CHAPTER 4 FINITE STATE MACHINE

A machine is a device which accepts some inputs, possibly produces some output, and has some memory storing information on the overall results of all previous inputs.

The condition of the machine at a particular instance and all of its memory is called a state of the machine. New input changes the state, may be to another state, and may be to the same state. The effect of a new input depends on the present state that the machine is in.

The effects of inputs on the states are represented by state transition functions. These functions may be expressed in the form of state transition tables or state transition diagrams. A state transition diagram is the digraph of the machine.

If the total number of possible states is finite, the machine is called finite state machine, eg, a computer.

We represent the state set by S, the input set by I, and the set of transition functions by F.

For $x \in I$, the function f_x shows the effect on the state when input x is received.

eg (1)

Let $S = \{s_o, s_1\}$, $I = \{0,1\}$. The state transition functions are described by this table:

	f_o	f_1
s_o	s_o	s_1
s_1	s_1	s_o

Draw the state transition diagram for this machine.

Machine congruence and quotient machine

An equivalence relation on the state set S of a machine M is called a machine congruence if

$s \, R \, t \Rightarrow f_x(s) \, R \, f_x(t) \quad \forall x \in I.$

eg (2)

Let M = [S, I, F] be given by S = {s_o, s_1, s_2, s_3, s_4, s_5}, I = {a,b}

F	a	b
s_o	s_o	s_4
s_1	s_1	s_o
s_2	s_2	s_4
s_3	s_5	s_2
s_4	s_4	s_3
s_5	s_3	s_2

$$\text{Let } M_R = \begin{array}{c} \begin{array}{cccccc} s_o & s_1 & s_2 & s_3 & s_4 & s_5 \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \end{array}$$

Suppose R is the equivalence relation whose matrix is given above.
Show that this R is a machine congruence.

Solution:

R partitions S into $\{\{s_o, s_2\}, \{s_1, s_3, s_5\}, \{s_4\}\}$

Quotient machine

Suppose R is a machine congruence on a machine M. As an equivalence relation, R partitions S into equivalence classes. The transitions from states to states may be considered as transitions from classes to classes.

$S/R = \{\text{equivalence classes}\} = \{[s] : s \in S\}$

We may define class transition function f_x by

$f_x([s]) = [f_x(s)]$

The machine $M/R = [S/R, I, \overline{F}]$ with transition functions $\overline{f_x}$ as defined above is called the quotient of M corresponding to R.

eg (3)

Continuing the last example, tabulate the class transition functions. Draw the digraphs of M and M/R.

Digraph of M

Word transition function

Let $w = x_1\ x_2\ x_3\ \ldots x_n \in I^*$ (ie. w is an input string).

Define $f_w = fx_n \circ \ldots \circ fx_3 \circ fx_2 \circ fx_1$. ($\circ$ is the composition of functions.)

This f_w represents the effect of the input string $x_1\ x_2\ x_3\ \ldots x_n$ on the state. This is the combined effect of the sequence of inputs $x_1\ x_2\ x_3\ \ldots x_n$.

eg (4)

Refer to the machine M in eg(2).

Consider the input string w = abb.

Compute $f_w(s_o)$, $f_w(s_1)$. Tabulate f_w .

We may read $f_w(s_o) = f_{abb}(s_o)$ from the digraph of the machine:

Similarly, $f_w(s_1) = f_{abb}(s_1) =$

$f_w = f_{abb}$ is tabulated as:

Moore machine or Recognition machine

Let S be the state set, I be the input set, F be the set of state transition functions.

Let s_o be the starting state. (For example, when a computer is just switched on, before any input.)

Certain states in S are considered as acceptance states. The set of all acceptance states is represented by T. If an input string w brings s_o into a state in T, this w is accepted by the machine as a meaningful word. Such a machine [S, I, F, s_o,T] is called a Moore machine.

The set of all input strings recognised by this machine is called the **language of the machine**:

$$L(M) = \{w: w \in I^{*} \text{ and } f_w(s_o) \in T\}.$$

In drawing state transition diagram, an acceptance state is enclosed in double circle, such as S_1 .

eg (5)

Consider the Moore machine $M = [S, I, F, s_o, T]$, where $S = \{s_o, s_{1,} s_2\}$, $I = \{a,b\}$, $T = \{s_2\}$. The state transition functions are tabulated below:

F	a	b
s_o	s_o	s_1
s_1	s_o	s_2
s_2	s_2	s_2

Draw the state transition diagram. Describe the input strings that are recognised by M. Give examples of some strings in L(M) and some strings not in L(M).

