

24-Mar-22

Experiment 9 - Computation of LR(0) Item

Dhawal Patil

RA1911003010575

CSE A2

Aim:

To write a program to show Computation of LR(0) Item using C++ language.

Algorithm:

1. Start.
2. Create structure for production with LHS and RHS.
3. Open file and read input from file.
4. Build state 0 from extra grammar Law $S' \rightarrow S \$$ that is all start symbol of grammar and one
5. Dot (.) before S symbol.
6. If Dot symbol is before a non-terminal, add grammar laws that this non-terminal is in Left
7. Hand Side of that Law and set Dot in before of first part of Right Hand Side.
8. If state exists (a state with this Laws and same Dot position), use that instead.
9. Now find set of terminals and non-terminals in which Dot exist in before.
10. If step 7 Set is non-empty go to 9, else go to 10.
11. For each terminal/non-terminal in set step 7 create new state by using all grammar law that Dot position is before of that terminal/non-terminal in reference state by increasing Dot point to next part in Right Hand Side of that laws.
12. Go to step 5.
13. End of state building.
14. Display the output.
15. End.

Code:

```
#include<iostream>
#include<conio.h>
#include<string.h>
using namespace std;

char prod[20][20],listofvar[26]="ABCDEFGHJKLMNOPQR";
int novar=1,i=0,j=0,k=0,n=0,m=0,arr[30];
int noitem=0;

struct Grammar
{
    char lhs;
    char rhs[8];
}g[20],item[20],clos[20][10];

int isvariable(char variable)
{
    for(int i=0;i<novar;i++)
        if(g[i].lhs==variable)
            return i+1;
```

```

        return 0;
    }
    void findclosure(int z, char a)
    {
        int n=0,i=0,j=0,k=0,l=0;
        for(i=0;i<arr[z];i++)
        {
            for(j=0;j<strlen(clos[z][i].rhs);j++)
            {
                if(clos[z][i].rhs[j]=='.' && clos[z][i].rhs[j+1]==a)
                {
                    clos[noitem][n].lhs=clos[z][i].lhs;
                    strcpy(clos[noitem][n].rhs,clos[z][i].rhs);
                    char temp=clos[noitem][n].rhs[j];
                    clos[noitem][n].rhs[j]=clos[noitem][n].rhs[j+1];
                    clos[noitem][n].rhs[j+1]=temp;
                    n=n+1;
                }
            }
        }
        for(i=0;i<n;i++)
        {
            for(j=0;j<strlen(clos[noitem][i].rhs);j++)
            {
                if(clos[noitem][i].rhs[j]=='.' && isvariable(clos[noitem][i].rhs[j+1])>0)
                {
                    for(k=0;k<novar;k++)
                    {
                        if(clos[noitem][i].rhs[j+1]==clos[0][k].lhs)
                        {
                            for(l=0;l<n;l++)
                            if(clos[noitem][l].lhs==clos[0][k].lhs &&
                                strcmp(clos[noitem][l].rhs,clos[0][k].rhs)==0)
                                break;
                            if(l==n)
                            {
                                clos[noitem][n].lhs=clos[0][k].lhs;
                                strcpy(clos[noitem][n].rhs,clos[0][k].rhs);
                                n=n+1;
                            }
                        }
                    }
                }
            }
        }
        arr[noitem]=n;
        int flag=0;
        for(i=0;i<noitem;i++)
        {
            if(arr[i]==n)
            {
                for(j=0;j<arr[i];j++)
                {
                    int c=0;
                    for(k=0;k<arr[i];k++)
                    if(clos[noitem][k].lhs==clos[i][k].lhs &&
                        strcmp(clos[noitem][k].rhs,clos[i][k].rhs)==0)

