



INSTITUTO POLITÉCNICO NACIONAL

Escuela Superior de Cómputo

Práctica #6

TABLERO

Axel Treviño Palacios

2CM5

21 de enero de 2020

1. Objetivo

Elaborar un programa implementando con un autómata de pila para reconocer el lenguaje libre de contexto $\{0^n 1^n \mid n \geq 1\}$.

2. Códigos

Hay dos archivos, el archivo del servidor y el archivo de cliente.

```

1  const xlsxFFile = require("read-excel-file/node");
2  const bodyParser = require("body-parser");
3  const readline = require("readline");
4  const express = require("express");
5  const aNode = require("./aNode.js");
6  const path = require("path");
7  const fs = require("fs");
8  const app = express();
9
10 const http = require("http").createServer(app);
11 const webPort = 8080;
12
13 const excelPath = path.join(__dirname, "archivos", "ajedrez.xlsx");
14
15 var transitionTree;
16 var stack = [];
17
18 // EJS INIT
19 // Set the view engine to ejs
20 app.set("view engine", "ejs");
21 // Set ejs files path
22 app.set("views", __dirname + "/dist/pages");
23 // Set body parser
24 app.use(bodyParser.urlencoded({ extended: true }));
25
26 // REQUESTS
27 app.get("/", (req, res) => {
28   res.render("index");
29 });
30
31 app.post("/procesar/cadena/", (req, res) => {
32   let response = {};
33   let output;
34
35   if (req.body.input.length == 0) {
36     req.body.input = generateBin(10);
37   }
38
39   output = automata(req.body.input);
40
41   console.log(output);
42
43   response = output.response;
44   response.binario = req.body.input;
45
46   fs.writeFile("./archivos/proceso.txt", output.file, () => {});
47
48   // Send response
49   res.send(JSON.stringify(response));
50 });
51
52 function generateBin(len) {
53   let resultado = "";
54   for (let i = 0; i < len; ++i) {

```

```

55     resultado += Math.floor(Math.random() * 2);
56   }
57   return resultado;
58 }
59
60 // AUTOMATA
61 // Main automata functionality
62 function automata(cadena) {
63   let limi = cadena.length;
64   let current, next;
65   let response = {
66     animations: [],
67   };
68   let result = {
69     file: "",
70   };
71
72   stack = [];
73   stack.push("F");
74
75   current = [startNode];
76   response.animations.push({
77     nodeName: current[0].name,
78     input: cadena,
79     stack: stack.join(""),
80   });
81
82   result.file += `d(${current[0].name}, ${cadena}, ${stack.join("")})`
83   `;
84   for (let i = 0; i < limi; ++i) {
85     next = [];
86     current.forEach((nodo) => {
87       next = next.concat(
88         nodo.evaluateChar(cadena[i], stack[stack.length - 1])
89       );
90     });
91
92     // Removing duplicates
93     current = [...new Set(next)];
94
95     // Set animation output
96     response.animations.push({
97       nodeName: current[0].name,
98       input:
99         cadena.substring(i + 1).length == 0
100         ? "e"
101         : cadena.substring(i + 1),
102       stack: stack.join(""),
103     });
104
105     //Set file output
106     result.file += `->\nd(${current[0].name}, ${cadena.substring(
107       i
108     )}, ${stack.join("")})`;

```

```

108 }
109
110 // Process remaining stack things
111 limi = stack.length;
112 for (let i = 0; i < limi; ++i) {
113     next = [];
114     current.forEach((nodo) => {
115         next = next.concat(nodo.evaluateChar(" ", stack[i]));
116     });
117
118     // Removing duplicates
119     current = [...new Set(next)];
120
121     // Set animation output
122     response.animations.push({
123         nodeName: current[0].name,
124         input: "e",
125         stack: stack.join("").substring(i + 1),
126     });
127
128     //Set file output
129     result.file += `->\nd(${current[0].name}, e, ${stack.join("")})`;
130 }
131
132 if (current[0].name == "qf") {
133     response.result = "La cadena es v lida!"
134 } else {
135     response.result = "La cadena no es v lida :c"
136 }
137
138 // Add animation list
139 result.response = response;
140
141 return result;
142 }
143
144 // Create automata
145 function initAutomata() {
146     startNode = new aNode("q0");
147     let q1 = new aNode("q1");
148     let q2 = new aNode("qf");
149
150     q2.callback = (char) => {
151         // Input was valid
152         console.log("Input was valid");
153     };
154
155     q1.nextNodes.push({
156         input: " ",
157         stack: "F",
158         node: q2,
159         callback: () => {},
160     });
161     q1.nextNodes.push({