Solution:

We may construct a phrase structure Grammar whose language is the same as the language of this machine:

$G = [I, V, v_o, \mapsto]$, with $I = \{a,b\}$, $V = I \cup S$, $v_o = s_o$.

The production relation $\mapsto$ in BNF notation is described by:

The above illustrate this general form of production rule:

<present state> ::= input1<resulting state>

|input2<resulting state>|……………

(|input if resulting state is an acceptance state)

The master syntax diagram for G is:

L(G) =

where x^* means x^n for $n \geq 0$.

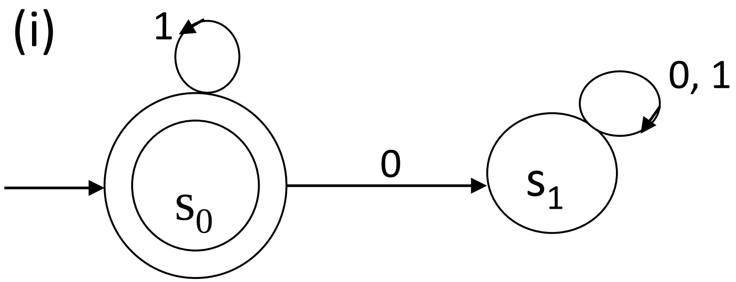
An expression obtained by combining symbols using $\vee$, *, (,) is called a <u>regular expression</u>. A language involving regular expressions is a regular language.

Every Moore machine, M, is equivalent to a certain regular grammar, G, in the sense that L(M) = L(G).

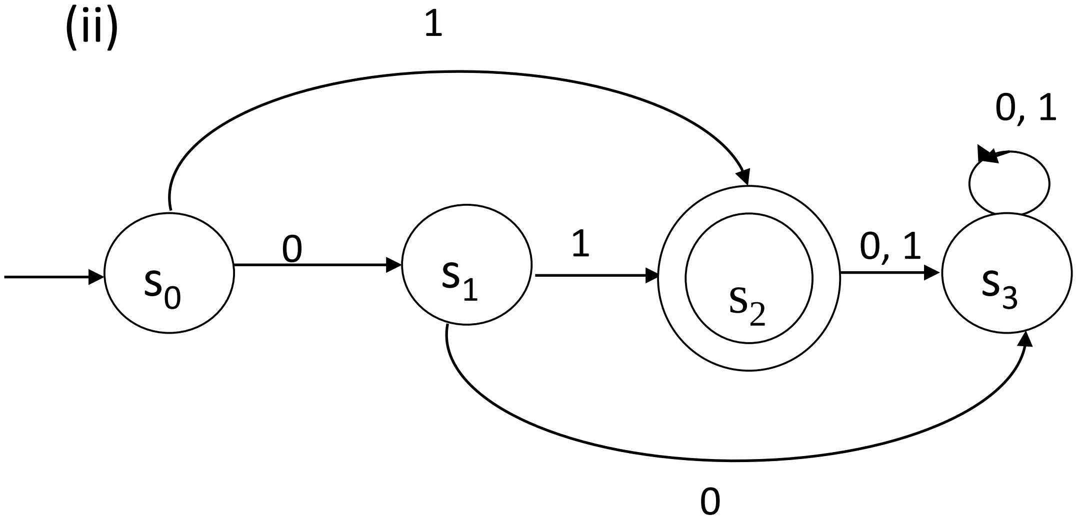
eg (5)(a)

Determine the language recognized by the finite-state machine whose state diagrams are sketched below:

