Instituto Politécnico Nacional

Escuela Superior de Cómputo

Práctica #5

Tablero

Axel Treviño Palacios
2CM5
09 de enero de 2020

1. Objetivo

Elaborar un programa para realizar movimientos ortogonales y diagonales en un tablero de ajedrez de 4x4 con dos piezas. Los movimientos y las reglas están explicadas en las láminas del curso de Stanford.

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 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;

// 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.auto) {
    if (req.body.input.length == 0) {
      req.body.input = generateMoves(20);
    }
  } else {
    req.body.input = generateMoves(10);
  }

  output = automata(req.body.input, 1, 16);

  response.moves = req.body.input;
  response.animations = output.animations;
  response.winner = output.winner;

  if (output.winner) {
    result.message = "Cadena ganadora!";
    fs.writeFile("./archivos/ganadores.txt", cadena, () => {});
  }

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

```
55
56  function generateMoves(num) {
57    let resultado = "";
58    for (let i = 0; i < num; ++i) {
59      resultado += Math.floor(Math.random() * 2) == 1 ? "r" : "b";
60    }
61    return resultado;
62  }
63
64  // AUTOMATA
65  // Main automata functionality
66  function automata(cadena, startNode, winNode) {
67    let current, next;
68    let animations = [];
69    let result = {};
70
71    current = [startNode];
72    animations.push(current);
73
74    cadena.split("").map((currentChar) => {
75      next = [];
76      current.forEach((nodo) => {
77        next = next.concat(processNode(currentChar, nodo));
78      });
79
80      // Removing duplicates
81      current = [...new Set(next)];
82
83      // Adding step to animation queue
84      animations.push(current);
85    });
86
87    // Check if winning condition
88    result.winner = current.includes(winNode);
89
90    // Add animation list
91    result.animations = animations;
92
93    return result;
94  }
95  // Node processing
96  function processNode(currentChar, nodeName) {
97    let resultado = [];
98
99    transitionTree.forEach((transition) => {
100     if (transition.nombre == nodeName) {
101       transition.pasos.forEach((paso) => {
102         if (paso.origen == currentChar) {
103           resultado = resultado.concat(paso.destinos);
104         }
105       });
106     }
107   });
108
```

```
  return resultado;
}
// Generate transition tree
function generateTree() {
  xlsxFile("./table.xlsx").then((rows) => {
    transitionTree = [];

    // Removing first row
    rows.shift();

    // Add a transition for each row
    rows.forEach((row) => {
      transitionTree.push({
        nombre: row[0],
        pasos: [
          {
            origen: "r",
            destinos: row[1]
              .toString()
              .split(",")
              .map((x) => +x),
          },
          {
            origen: "b",
            destinos: row[2]
              .toString()
              .split(",")
              .map((x) => +x),
          },
        ],
      });
    });
  });
}

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

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

// Init
generateTree();
```

```

@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 currAnimation, lastAnimation;

      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: $("#txtMovimientos").val(),
            auto: $("input[name=radioModo]:checked").val() == "auto",
            num: $("input[name=radioNum]:checked").val() == "1" ? 1 : 2
          }, function (data, status) {
            data = JSON.parse(data);

            // Setting move text
            $("#txtMovimientos").val(data.moves);

            // Queueing animations
            animationQueue = data.animations;
```

```
49
50              // Setting up first animation
51              lastAnimation = animationQueue.shift();
52
53              if (data.winner) {
54                alert("Movimientos ganadores!");
55              }
56            });
57          });
58
59          // Node animation dequeuing
60          setInterval(function () {
61            if (animationQueue.length == 0) {
62              return;
63            }
64
65            $(".step").remove();
66
67            currAnimation = animationQueue.shift();
68
69            // Animating current
70            lastAnimation.forEach(lastStep => {
71              currAnimation.forEach(newStep => {
72                transportPiece(lastStep, newStep);
73              });
74            });
75
76            lastAnimation = currAnimation;
77          }, animationTime);
78
79        });
80      </script>
81  </head>
82
83  <body class="d-block">
