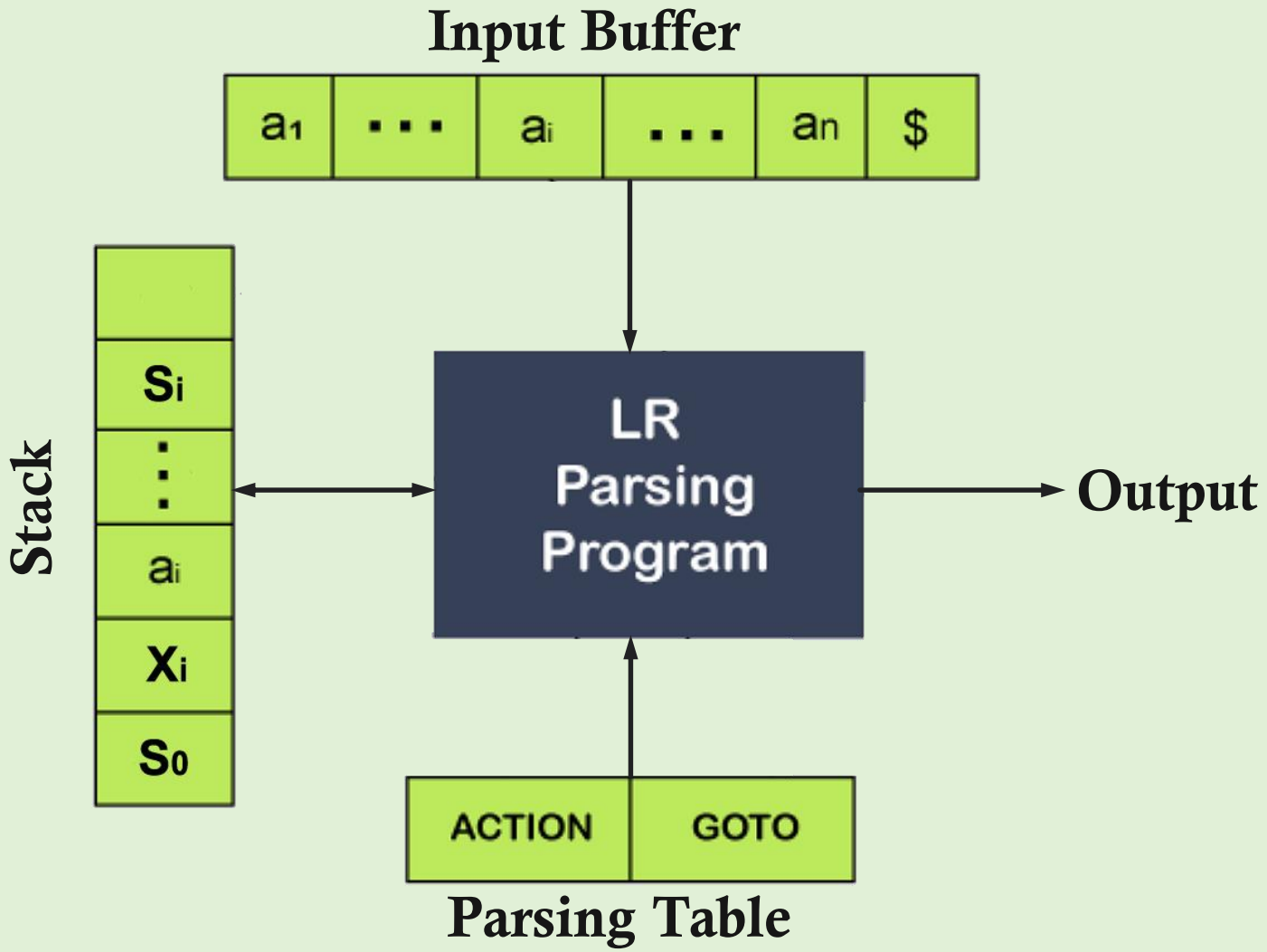