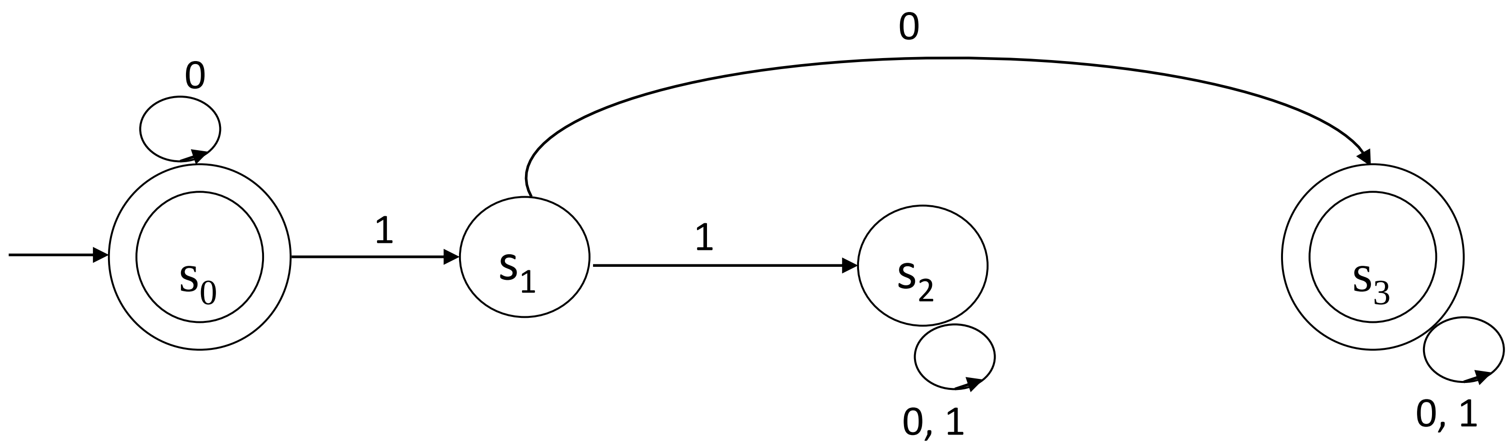
(i)



(ii)



(iii)



eg (6)

Construct the digraph of a Moore machine that has input elements 0, 1, and accepts only input strings ending in 11000.

[i.e. $L(M) = \{ (0 \vee 1)^*11000 \}$.]

Simplification of Moore Machine

Suppose a Moore machine $M = (S, I, F, s_o, T)$ is designed. There may be some redundancy, which we wish to eliminate. We attempt to obtain a simpler machine, involving fewer states, which is equivalent to the original machine, in the sense that it recognises the same input strings as the given machine.

Two states s and t are said to be w-compatible if $f_w(s)$ and $f_w(t)$ are both in T or both not in T.

We define a relation R by:

s R t if they are w-compatible for all $w \in I^*$

This R is an equivalence relation on S, and is a machine congruence. The quotient machine $\overline{M} = (S/R, I, \overline{F}, [s_o], T/R)$ is equivalent to the original machine M, where F is defined by $\bar{f}_x([s]) = [f_x(s)]$.

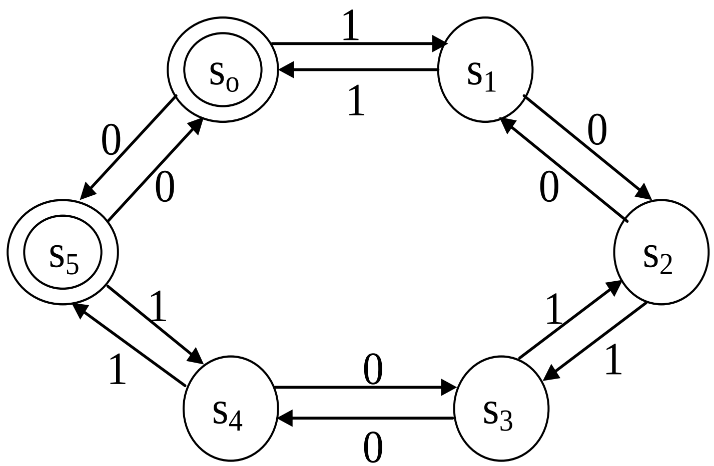
The relation R may be found by obtaining a sequence of partitions of S: P_o, P_1, P_2 ,... with $P_o = \{T, \overline{T}\}$. After obtaining P_k, we obtain P_{k+1} from P_k by further decomposing the classes in P_k such that for 2 states to remain in the same block in P_{k+1}, every input $x \in I$ should bring these 2 states into the same block in P_k.

