	
apwbwwqwwcrwdwwsww	Accept
apwbwwqwwcrwdwwswwapwbwwqwwcrwdwwsww	Accept
apwbwwqwwmcrwdwwswwapwbwwqwwcrwdwwsww	Accept
apwbwwqwwmcrwdwwswwapwbmwwqwwcrwdwwswwapwbwwqwwcrwdwwsww	Accept
apwbwwqwwmcrwdwwswwapwbmwwqwwcrwdwwsww	Reject
apwbwwwqwwcrwdwwsww	Reject

Stack Representation for apwbwwqwwcrwdwwsww





q3
apwbwwqwwcrwdwwsww
aZ
q3
apwbwwqwwcrwdwwsww
bZ
q4
apwbwwqwwcrwdwwsww
qbZ
q5
apwbwwqwwcrwdwwsww
bZ
q5
apwbwwqwwcrwdwwsww
Z
q6
apwbwwqwwcrwdwwsww
cZ
q6
apwbwwqwwcrwdwwsww
rcZ
q7
apwbwwqwwcrwdwwsww
cZ
q7
apwbwwqwwcrwdwwsww
dcZ
q8
apwbwwqwwcrwdwwsww
cZ
q8
apwbwwqwwcrwdwwsww
dZ
q9
apwbwwqwwcrwdwwsww
sdZ
q10
apwbwwqwwcrwdwwsww
dZ
q10
apwbwwqwwcrwdwwsww
Z
q0
apwbwwqwwcrwdwwsww

DESIGN EXPLANATION BY NPDA

NPDA is defined by the set of 6 tuples (Q,Σ,δ,q0,qf) where

- Q-> set of all possible internal states
  Here in our NPDA we have the following internal states{q1,q2,q3,q4,q5,q6,q7,q8,q9,q10}
- Sigma here represents the alphabet set we use as input in our NPDA. Here sigma consists of the following symbols {a,b,c,d,n,p,q,r,s,w}
- The greek symbol gamma represents the set of alphabets present in the stack memory.
- δ represents the transition function.
  δ(q1,x,Z)=(q2,y) where q1 is the current state , x is the input symbol,Z is the current top most element in the stack, q2 is the next state, y is the current top most element in the stack.
- q0 represents the starting state of the NPDA.
- qf represents the final state of the NPDA.

Here the terminals are:
a,b,c,d,m,p,q,r,s,w

SOLUTION USING TURING MACHINE:

A Turing Machine(TM) is a special type of machine which uses a tape of infinite length, with multiple sections/cells and has a read/write head which can be used to perform operations in these sections.
It is a very powerful deterministic machine which solves complex problems across various domains.
It constitutes 7 tuples (Q,T,Σ,δ,q0,▢,qf)

- Q is the set of all internal states in the turing machine.
  In our design the turing machine has the following states {q0,q1,q2,q3,q4,q5,q6,q7,q8,q9,q10,q11,q12,q13,q14,
  q15,q16,q17,q18,q19,q20q,21,q22,q23,q24,q25,q26,q27}
- T denotes the input tape and ▢ is the blank symbol.
- Sigma represents the set of all input characters.
- δ denotes the partial transition function.
- q0 is the initial state and qf is the final state.