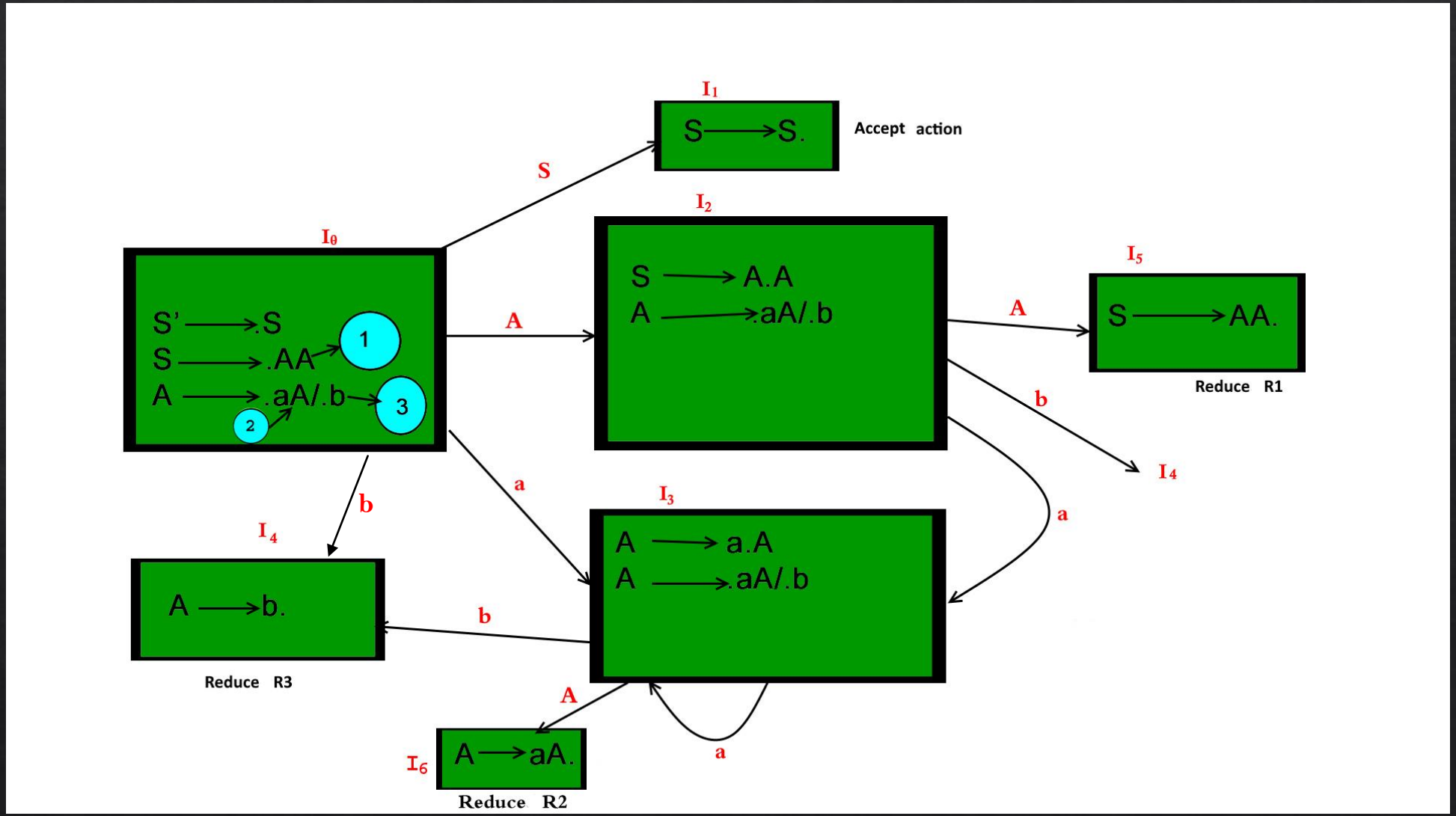
LR(0) Parser