Proceed until $P_{k+1} = P_k$.

The last partition P_k corresponds to the congruence relation R required.

eg (7)

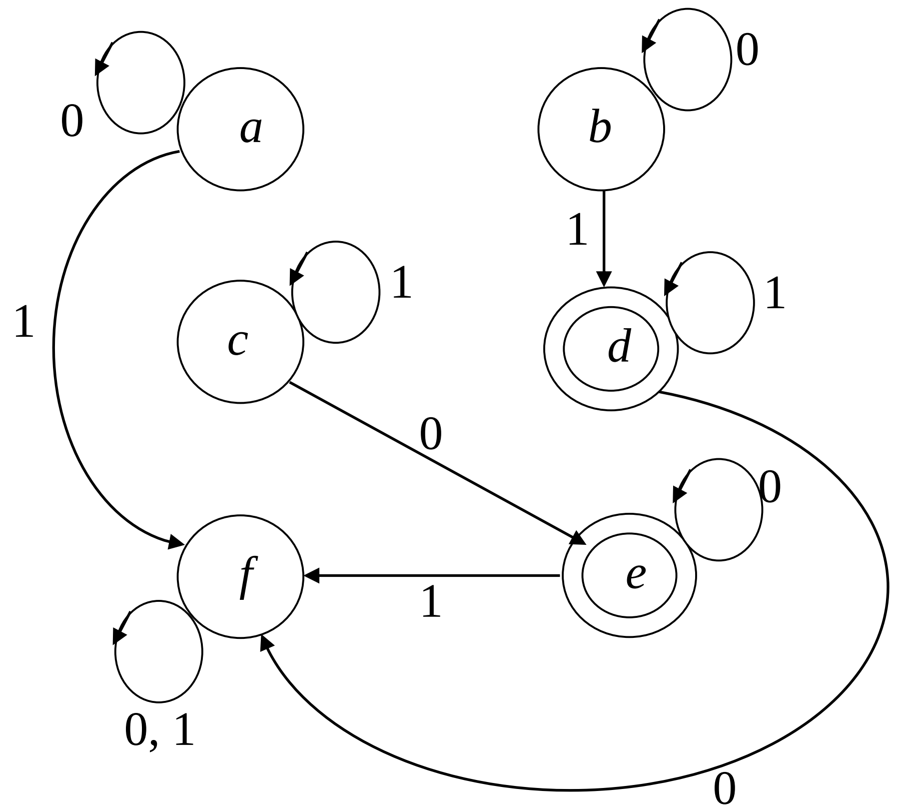
Simplify this Moore machine.



Quotient machine

eg (8)

Consider the digraph of the following finite state machine, *M*.



(a) Construct the state transition table of M.

(b) Find the partition of the state set corresponding to a machine congruence relation R, with as few classes as possible.

(c) Construct the state transition table of the quotient Moore machine, M/R and draw the digraph of M/R.

(d) Tabulate the word transition function f_{000} for the original machine M.

Solution:

Nondeterministic Finite-state Machine

The finite-state machine discussed so far are deterministic, since for each pair of state and input value there is a unique next state given by the transition function.

Another type of finite-state machine in which there may be several possible next states for each pair of input value and state. Such machines are called nondeterministic.

A nondeterministic finite-state machine M = (S, I, F, s_o, T) consists of a set S of states, an input set I, a transition function F that assigns a set of states to each pair of state and input (so that $F: S \times I \rightarrow P(S)$), a starting state s_o, and a subset T of S consisting of the acceptance states.

Nondeterministic finite-state machine can be represented by using state transition tables or state transition diagrams.

eg (9)

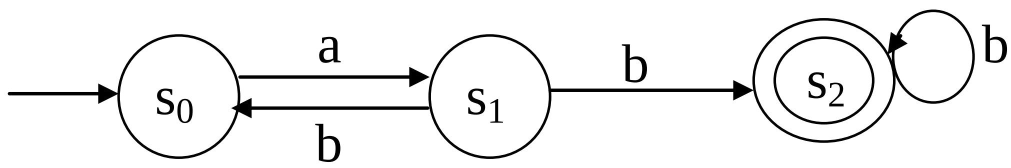
Find the state transition diagram for the nondeterministic finite state machine with the state transition table shown below. The acceptance states are s_2 and s_3.