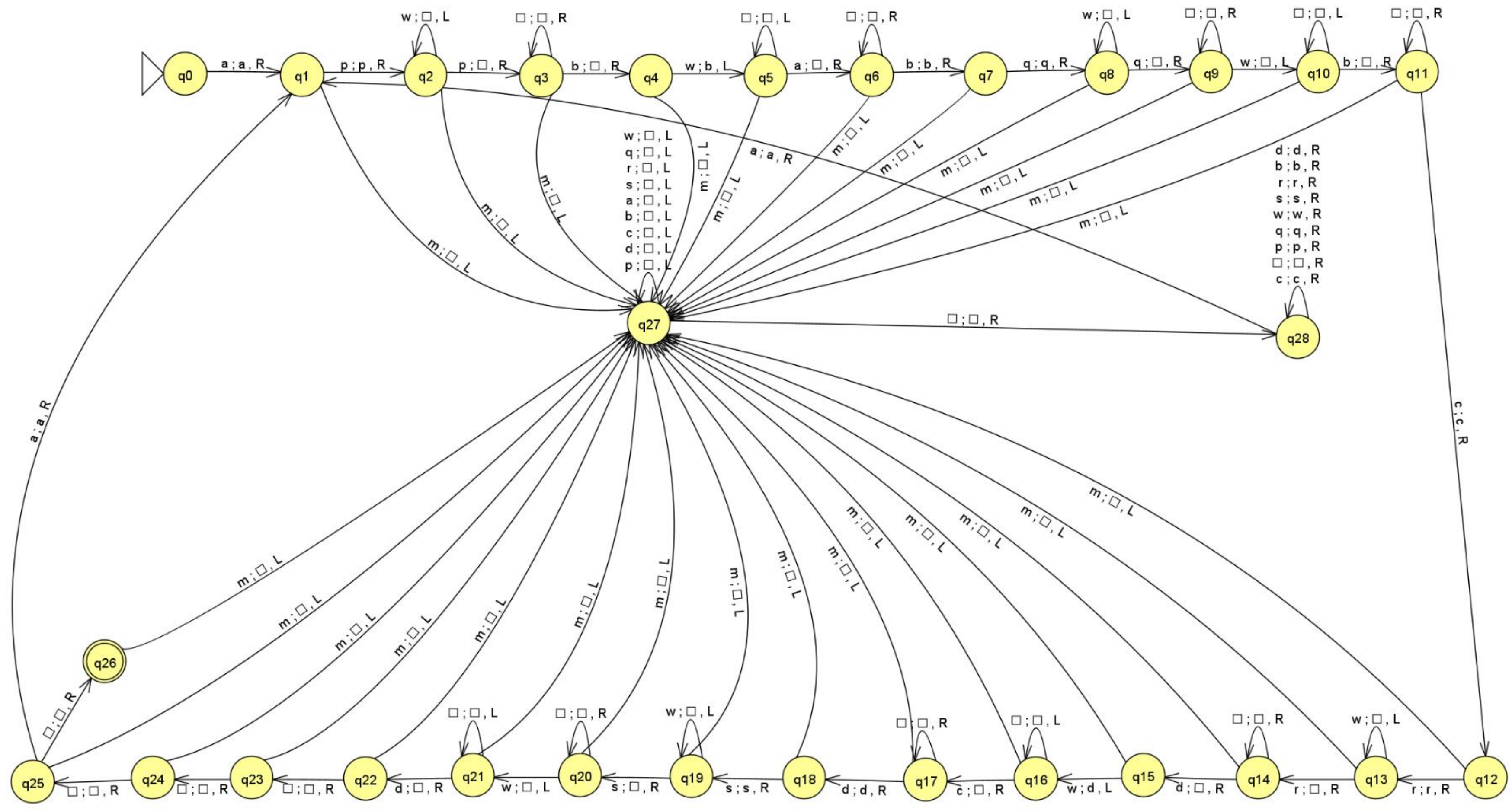
TRANSITIONS FOR SINGLE TAPE TURING MACHINE:

Below is the transition table for single tape turing machine

Current state	Input symbol	Symbol to write	Direction To move	Next state
q0	a	a	R	q1
q1	p	p	R	q2
q2	w	▢	L	q2
q2	p	▢	R	q3
q3	▢	▢	R	q3
q3	b	▢	R	q4
q4	w	b	L	q5
q5	▢	▢	L	q5
q5	a	▢	R	q6
q6	▢	▢	R	q6
q6	b	b	R	q7
q7	q	q	R	q8
q8	w	▢	L	q8
q8	q	▢	R	q9
q9	▢	▢	R	q9
q9	w	▢	L	q10
q10	▢	▢	L	q10
q10	b	▢	R	q11
q11	▢	▢	R	q11
q11	c	c	R	q12
q12	r	r	R	q13

q13	w	□	L	q13
q13	r	□	R	q14
q14	□	□	R	q14
q14	d	□	R	q15
q15	w	d	L	q16
q16	□	□	L	q16
q16	c	□	R	q17
q17	□	□	R	q17
q17	d	d	R	q18
q18	s	s	R	q19
q19	w	□	L	q19
q19	s	□	R	q20
q20	□	□	R	q20
q20	w	□	L	q21
q21	□	□	L	q21
q21	d	□	R	q22
q22	□	□	R	q23
q23	□	□	R	q24
q24	□	□	R	q25
q25	□	□	R	q26

