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 { $0^n 1^n$ — n >= 1 }.

2. Códigos

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

```
const xlsxFile = require("read-excel-file/node");
const bodyParser = require("body-parser");
const readline = require("readline");
const express = require("express");
const aNode = require("./aNode.js");
const path = require("path");
const fs = require("fs");
const app = express();

const http = require("http").createServer(app);
const webPort = 8080;

const excelPath = path.join(__dirname, "archivos", "ajedrez.xlsx");

var transitionTree;
var stack = [];

// EJS INIT
// Set the view engine to ejs
app.set("view engine", "ejs");
// Set ejs files path
app.set("views", __dirname + "/dist/pages");
// Set body parser
app.use(bodyParser.urlencoded({ extended: true }));

// REQUESTS
app.get("/", (req, res) => {
  res.render("index");
});

app.post("/procesar/cadena/", (req, res) => {
  let response = {};
  let output;

  if (req.body.input.length == 0) {
    req.body.input = generateBin(10);
  }

  output = automata(req.body.input);

  console.log(output);

  response = output.response;
  response.binario = req.body.input;

  fs.writeFile("./archivos/proceso.txt", output.file, () => {});

  // Send response
  res.send(JSON.stringify(response));
});

function generateBin(len) {
  let resultado = "";
  for (let i = 0; i < len; ++i) {
```

```
    resultado += Math.floor(Math.random() * 2);
  }
  return resultado;
}

// AUTOMATA
// Main automata functionality
function automata(cadena) {
  let limi = cadena.length;
  let current, next;
  let response = {
    animations: [],
  };
  let result = {
    file: "",
  };

  stack = [];
  stack.push("F");

  current = [startNode];
  response.animations.push({
    nodeName: current[0].name,
    input: cadena,
    stack: stack.join(""),
  });

  result.file += `d(${current[0].name}, ${cadena}, ${stack.join("")})
    `;
  for (let i = 0; i < limi; ++i) {
    next = [];
    current.forEach((nodo) => {
      next = next.concat(
        nodo.evaluateChar(cadena[i], stack[stack.length - 1])
      );
    });

    // Removing duplicates
    current = [...new Set(next)];

    // Set animation output
    response.animations.push({
      nodeName: current[0].name,
      input:
        cadena.substring(i + 1).length == 0
          ? "e"
          : cadena.substring(i + 1),
      stack: stack.join(""),
    });

    //Set file output
    result.file += `->\nd(${current[0].name}, ${cadena.substring(
      i
    )}, ${stack.join("")})`;
```

```
  }

  // Process remaining stack things
  limi = stack.length;
  for (let i = 0; i < limi; ++i) {
    next = [];
    current.forEach((nodo) => {
      next = next.concat(nodo.evaluateChar("  ", stack[i]));
    });

    // Removing duplicates
    current = [...new Set(next)];

    // Set animation output
    response.animations.push({
      nodeName: current[0].name,
      input: "e",
      stack: stack.join("").substring(i + 1),
    });

    //Set file output
    result.file += `->\nd(${current[0].name}, e, ${stack.join("")})`;
  }

  if (current[0].name == "qf") {
    response.result = "La cadena es v lida!"
  } else {
    response.result = "La cadena no es v lida :c"
  }

  // Add animation list
  result.response = response;

  return result;
}

// Create automata
function initAutomata() {
  startNode = new aNode("q0");
  let q1 = new aNode("q1");
  let q2 = new aNode("qf");

  q2.callback = (char) => {
    // Input was valid
    console.log("Input was valid");
  };

  q1.nextNodes.push({
    input: "  ",
    stack: "F",
    node: q2,
    callback: () => {},
  });
  q1.nextNodes.push({
```

```
      input: "1",
      stack: "  ",
      node: q1,
      callback: () => {
        // Pop from stack
        stack.pop();
      },
    });

    startNode.nextNodes.push({
      input: "0",
      stack: "  ",
      node: startNode,
      callback: () => {
        // Push 'a' to stack
        stack.push("a");
      },
    });
    startNode.nextNodes.push({
      input: "1",
      stack: "  ",
      node: q1,
      callback: () => {
        // Pop from stack
        stack.pop();
      },
    });
}

// SERVER SET-UP
app.use(express.static(__dirname + "/dist/public/"));

// SERVER LISTEN INIT
http.listen(webPort, () => {
  console.log("Listening on port: " + webPort);
});

// Init
initAutomata();
```

```
@media only screen and (min-width: 768px) and (max-width: 991px) {

  #main {
    width: 712px;
    padding: 100px 28px 120px;
  }

  /* .mono {
    font-size: 90%;
  } */

  .cssbtn a {
    margin-top: 10px;
```

```
    margin-bottom: 10px;
    width: 60px;
    height: 60px;
    font-size: 28px;
    line-height: 62px;
  }
```

```
class TelgramRequestHandler(object):
    def handle(self):
        addr = self.client_address[0]          # Client IP-adress