```

```

                                c=c+1;
                                if(c==arr[i])
                                {
                                    flag=1;
                                    goto exit;
                                }
                            }
                        }
                    }
                }
            }
        }
        exit;;
        if(flag==0)
            arr[noitem++]=n;
    }

int main()
{
    cout<<"ENTER THE PRODUCTIONS OF THE GRAMMAR(0 TO END) :\n";
    do
    {
        cin>>prod[i++];
    }while(strcmp(prod[i-1],"0")!=0);
    for(n=0;n<i-1;n++)
    {
        m=0;
        j=novar;
        g[novar++].lhs=prod[n][0];
        for(k=3;k<strlen(prod[n]);k++)
        {
            if(prod[n][k] != '|' )
                g[j].rhs[m++]=prod[n][k];
            if(prod[n][k]=='|' )
            {
                g[j].rhs[m]='\0';
                m=0;
                j=novar;
                g[novar++].lhs=prod[n][0];
            }
        }
    }
    for(i=0;i<26;i++)
        if(!isvariable(listofvar[i]))
            break;
    g[0].lhs=listofvar[i];
    char temp[2]={g[1].lhs,'\0'};
    strcat(g[0].rhs,temp);
    cout<<"\n\n augmented grammar \n";
    for(i=0;i<novar;i++)
        cout<<endl<<g[i].lhs<<"->"<<g[i].rhs<<" ";

    for(i=0;i<novar;i++)
    {
        clos[noitem][i].lhs=g[i].lhs;
        strcpy(clos[noitem][i].rhs,g[i].rhs);
        if(strcmp(clos[noitem][i].rhs,"ε")==0)
            strcpy(clos[noitem][i].rhs,".");
        else
        {

```

```

        for(int j=strlen(clos[noitem][i].rhs)+1;j>=0;j--)
            clos[noitem][i].rhs[j]=clos[noitem][i].rhs[j-1];
        clos[noitem][i].rhs[0]='.';
    }
}
arr[noitem++]=novar;
for(int z=0;z<noitem;z++)
{
    char list[10];
    int l=0;
    for(j=0;j<arr[z];j++)
    {
        for(k=0;k<strlen(clos[z][j].rhs)-1;k++)
        {
            if(clos[z][j].rhs[k]=='.')
            {
                for(m=0;m<l;m++)
                    if(list[m]==clos[z][j].rhs[k+1])
                        break;
                if(m==l)
                    list[l++]=clos[z][j].rhs[k+1];
            }
        }
    }
    for(int x=0;x<l;x++)
        findclosure(z,list[x]);
}
cout<<"\n THE SET OF ITEMS ARE \n\n";
for(int z=0; z<noitem; z++)
{
    cout<<"\n I"<<z<<"\n\n";
    for(j=0;j<arr[z];j++)
        cout<<clos[z][j].lhs<<"->"<<clos[z][j].rhs<<"\n";
}
}