SINGLE TAPE TURING MACHINE DESIGN



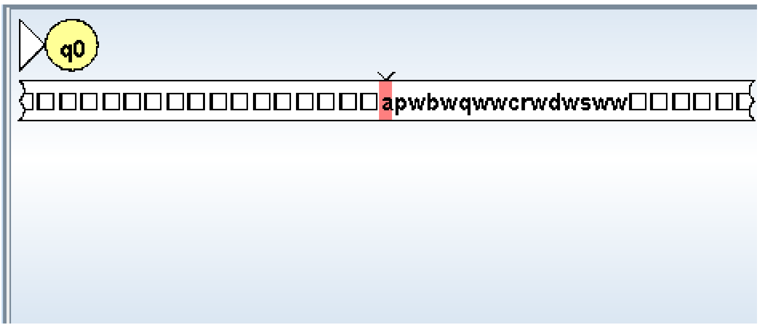
INPUTS

- Without Ambulance “m”
  - apwbwqwwcrwdwsww
  - apwbwqwwcrwdwswwapwbwqwwcrwdwsww
  - apwbwqwwcrwdwswwdfg
- With Ambulance “m”
  - apwbwqwmwcrwdwswwapwbwqwwcrwdwsww
  - apwbwqwwcrmwdwsww

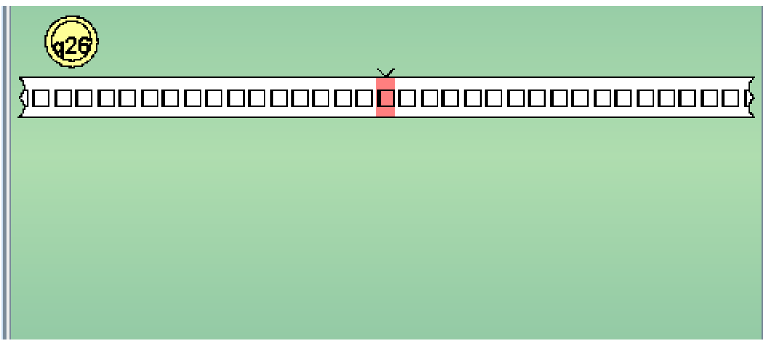
Table Text Size

Input	Output	Result
apwbwqwwcrwdwsww		Accept
apwbwqwwcrwdwswwapwbwqwwcrwdwsww		Accept
apwbwqwmwcrwdwswwapwbwqwwcrwdwsww		Accept
apwbwqwwcrwdwswwdfg		Reject
apwbwqwwcrmwdwsww		Cancelled

