

8.1 Ex – Parsing using a Finite State Machine (FSM) algorithm

A finite state machine also known as a finite state acceptor is a theoretical machine or algorithm that uses a table to determine its current state and its action. The system is described with states. Then depending on the current state and on the input you go to a new state and perform some action. The operation of parsing is to convert a string into its meaningful data. For example to convert a string into a real number. In this example we are going to convert a string into a real number or double. We decompose the problem into 4 states:

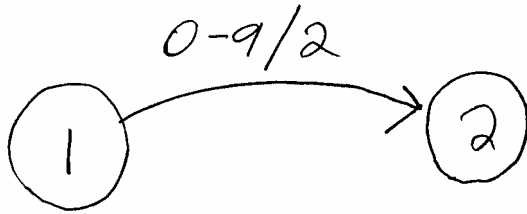
- State 0: Represents the starting and ending state. It says we have not yet started and are still expecting the first meaningful data in the number.
- State 1: In this state we have seen the sign of the value, if for example one put a “-“ or “+” sign in front of the number.
- State 2: In this state the algorithm is seeing the digits to the left of the decimal. It must multiply the sum by 10 and add the value of the new digit.
- State 3: In this state the algorithm is seeing the digits to the right of the decimal. It must multiply the denominator by 10 and add the next digit divided by the denominator to the sum.

At the end the state goes back to 0 and the function quits. Note the sign, decimal and digits after the decimal are optional.

The list of actions are:

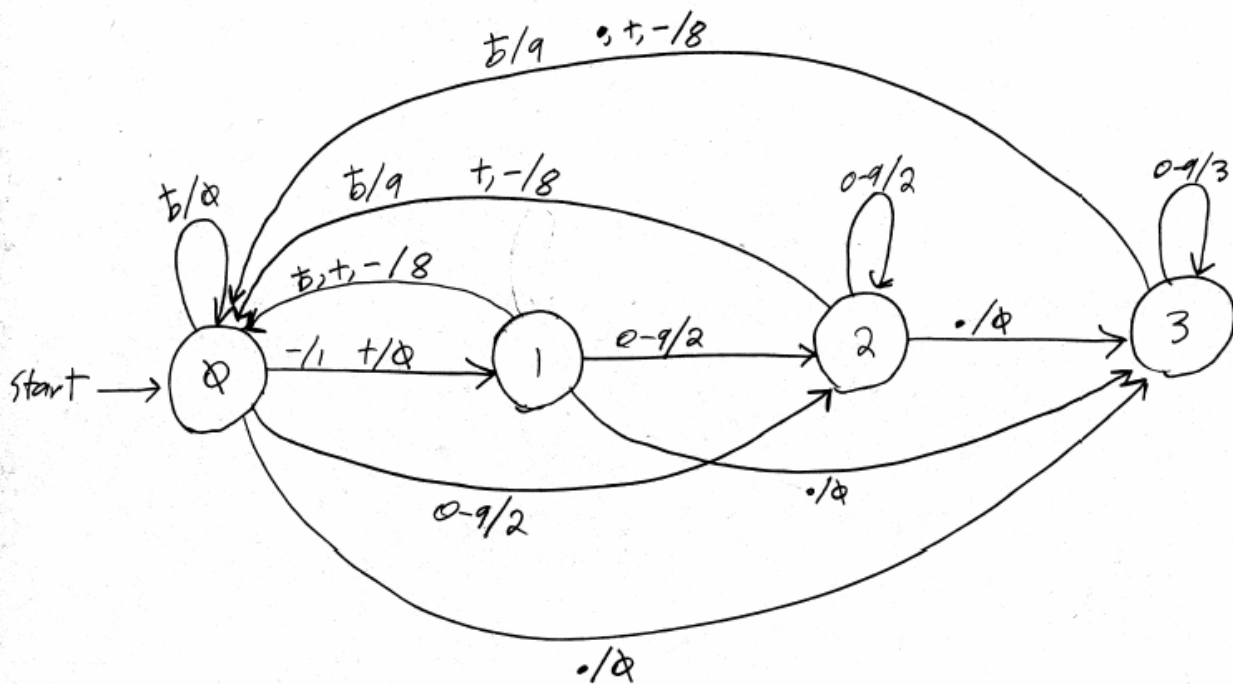
- Action 0: Do nothing
- Action 1: Change the sign to -1
- Action 2: $sum = sum * 10 + (ch - '0')$
- Action 3: $den = den * 10$
 $Sum = sum + (ch - '0') / den;$
- Action 8: $sum = 0$
Error
stop
- Action 9: $sum = sum * sign$
stop

For the graphical view of the FSM we use



To mean that if the state is 1 and the next input is a digit then go to state 2 and perform action 2.

The following is the state diagram for our FSM.



The following table represents the FSM in the diagram. We use the notation X/Y to mean “go to state X and perform action Y.”

	-	+	digit	period	white space
State 0	1/1	1/0	2/2	3/0	0/0
State 1	0/8	0/8	2/2	3/0	0/8
State 2	0/8	0/8	2/2	3/0	0/9
State 3	0/8	0/8	3/3	0/8	0/9

The following is the code:

```
#include "stdio.h"
#include "conio.h"
#include "string.h"

#define MINUS 0
#define PLUS 1
#define DIGIT 2
#define DOT 3
#define WHITESPACE 4

#define FALSE 0
#define TRUE !FALSE

#define DEBUG TRUE

int toToken(int ch)
{
    int token;

    if (ch == '-')
        token = MINUS;
    else if (ch == '+')
        token = PLUS;
    else if (ch == '.')
        token = DOT;
    else if (ch >= '0' && ch <= '9')
        token = DIGIT;
    else
        token = WHITESPACE;

    return token;
}

double
ascii_to_double(char str[80])
{
    struct entry
    {
        int state;
        int action;
    };
```

```

struct entry FSAtable[4][5] = {
    {{1,1},{1,0},{2,2},{3,0},{0,0}},
    {{0,8},{0,8},{2,2},{3,0},{0,8}},
    {{0,8},{0,8},{2,2},{3,0},{0,9}},
    {{0,8},{0,8},{3,3},{0,8},{0,9}}
};

int      state = 0,
         sign = 1;
double   sum = 0,
         dem = 1;

int      done = FALSE,
         i = 0,
         len,
         ch,
         NextState,
         Action,
         token;

len = strlen(str);

while (!done)
    {
    ch = str[i++];
    token = toToken(ch);

    NextState = FSAtable[state][token].state;
    Action = FSAtable[state][token].action;

    if (DEBUG)
        printf("got %c so went from state %d to %d with action %d\n",
              ch,state,NextState,Action);

    state = NextState;

    switch (Action)
        {
        case 0:
            break;

        case 1:
            sign = -1;
            break;

        case 2:
            sum = (sum * 10) + (ch - '0');
            break;
        }
    }

```

```

        case 3:
            dem = dem * 10;
            sum = sum + (ch - '0') / dem;
            break;

        case 8:
            sum = 0;
            printf("Error in ascii_to_double\n");
            printf("Can not convert %s to a double\n",str);
            done = TRUE;
            break;

        case 9:
            sum = sum * sign;
            done = TRUE;
            break;
    }

    if (i > len)
        done = TRUE;
    }

    return sum;
}

void main()
{
    char str[80];

    printf("Enter a string to convert: ");
    gets(str); // includes all of the string including spaces until the new line.
    printf("The value of \"%s\" is %lf\n", str, ascii_to_double(str) );
}

```