	F	
	Input	
State	0	1
s_o	s_o, s_1	s_3
s_1	s_o	s_1, s_3
s_2		s_o, s_2
s_3	s_o, s_1, s_2	s_1

Solution

eg (10)

Find the state transition table for the nondeterministic finite-state machine with the state transition diagram shown below.



Solution

Language recognised by a non-deterministic Moore machine

An input string is recognised by a non-deterministic Moore machine if it can bring the initial state to an acceptance state. The set of all recognised strings is called the language of the machine.

eg (11):

In eg (10), write some strings which are recognised by the machine and determine whether the input string abab is in the language of the machine. Hence write the language of the machine, L(M).

Solution:

Some strings which are recognised by the machine are

Hence abab is in the language of the machine (accepted by the machine).

The language of the machine is L(M) =

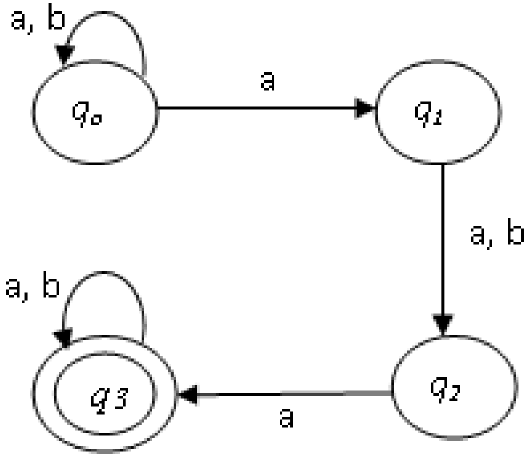
eg (12)

Find a deterministic finite-state machine that recognizes the same language as the nondeterministic finite-state machine in eg(10) above.

Solution

eg (13)

For the following nondeterministic finite state machine, *M* whose state transition diagram is shown below,

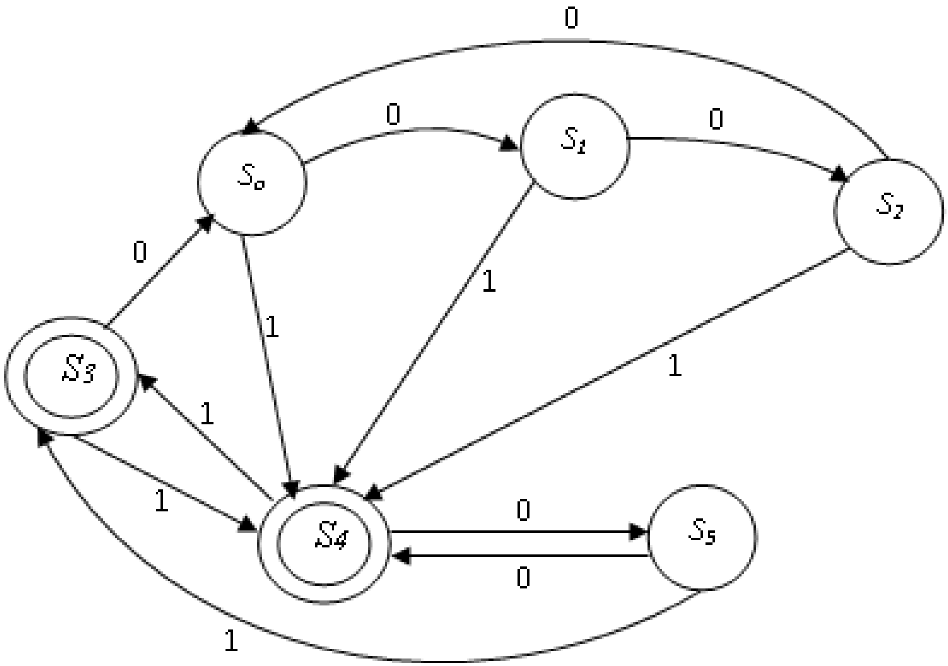


(a) construct a state transition table of *M*,

(b) construct a state transition table and state transition diagram of the corresponding deterministic finite state machine of *M*.

Extra Example 1:

Consider the finite state machine, M given by the following transition diagram:



(i) Construct the state transition table of M.