Prepared By:
Assoc. Lec. Ahmed A. Alsamman
University of Mosul/ College of Computer Science and Mathematics/ Computer Science Dept.

LR(0) Parser Block Diagram



Step 1: Generate I0 Item
Step 2: Generate LR(0) Items Collection



Step 3: Construct LR(0) parsing table

States	Action			Go to	
	a	b	$	A	S
I_0	S3	S4	err	2	1
I_1	err	err	accept	err	err
I_2	S3	S4	err	5	err
I_3	S3	S4	err	6	err
I_4	r3	r3	r3	err	err
I_5	r1	r1	r1	err	err
I_6	r2	r2	r2	err	err

```
1  //########################### LR0 BOTTOM-UP PARSER ############################
2  //######################## CODED BY: AHMED ALI ALSAMMAN ########################
3  // Input: string of terminals
4  // Output: parsing result (Accepted / Not Accepted)
5  using System;
6  using System.Collections;
7  using System.Collections.Generic;
8  using System.Linq;
9  using System.Text;
10 using System.Threading.Tasks;
11
12
13 namespace LR0_Parser
14 {
15     class Program
16     {
17         public struct Production
18         {
19             public char lhs;
20             public string rhs;
21         }
22
23         static Stack<char> stack = new Stack<char>();
24         static string[,] parsTab = new string[,]{
25
26             //         Action                        Goto
27             //-----------------------------------------------------
28             //  a        b         $              A        S
29             //-----------------------------------------------------
30               {"s3",   "s4",     "err",          "2",     "1"  },   //I0
31               {"err",  "err",    "accept",       "err",   "err"},   //I1
32               {"s3",   "s4",     "err",          "5",     "err"},   //I2
33               {"s3",   "s4",     "err",          "6",     "err"},   //I3
34               {"r3",   "r3",     "r3",           "err",   "err"},   //I4
35               {"r1",   "r1",     "r1",           "err",   "err"},   //I5
36               {"r2",   "r2",     "r2",           "err",   "err"},   //I6
37
38          };
39
40         static char[] Column = { 'a', 'b', '$', 'A', 'S' };
41
42         static int ip = 0;
43         static void Main(string[] args)
44         {
45
46             /*
47                 Grammer G:
48
49                 S -> AA
50                 A -> aA
51                 A -> b
52             */
```

```
53
54              Production[] G = new Production[4];
55              G[1].lhs = 'S'; G[1].rhs = "AA";
56              G[2].lhs = 'A'; G[2].rhs = "aA";
57              G[3].lhs = 'A'; G[3].rhs = "b";
58
59
60              while (true)
61              {
62                  string input = Console.ReadLine();
63                  if (input.Contains('A') || input.Contains('S'))
64                  {
65                      Console.WriteLine("Error");
66                      System.Environment.Exit(0);
67                  }
68                  input = input + '$';
69                  string ParsTabCell = null;
70                  stack.Push('0'); //state 0 is the top of the statck.
71
72                  do
73                  {
74                      ParsTabCell = getCell(stack.Peek() - 48, input[ip]);
75                      if (ParsTabCell == "accept" || ParsTabCell == "err") break;
76                      if (ParsTabCell[0] == 's') //Shift operation.
77                          shift(ParsTabCell, input[ip]);
78                      else //Reduce operation
79                          reduce(ParsTabCell, G);
80                  } while (true);
81
82                  if (ParsTabCell == "accept") Console.WriteLine("Accepted");
83                  else Console.WriteLine("Not Accepted");
84                  stack.Clear(); //reset stack.
85                  ip = 0; //reset ip.
86              }
87          }
88          public static string getCell(int state, char c)
89          {
90              int col = Array.IndexOf(Column, c);
91              if (col == -1) return "err";
92
93              else
94                  return parsTab[state, col];
95          }
96
97          public static void shift(string parsTabCell, char c)
98          {
99              stack.Push(c);
100             stack.Push(parsTabCell[1]); //push only the numeric part.
101             ip++;
102         }
103
104
```

```
105
106         public static void reduce(string ParsTabCell, Production[] g)
107         {
108             int gIndex = ParsTabCell[1] - 48;
109             for (int j = 0; j < g[gIndex].rhs.Length * 2; j++) //Step 1. pop
                  twice the length of right hand side (RHS).
110                 stack.Pop();
111             int state = stack.Peek() - 48;
112             stack.Push(g[gIndex].lhs); //Step 2. push left hand side (LHS).
113             char NT = stack.Peek();
114             stack.Push(getCell(state, NT)[0]); //Step 3. get a new state and push
                   it into the stack. [0] to push one character.
115         }
116
117     }
118 }
```


Thank You!


Any
Questions