        telgram = self.request.recv(1024)      # Recieve telgram
        print "From: %s, Received: %s" % (addr, telgram)
        return
```

```

@media only screen and (min-width: 768px) and (max-width: 991px) {

  #main {
    width: 712px;
    padding: 100px 28px 120px;
  }

  /* .mono {
    font-size: 90%;
  } */

  .cssbtn a {
    margin-top: 10px;
    margin-bottom: 10px;
    width: 60px;
    height: 60px;
    font-size: 28px;
    line-height: 62px;
  }
```

```
<!DOCTYPE html>
<html>

<head>
  <%- include('../partials/head'); %>

    <script>
      var animationTime = 1000;
      var animationQueue = [];
      var currentAnimation;
      var result;

      function movePiece(piece, elementBoard, time) {
        return piece.animate({
          left: elementBoard.offset().left + ((elementBoard.outerWidth() - piece.outerWidth()) / 2),
          top: elementBoard.offset().top + ((elementBoard.outerHeight() - piece.outerHeight()) / 2)
        }, time).promise();
      }
      async function transportPiece(idOrigin, idDestination, time = animationTime) {
        let piece = $("#divFicha").clone().removeAttr("id").addClass("step").removeClass("d-none").appendTo("body");

        let elementOrigin = findBoardElement("Q" + idOrigin);
        let elementDestination = findBoardElement("Q" + idDestination);

        await movePiece(piece, elementOrigin, 0);
        await movePiece(piece, elementDestination, time);
      }


      function findBoardElement(idBoard) {
        return $("td").filter(function () {
          return $(this).text() == idBoard;
        });
      }

      $(document).ready(function () {
        $("#btnStart").click(function () {
          $.post("/procesar/cadena/", {
            input: $("#txtBinario").val()
          }, function (data, status) {
            data = JSON.parse(data);

            // Setting input text
            $("#txtBinario").val(data.binario);

            // Setting future result text
            result = data.result;

            console.log(data);
```

```
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
          fixed-top">
83      <div id="div-alert" class="alert text-justify" style="display:
            none;"></div>
84    </div>
85    <div class="container">
86      <div class="row">
87        <div class="d-flex w-100 flex-column">
88          <%- include('../partials/header'); %>
89
90            <main role="main" class="inner cover text-center w-100">
91              <h1 class="cover-heading">Automata de Pila</h1>
92              <p>
93                Implementaci n de un automata de pila con el lenguaje
                      libre de contexto {0^n 1^n | n >= 1}.
94              </p>
95              <br>
96              <div class="row d-flex justify-content-center">
97                <div id="divControles" class="col-sm-6 text-center">
98                  <div id="divInput" class="form-group">
99                    <input type="text" class="form-control" id="
                          txtBinario" placeholder="Inserte la entrada">
```

```
                    <small class="form-text text-muted">Dejar esto vac o para que se genere aleatoriamente.</small>
                  </div>
                </div>
              </div>
              <br>
              <div id="divStart">
                <button id="btnStart" type="button" class="btn btn-primary">Iniciar</button>
              </div>
            </main>
        </div>
      </div>
    </div>
    <br>
    <div class="d-block">
      <div id="divContent" class="d-block text-center">

        <h1>Proceso:</h1>
        <h2 id="titleResult"></h2>
        <br><br><br>
        <h3>
          <div id="divProgress"></span>
        </h3>
      </div>
    </div>
  </body>

  </html>
```

```
@media only screen and (min-width: 768px) and (max-width: 991px) {

  #main {
    width: 712px;
    padding: 100px 28px 120px;
  }

  /* .mono {
    font-size: 90%;
  } */

  .cssbtn a {
    margin-top: 10px;
    margin-bottom: 10px;
    width: 60px;
    height: 60px;
    font-size: 28px;
    line-height: 62px;
  }
```

```
class TelgramRequestHandler(object):
```

```
    def handle(self):
        addr = self.client_address[0]          # Client IP-adress
        telgram = self.request.recv(1024)      # Recieve telgram
        print "From: %s, Received: %s" % (addr, telgram)
        return
```

```

@media only screen and (min-width: 768px) and (max-width: 991px) {

  #main {
    width: 712px;
    padding: 100px 28px 120px;
  }

  /* .mono {
    font-size: 90%;
  } */

  .cssbtn a {
    margin-top: 10px;
    margin-bottom: 10px;
    width: 60px;
    height: 60px;
    font-size: 28px;
    line-height: 62px;
  }
```

Figura 1: Tabla de Conversión

3. Resultados

Autómata ejecutándose:

Figura 2: Tabla de Conversión