```

```
main.cpp
1 #include<iostream>
2 #include<conio.h>
3 #include<string.h>
4
5 using namespace std;
6
7 char prod[20][20],listofvar[26]="ABCDEFGHIJKLMNOPQRSTUVWXYZ";
8 int novar=1,i=0,j=0,k=0,n=0,m=0,arr[30];
9 int noitem=0;
10
11 struct Grammar
12 {
13     char lhs;
14     char rhs[8];
15 }g[20],item[20],clos[20][10];
16
17 int isvariable(char variable)
18 {
19     for(int i=0;i<novar;i++)
20         if(g[i].lhs==variable)
21             return i+1;
22     return 0;
23 }
24 void findclosure(int z, char a)
25 {
26     int n=0,i=0,j=0,k=0,l=0;
27     for(i=0;i<arr[z];i++)
28     {
29         for(j=0;j<strlen(clos[z][i].rhs);j++)
30         {
31             if(clos[z][i].rhs[j]=='.' && clos[z][i].rhs[j+1]==a)
32             {
33                 clos[noitem][n].lhs=clos[z][i].lhs;
34                 strcpy(clos[noitem][n].rhs,clos[z][i].rhs);
35                 char temp=clos[noitem][n].rhs[j];
36                 clos[noitem][n].rhs[j]=clos[noitem][n].rhs[j+1];
37                 clos[noitem][n].rhs[j+1]=temp;
38                 n=n+1;
39             }
40         }
41     }
42     for(i=0;i<n;i++)
43     {
44         for(j=0;j<strlen(clos[noitem][i].rhs);j++)
45         {
46             if(clos[noitem][i].rhs[j]=='.' && isvariable(clos[noitem][i].rhs[j+1])>0)
47             {
48                 for(k=0;k<novar;k++)
49                 {
50                     if(clos[noitem][i].rhs[j+1]==clos[0][k].lhs)
51                     {
52                         for(l=0;l<n;l++)
53                             if(clos[noitem][l].lhs==clos[0][k].lhs && strcmp(clos[noitem][l].rhs,clos[0][k].rhs)==0)
54                                 break;
55                         if(l==n)
56                         {
57                             clos[noitem][n].lhs=clos[0][k].lhs;
58                             strcpy(clos[noitem][n].rhs,clos[0][k].rhs);
59                             n=n+1;
60                         }
61                     }
62                 }
63             }
64         }
65     }
66     arr[noitem]=n;
67     int flag=0;
68     for(i=0;i<noitem;i++)
69     {
70         if(arr[i]==n)
71         {
72             for(j=0;j<arr[i];j++)
73             {
74                 int c=0;
```

```

75         for(k=0;k<arr[i];k++)
76             if(clos[noitem][k].lhs==clos[i][k].lhs && strcmp(clos[noitem][k].rhs,clos[i][k].rhs)==0)
77                 c=c+1;
78             if(c==arr[i])
79             {
80                 flag=1;
81                 goto exit;
82             }
83         }
84     }
85 }
86 exit;;
87 if(flag==0)
88     arr[noitem++]=n;
89 }
90
91 int main()
92 {
93     cout<<"ENTER THE PRODUCTIONS OF THE GRAMMAR(0 TO END) :\n";
94     do
95     {
96         cin>>prod[i++];
97     }while(strcmp(prod[i-1],"0")!=0);
98     for(n=0;n<i-1;n++)
99     {
100         m=0;
101         j=novar;
102         g[novar++].lhs=prod[n][0];
103         for(k=3;k<strlen(prod[n]);k++)
104         {
105             if(prod[n][k] != '|' )
106                 g[j].rhs[m++]=prod[n][k];
107             if(prod[n][k]=='|')
108             {
109                 g[j].rhs[m]='\0';
110                 m=0;
111                 j=novar;
112                 g[novar++].lhs=prod[n][0];
113             }
114         }
115     }
116     for(i=0;i<26;i++)
117         if(!isvariable(listofvar[i]))
118             break;
119     g[0].lhs=listofvar[i];
120     char temp[2]={g[1].lhs,'\0'};
121     strcat(g[0].rhs,temp);
122     cout<<"\n\n augmented grammar \n";
123     for(i=0;i<novar;i++)
124         cout<<endl<<g[i].lhs<<"->"<<g[i].rhs<<" ";
125
126     for(i=0;i<novar;i++)
127     {
128         clos[noitem][i].lhs=g[i].lhs;
129         strcpy(clos[noitem][i].rhs,g[i].rhs);
130         if(strcmp(clos[noitem][i].rhs,"ε")==0)
131             strcpy(clos[noitem][i].rhs,".");
132         else
133         {
134             for(int j=strlen(clos[noitem][i].rhs)+1;j>0;j--)
135                 clos[noitem][i].rhs[j]=clos[noitem][i].rhs[j-1];
136             clos[noitem][i].rhs[0]='.';
137         }
138     }
139     arr[noitem++]=novar;
140     for(int z=0;z<noitem;z++)
141     {
142         char list[10];
143         int l=0;
144         for(j=0;j<arr[z];j++)
145         {
146             for(k=0;k<strlen(clos[z][j].rhs)-1;k++)
147             {
148                 if(clos[z][j].rhs[k]!='.')
149                 {
150                     for(m=0;m<l;m++)
151                         if(list[m]==clos[z][j].rhs[k+1])
152                             break;
153                     if(m==l)
154                         list[l++]=clos[z][j].rhs[k+1];
155                 }
156             }
157         }
158         for(int x=0;x<l;x++)
159             findclosure(z,list[x]);
160     }
161     cout<<"\n THE SET OF ITEMS ARE \n\n";
162     for(int z=0; z<noitem; z++)
163     {
164         cout<<"\n I"<<z<<"\n\n";
165         for(j=0;j<arr[z];j++)
166             cout<<clos[z][j].lhs<<"->"<<clos[z][j].rhs<<"\n";
167     }
168 }
169
170 }

```

Output:

```
input
ENTER THE PRODUCTIONS OF THE GRAMMAR(0 TO END) :
E->E+T
E->T
T->T*F
T->F
F->(E)
F->i
0

augmented grammar
A->E
E->E+T
E->T
T->T*F
T->F
F->(E)
F->i
THE SET OF ITEMS ARE

I0
A->.E
E->.E+T
E->.T
T->.T*F
T->.F
F->.(E)
F->.i

I1
A->E.
E->E.+T

I2
E->T.
T->T.*F

I3
T->F.

I4
F->(.E)
E->.E+T
E->.T
T->.T*F
T->.F
F->.(E)
F->.i

I5
F->i.

I6
E->E+.T
T->.T*F
T->.F
F->.(E)
F->.i

I7
T->T*.F
F->.(E)
F->.i

I8
F->(E.)
E->E.+T

I9
E->E+T.
T->T.*F

I10
T->T*F.

I11
F->(E).

...Program finished with exit code 0
Press ENTER to exit console.
```

Result:

A program for Computation of LR(0) Item was run successfully.