INPUT



OUTPUT



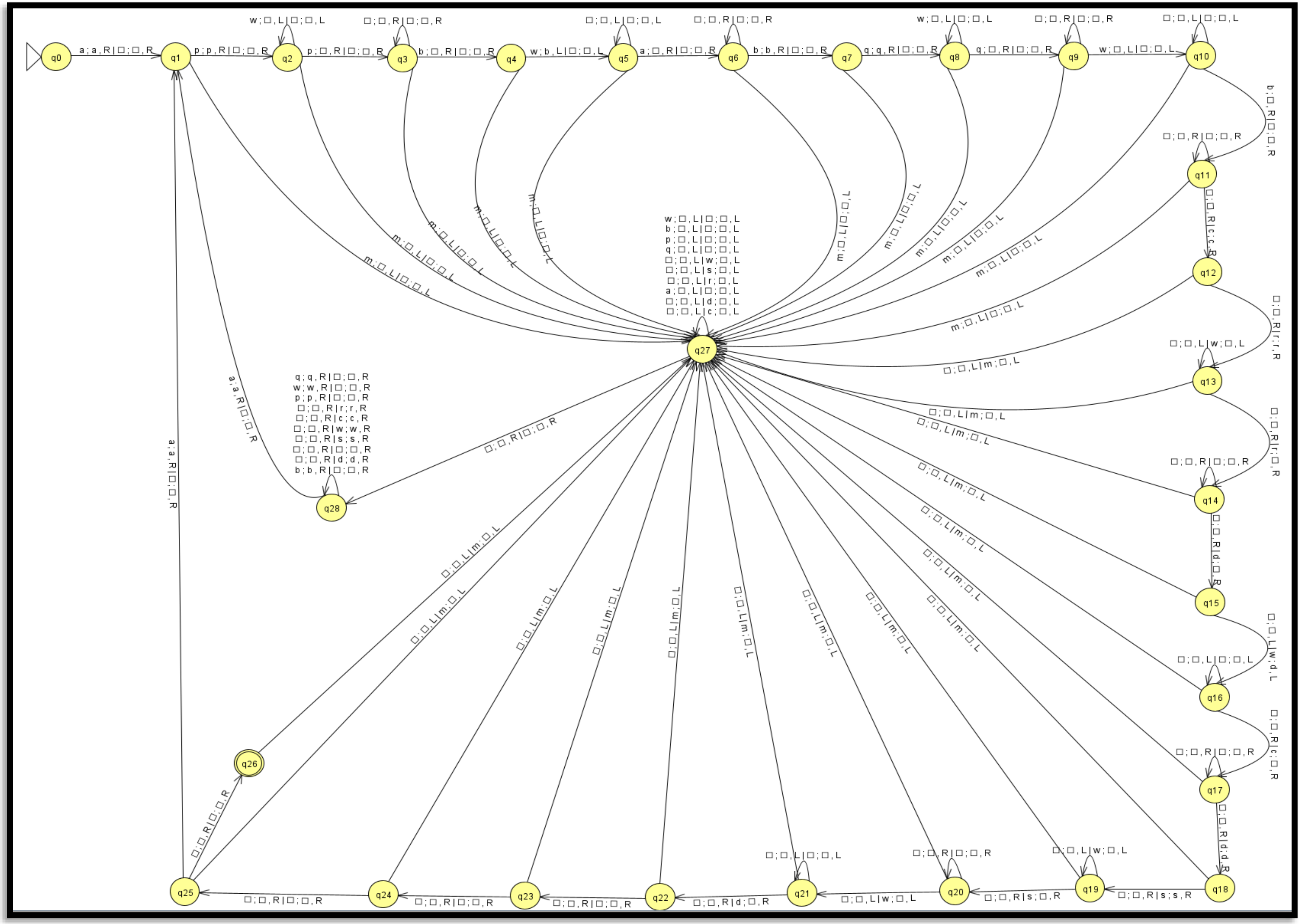
TRANSITIONS FOR MULTITAPE TAPE TURING MACHINE:

Below is the transition table for single tape turing machine

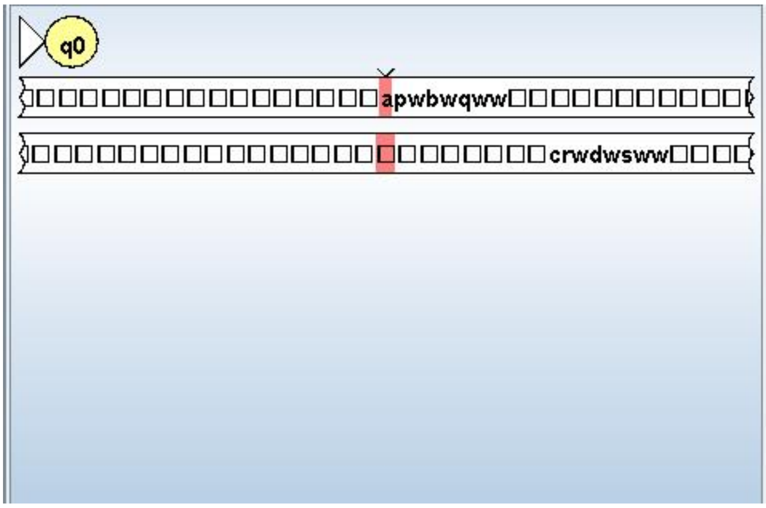
CS	I/P	W1	D1	I/P 2	W2	D2	NS
q0	a	a	R	▢	▢	R	q1
q1	p	p	R	▢	▢	R	q2
q2	w	▢	L	▢	▢	L	q2
q2	p	▢	R	▢	▢	R	q3
q3	▢	▢	R	▢	▢	R	q3
q3	b	▢	R	▢	▢	R	q4
q4	w	b	L	▢	▢	L	q5
q5	▢	▢	L	▢	▢	L	q5
q5	a	▢	R	▢	▢	R	q6
q6	▢	▢	R	▢	▢	R	q6
q6	b	b	R	▢	▢	R	q7
q7	q	q	R	▢	▢	R	q8
q8	w	▢	L	▢	▢	L	q8
q8	q	▢	R	▢	▢	R	q9
q9	▢	▢	R	▢	▢	R	q9
q9	w	▢	L	▢	▢	L	q10
q10	▢	▢	L	▢	▢	L	q10
q10	b	▢	R	▢	▢	R	q11
q11	▢	▢	R	▢	▢	R	q11