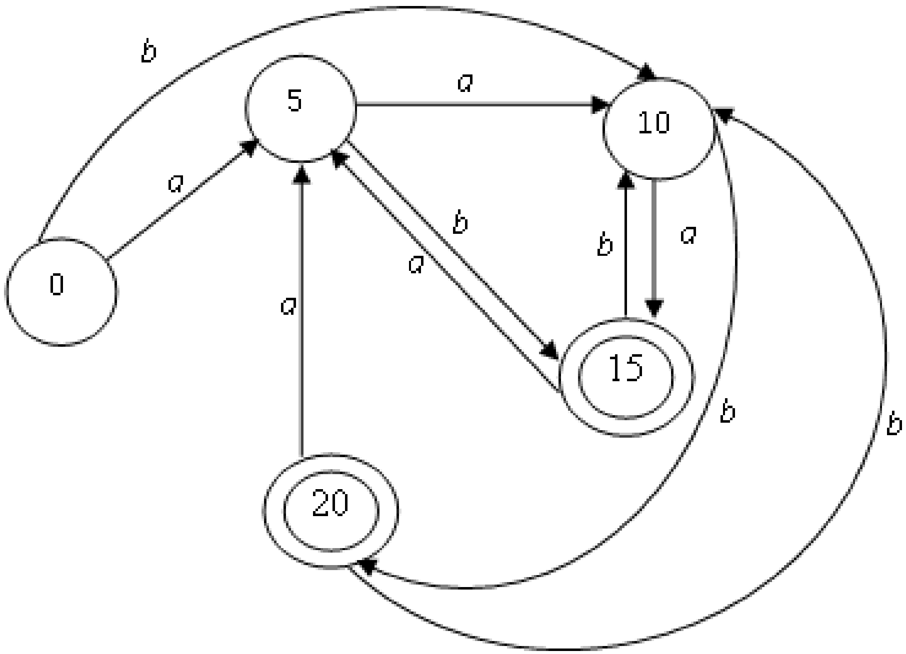
(ii) Tabulate the word transition function f_{01101} for the original machine M.

(iii) Find the partition of the state set corresponding to a machine congruence relation R, with as few classes as possible.

(iv) Draw the state transition diagram of the quotient Moore machine, M/R.

Extra Example 2:

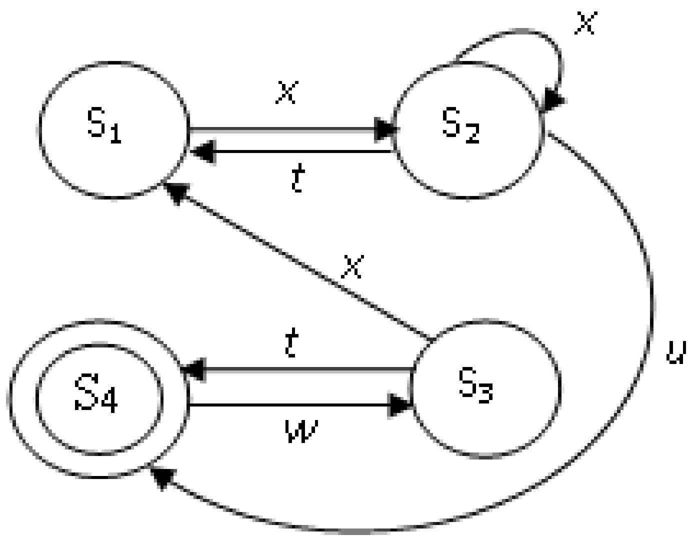
(a) Consider the finite state machine, M, where the initial state is 0, input set is $\{a, b\}$ and the state transition diagram is shown below.



(i) Construct the state transition table of M.

(ii) Determine $\overline{M}$ the minimal quotient machine of M, by constructing the state transition table for $\overline{M}$ and draw the digraph for $\overline{M}$. Show all the steps clearly.

(b) Consider the finite state machine above where the initial state is S_1 and the input set is $\{x, t, u, w\}$, determine which of the following input strings are valid. Give a reason if your answer is not valid.



(i) xtuwt

(ii) xuwx

(iii) xuwtwt

(iv) xxxuwt

(v) xtxtxuwxxxu