```

```

162     input: "1",
163     stack: " ",
164     node: q1,
165     callback: () => {
166         // Pop from stack
167         stack.pop();
168     },
169 });
170
171 startNode.nextNodes.push({
172     input: "0",
173     stack: " ",
174     node: startNode,
175     callback: () => {
176         // Push 'a' to stack
177         stack.push("a");
178     },
179 });
180 startNode.nextNodes.push({
181     input: "1",
182     stack: " ",
183     node: q1,
184     callback: () => {
185         // Pop from stack
186         stack.pop();
187     },
188 });
189 }
190
191 // SERVER SET-UP
192 app.use(express.static(__dirname + "/dist/public/"));
193
194 // SERVER LISTEN INIT
195 http.listen(webPort, () => {
196     console.log("Listening on port: " + webPort);
197 });
198
199 // Init
200 initAutomata();

```

```

1
2 @media only screen and (min-width: 768px) and (max-width: 991px) {
3
4     #main {
5         width: 712px;
6         padding: 100px 28px 120px;
7     }
8
9     /* .mono {
10         font-size: 90%;
11     } */
12
13     .cssbtn a {
14         margin-top: 10px;

```

```

15     margin-bottom: 10px;
16     width: 60px;
17     height: 60px;
18     font-size: 28px;
19     line-height: 62px;
20 }

```

```

1 class TelegramRequestHandler(object):
2     def handle(self):
3         addr = self.client_address[0]           # Client IP-address
4         telgram = self.request.recv(1024)       # Recieve telgram
5         print "From: %s, Received: %s" % (addr, telgram)
6         return

```

```

1
2 @media only screen and (min-width: 768px) and (max-width: 991px) {
3
4     #main {
5         width: 712px;
6         padding: 100px 28px 120px;
7     }
8
9     /* .mono {
10         font-size: 90%;
11     } */
12
13     .cssbtn a {
14         margin-top: 10px;
15         margin-bottom: 10px;
16         width: 60px;
17         height: 60px;
18         font-size: 28px;
19         line-height: 62px;
20     }

```

```

1  <!DOCTYPE html>
2  <html>
3
4  <head>
5    <%- include('../partials/head'); %>
6
7    <script>
8      var animationTime = 1000;
9      var animationQueue = [];
10     var currentAnimation;
11     var result;
12
13     function movePiece(piece, elementBoard, time) {
14       return piece.animate({
15         left: elementBoard.offset().left + ((elementBoard.outerWidth
16           () - piece.outerWidth()) / 2),
17         top: elementBoard.offset().top + ((elementBoard.outerHeight
18           () - piece.outerHeight()) / 2)
19       }, time).promise();
20     }
21
22     async function transportPiece(idOrigin, idDestination, time =
23       animationTime) {
24       let piece = $("#divFicha").clone().removeAttr("id").addClass("
25         step").removeClass("d-none").appendTo("body");
26
27       let elementOrigin = findBoardElement("Q" + idOrigin);
28       let elementDestination = findBoardElement("Q" + idDestination)
29         ;
30
31       await movePiece(piece, elementOrigin, 0);
32       await movePiece(piece, elementDestination, time);
33     }
34
35     function findBoardElement(idBoard) {
36       return $("td").filter(function () {
37         return $(this).text() == idBoard;
38       });
39     }
40
41     $(document).ready(function () {
42       $("#btnStart").click(function () {
43         $.post("/procesar/cadena/", {
44           input: $("#txtBinario").val()
45         }, function (data, status) {
46           data = JSON.parse(data);
47
48           // Setting input text
49           $("#txtBinario").val(data.binario);
50
51           // Setting future result text
52           result = data.result;
53
54           console.log(data);
55         });
56       });
57     });
58   </script>
59 </head>
60 <body>
61   <div id="board">
62     <table border="1">
63       <tr>
64         <td>Q1</td>
65         <td>Q2</td>
66         <td>Q3</td>
67         <td>Q4</td>
68       </tr>
69     </table>
70   </div>
71   <div id="divFicha">
72     <div>
73       <div>
74         <div>
75           <div>
76             <div>
77               <div>
78                 <div>
79                   <div>
80                     <div>
81                       <div>
82                         <div>
83                           <div>
84                             <div>
85                               <div>
86                                 <div>
87                                   <div>
88                                     <div>
89                                       <div>
90                                         <div>
91                                           <div>
92                                             <div>
93                                             </div>
94                                           </div>
95                                         </div>
96                                       </div>
97                                     </div>
98                                   </div>
99                                 </div>
100                               </div>
101                             </div>
102                           </div>
103                         </div>
104                       </div>
105                     </div>
106                   </div>
107                 </div>
108               </div>
109             </div>
110           </div>
111         </div>
112       </div>
113     </div>
114   </div>
115   <div id="divResult">
116     <div>
117       <div>
118         <div>
119           <div>
120             <div>
121               <div>
122                 <div>
123                   <div>
124                     <div>
125                       <div>
126                         <div>
127                           <div>
128                             <div>
129                               <div>
130                                 <div>
131                                   <div>
132                                     <div>
133                                       <div>
134                                         <div>
135                                           <div>
136                                             <div>
137                                             </div>
138                                           </div>
139                                         </div>
140                                       </div>
141                                     </div>
142                                   </div>
143                                 </div>
144                               </div>
145                             </div>
146                           </div>
147                         </div>
148                       </div>
149                     </div>
150                   </div>
151                 </div>
152               </div>
153             </div>
154           </div>
155         </div>
156       </div>
157     </div>
158   </div>
159   </body>
160 </html>