q11	▢	▢	R	c	c	R	q12
q12	▢	▢	R	r	r	R	q13
q13	▢	▢	L	w	▢	L	q13
q13	▢	▢	R	r	▢	R	q14
q14	▢	▢	R	▢	▢	R	q14
q14	▢	▢	R	d	▢	R	q15
q15	▢	▢	L	w	d	L	q16
q16	▢	▢	L	▢	▢	L	q16
q16	▢	▢	R	c	▢	R	q17
q17	▢	▢	R	▢	▢	R	q17
q17	▢	▢	R	d	d	R	q18
q18	▢	▢	R	s	s	R	q19
q19	▢	▢	L	w	▢	L	q19
q19	▢	▢	R	s	▢	R	q20
q20	▢	▢	R	▢	▢	R	q20
q20	▢	▢	L	w	▢	L	q21
q21	▢	▢	L	▢	▢	L	q21
q21	▢	▢	R	d	▢	R	q22
q22	▢	▢	R	▢	▢	R	q23
q23	▢	▢	R	▢	▢	R	q24
q24	▢	▢	R	▢	▢	R	q25
q25	▢	▢	R	▢	▢	R	q26

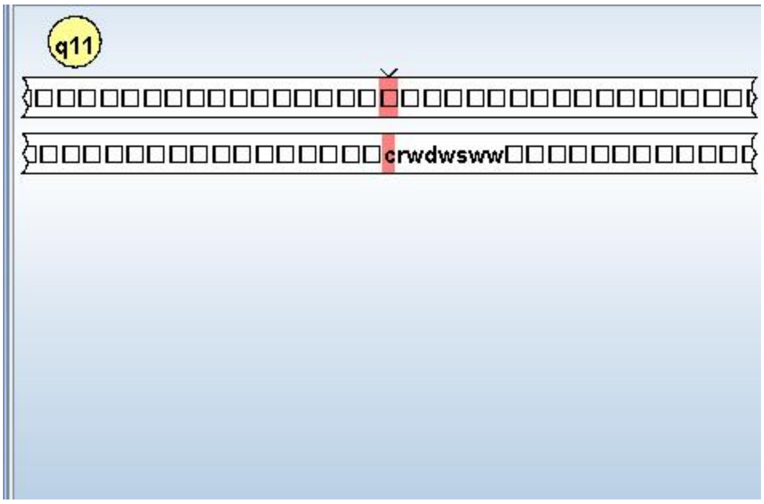
MULTIPLE TAPE TURING MACHINE DESIGN



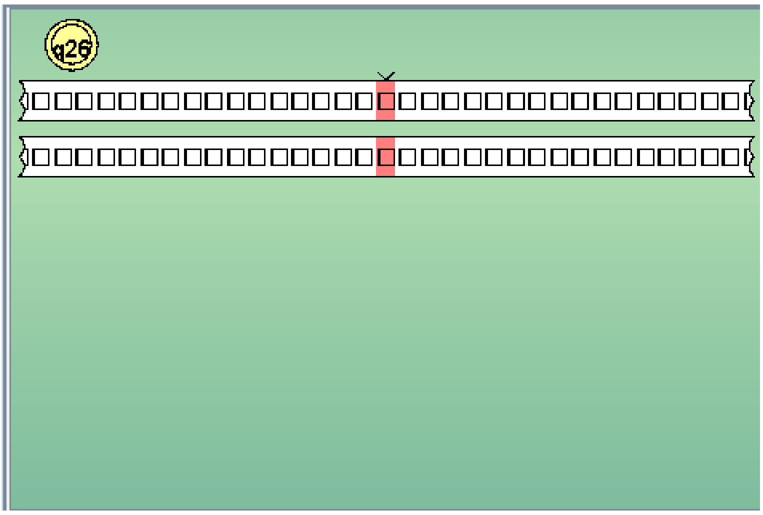
INITIAL STATE



IN MIDDLE TAPE CHANGE



FINAL



TURING MACHINE SIMULATOR CODE

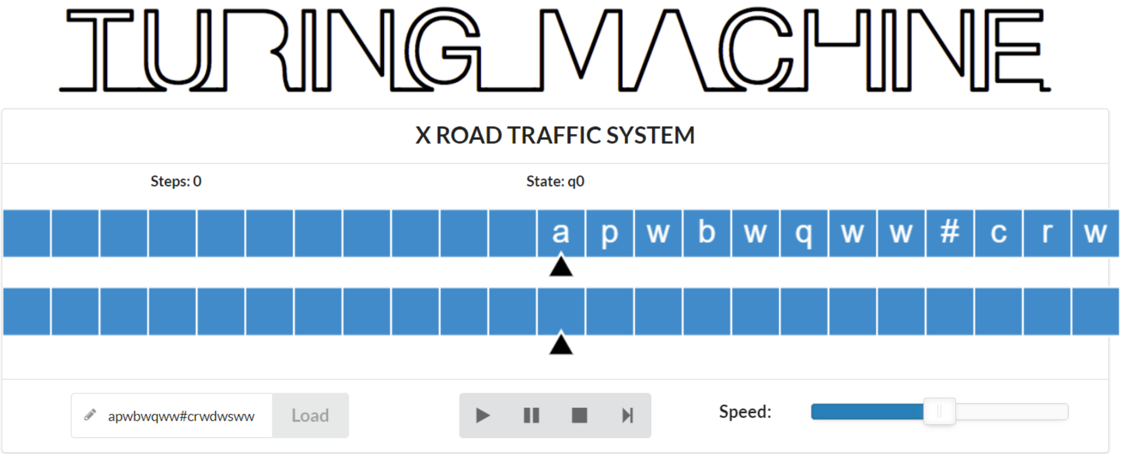
Website Link → [TURING MACHINE SIMULATOR]

```
//input = apwbwqww#crwdwsww
//input loop = apwbwqww#crwdwswwapwbwqww#crwdwsww
//input reject = apwbwbqww#crwdwsww

name: TRAFFIC
init: q0
accept: q26

q0,a,_
q1,a,_,>,-

q1,p,_
q2,p,_,>,-

q2,w,_
q2,_,_,<,-

q2,p,_
q3,_,_,>,-

q3,_,_
q3,_,_,>,-

q3,b,_
q4,_,_,>,-

q4,w,_
q5,b,_,<,-

q5,_,_
q5,_,_,<,-

q5,a,_
q6,_,_,>,-

q6,_,_
q6,_,_,>,-

q6,b,_
```