```



```

50
51         // Queueing animations
52         animationQueue = data.animations;
53
54         if (data.valid) {
55             alert("Cadena v lida");
56         }
57     });
58 });
59
60 // Node animation dequeuing
61 setInterval(function () {
62     if (animationQueue.length == 0) {
63         $("#titleResult").text(result);
64         return;
65     }
66
67     $(".step").remove();
68
69     // Animating current
70     currentAnimation = animationQueue.shift();
71     $("#divProgress").html(`(${currentAnimation.nodeName})<br>
72     Entrada: ${currentAnimation.input}<br>
73     Pila: ${currentAnimation.stack}`
74 );
75
76     }, animationTime);
77 });
78 </script>
79 </head>
80
81 <body class="d-block">
82     <div id="div-alert-container" class="d-flex justify-content-center
83     fixed-top">
84         <div id="div-alert" class="alert text-justify" style="display:
85         none;"></div>
86     </div>
87     <div class="container">
88         <div class="row">
89             <div class="d-flex w-100 flex-column">
90                 <%- include('../partials/header'); %>
91
92                 <main role="main" class="inner cover text-center w-100">
93                     <h1 class="cover-heading">Automata de Pila</h1>
94                     <p>
95                         Implementaci n de un automata de pila con el lenguaje
96                         libre de contexto  $\{0^n 1^n \mid n \geq 1\}$ .
97                     </p>
98                     <br>
99                     <div class="row d-flex justify-content-center">
100                         <div id="divControles" class="col-sm-6 text-center">
101                             <div id="divInput" class="form-group">
102                                 <input type="text" class="form-control" id="
103                                     txtBinario" placeholder="Inserte la entrada">

```

```

100         <small class="form-text text-muted">Dejar esto
           vac o para que se genere aleatoriamente.</small
           >
101     </div>
102 </div>
103 </div>
104 <br>
105 <div id="divStart">
106     <button id="btnStart" type="button" class="btn
           btn-primary">Iniciar</button>
107 </div>
108 </main>
109 </div>
110 </div>
111 </div>
112 <br>
113 <div class="d-block">
114     <div id="divContent" class="d-block text-center">
115
116         <h1>Proceso:</h1>
117         <h2 id="titleResult"></h2>
118         <br><br><br>
119         <h3>
120             <div id="divProgress"></span>
121         </h3>
122     </div>
123 </div>
124 </body>
125
126 </html>

```

```

1
2 @media only screen and (min-width: 768px) and (max-width: 991px) {
3
4     #main {
5         width: 712px;
6         padding: 100px 28px 120px;
7     }
8
9     /* .mono {
10         font-size: 90%;
11     } */
12
13     .cssbtn a {
14         margin-top: 10px;
15         margin-bottom: 10px;
16         width: 60px;
17         height: 60px;
18         font-size: 28px;
19         line-height: 62px;
20     }

```

```

1 class TelegramRequestHandler(object):

```

```

2     def handle(self):
3         addr = self.client_address[0]           # Client IP-address
4         telgram = self.request.recv(1024)       # Recieve telgram
5         print "From: %s, Received: %s" % (addr, telgram)
6         return

```

```

1
2 @media only screen and (min-width: 768px) and (max-width: 991px) {
3
4     #main {
5         width: 712px;
6         padding: 100px 28px 120px;
7     }
8
9     /* .mono {
10         font-size: 90%;
11     } */
12
13     .cssbtn a {
14         margin-top: 10px;
15         margin-bottom: 10px;
16         width: 60px;
17         height: 60px;
18         font-size: 28px;
19         line-height: 62px;
20     }

```



Figura 1: Tabla de Conversión

3. Resultados

Autómata ejecutándose:



Figura 2: Tabla de Conversión