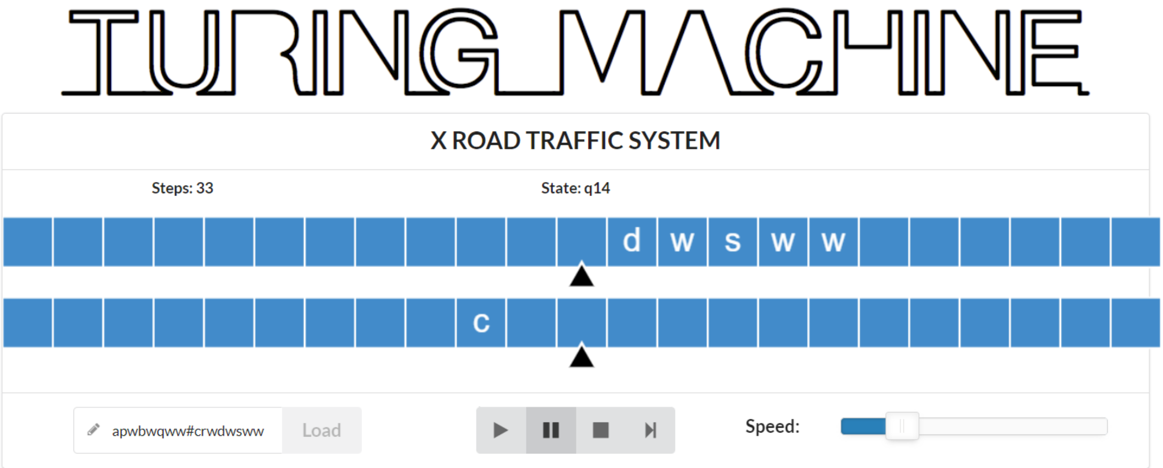
```
q7,b,_,>,-

q7,q,_
q8,q,_,>,-

q8,w,_
q8,_,_,<,-

q8,q,_
q9,_,_,>,-

q9,_,_
q9,_,_,>,-

q9,w,_
q10,_,_,<,-

q10,_,_
q10,_,_,<,-

q10,b,_
q11,_,_,<,-

q11,_,_
q11,_,_,>,-

q11,#,_
q11,_,_,>,-

q11,c,_
q12,_,c,>,>

q12,r,_
q13,_,r,>,>

q13,w,_
q13,_,_,<,<

q13,_,r
q14,_,_,>,>

q14,_,_
q14,_,_,>,>
```

```
q14,d,_
q15,_,_,>,>

q15,w,_
q16,_,d,<,<

q16,_,_
q16,_,_,<,<

q16,_,c
q17,_,_,>,>

q17,_,_
q17,_,_,>,>

q17,_,d
q18,_,d,>,>

q18,s,_
q19,_,s,>,>

q19,w,_
q20,_,_,<,<

q20,_,s
q21,_,_,>,>

q21,_,_
q21,_,_,>,>

q21,w,_
q22,_,_,<,<

q22,_,_
q22,_,_,<,<

q22,_,d
q23,_,_,>,>

q23,_,_
q24,_,_,>,>

q24,_,_
q25,_,_,>,>
```

```
q25,_,_
q26,_,_,>,>

q26,a,_
q1,a,_,>,>
```

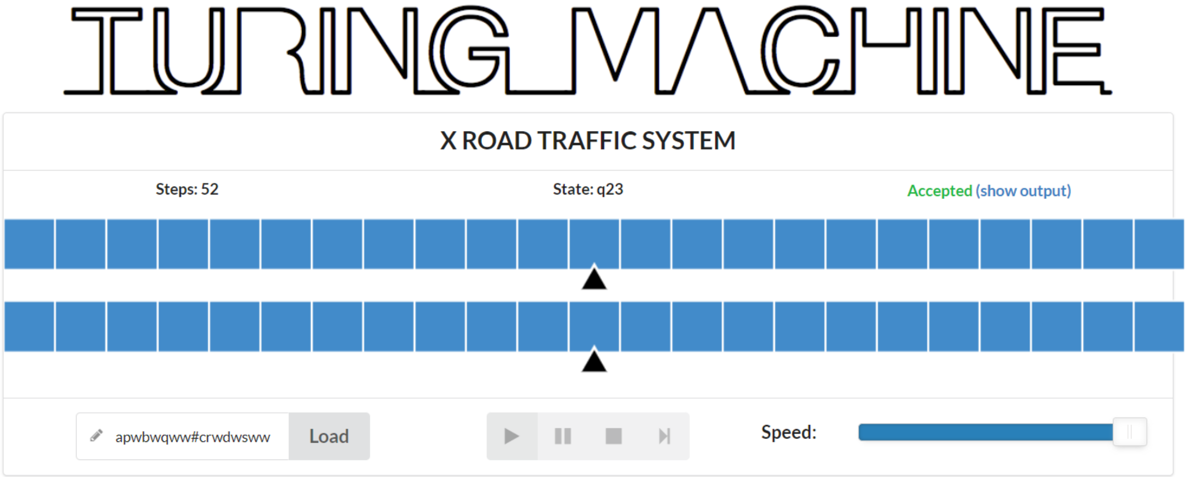


After Reaching # It moves to second tape



Case 1:

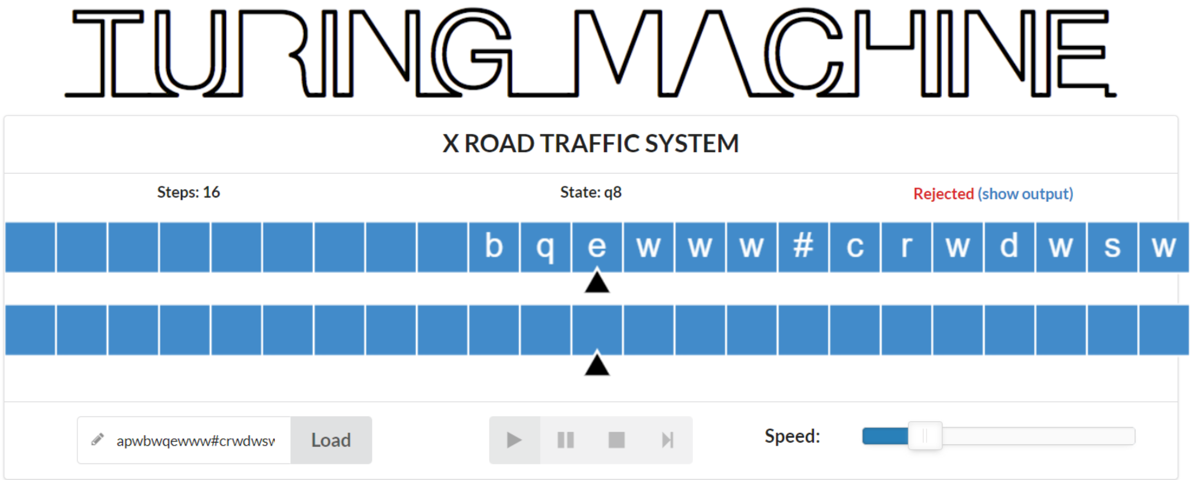
After fully executing if the input is valid it gives Accepted



Case 2:

If the provided input is not in range of the system then it will get struct in some state so leading it not to reach accepted state

Instead it reaches to Rejected State



CHALLENGES FACED WHILE DEVELOPING THE SOLUTIONS IN NPDA AND TURING MACHINE

We faced some really challenging situations while developing the solutions using NPDA and Turing Machine

Some of them are:

- Single tape to multi tape conversion: The problem could be easily solved using a single tape turing machine as the controls of each signal repeats only after 180 seconds, we managed to analyse and solve the problem in single tape much more effectively than a multi-tape, where 2 tapes were used for each pair of opposite signals
- While handling the ambulance condition, we have to flush out all the contents of the stack and reset the timer to zero. Suppose if an ambulance signal is received just before the cycle resets , the signal s4 would have to wait for another 180 seconds in the X-Road which is undesirable.
- Here we have considered the possibility of straight and right turns. For certain signals these straight and right turns might conflict, that is a vehicle turning right at one signal might hit a vehicle which is coming straight/left in the other signal. This too is not acceptable.
- Designing a Turing Machine requires the exact representation of junction components on the tape used by the machine.

Selecting the right model and addressing the challenges associated with each approach are vital in creating a practical and precise representation of a cross road junction. It is important to choose a model that suits the problem's characteristics and overcome the difficulties specific to that model. This ensures that the representation accurately captures the behaviour of the junction and provides valuable insights for analysis and optimization.

CONTRIBUTIONS

- Problem Statement Creator - Vignesh
- Project Ideators – Vignesh, Mitesh
- Project Designers for both NPDA and TM – Viswaa, Ganesh
- Stimulations
  - NPDA – Viswaa
  - Single Tape – Rajkumar
  - Multi Tape – Ganesh
- Turing Machine Coder – Vignesh
- Images Designer – Rajkumar
- Document Writer - Mitesh

RESOURCES

- https://www.jflap.org/jflaptmp/
- https://turingmachinesimulator.com/
- https://www.figma.com/

GIT HUB

https://github.com/TOC-NPDA-TURINGMACHINE-PROJECT/X-ROAD-TRAFFIC-SYSTEM