84    <div id="div-alert-container" class="d-flex justify-content-center
        fixed-top">
85      <div id="div-alert" class="alert text-justify" style="display:
          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">Tablero de Ajedrez</h1>
94              <p>
95                Simula todos los movimientos con un tablero de ajedrez
                      de 4x4
96              </p>
97              <br>
98              <div class="row d-flex justify-content-center">
99                <div id="divModo" class="col-sm-6 text-center">
```

```
100                 <div class="form-check">
101                   <input class="form-check-input" name="radioModo" id=
                          "radioModoAuto" type="radio" value="auto"
                          checked>
102                   <label class="form-check-label" for="radioModoAuto">
103                     Autom tico
104                   </label>
105                 </div>
106                 <div class="form-check">
107                   <input class="form-check-input" name="radioModo" id=
                          "radioModoManual" type="radio" value="manual">
108                   <label class="form-check-label" for="radioModoManual
                          ">
109                     Manual
110                   </label>
111                 </div>
112                 <br>
113                 <div id="divMovimientos" class="form-group">
114                   <input type="text" class="form-control" id="
                          txtMovimientos" placeholder="Inserte los
                          movimientos">
115                   <small class="form-text text-muted">Dejar esto
                          vac o para que se genere aleatoriamente.</small
                          >
116                 </div>
117               </div>
118               <div id="divNum" class="col-sm-6 d-none">
119                 <span class="float-left">
120                   <div class="form-check">
121                     <input class="form-check-input" name="radioNum"
                            id="radioNum1" type="radio" value="1" checked>
122                     <label class="form-check-label" for="radioNum2">
123                       Uno
124                     </label>
125                   </div>
126                   <div class="form-check">
127                     <input class="form-check-input" name="radioNum"
                            id="radioNum2" type="radio" value="2">
128                     <label class="form-check-label" for="radioNum2">
129                       Dos
130                     </label>
131                   </div>
132                   <div class="form-group">
133                     <label id="lblFichaAuto"></label>
134                   </div>
135                 </span>
136               </div>
137             </div>
138             <br>
139             <div id="divControl">
140               <button id="btnStart" type="button" class="btn
                      btn-primary">Iniciar</button>
141             </div>
142           </main>
```

```
      </div>
    </div>
  </div>
  <br>
  <div class="d-block">
    <div class="d-flex justify-content-center">
      <div class="col-auto">
        <table class="table table-responsive">
          <tr>
            <td class="bg-dark text-center align-middle">Q1</td>
            <td class="bg-danger text-center align-middle">Q2</td>
            <td class="bg-dark text-center align-middle">Q3</td>
            <td class="bg-danger text-center align-middle">Q4</td>
          </tr>
          <tr>
            <td class="bg-danger text-center align-middle">Q5</td>
            <td class="bg-dark text-center align-middle">Q6</td>
            <td class="bg-danger text-center align-middle">Q7</td>
            <td class="bg-dark text-center align-middle">Q8</td>
          </tr>
          <tr>
            <td class="bg-dark text-center align-middle">Q9</td>
            <td class="bg-danger text-center align-middle">Q10</td>
            <td class="bg-dark text-center align-middle">Q11</td>
            <td class="bg-danger text-center align-middle">Q12</td>
          </tr>
          <tr>
            <td class="bg-danger text-center align-middle">Q13</td>
            <td class="bg-dark text-center align-middle">Q14</td>
            <td class="bg-danger text-center align-middle">Q15</td>
            <td class="bg-dark text-center align-middle">Q16</td>
          </tr>
        </table>
      </div>
    </div>
  </div>

  <div id="divFicha" class="d-none position-absolute">
    <img class="imgFicha" src="/img/fichaAzul.png" alt="Ficha">
  </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;
  }
```

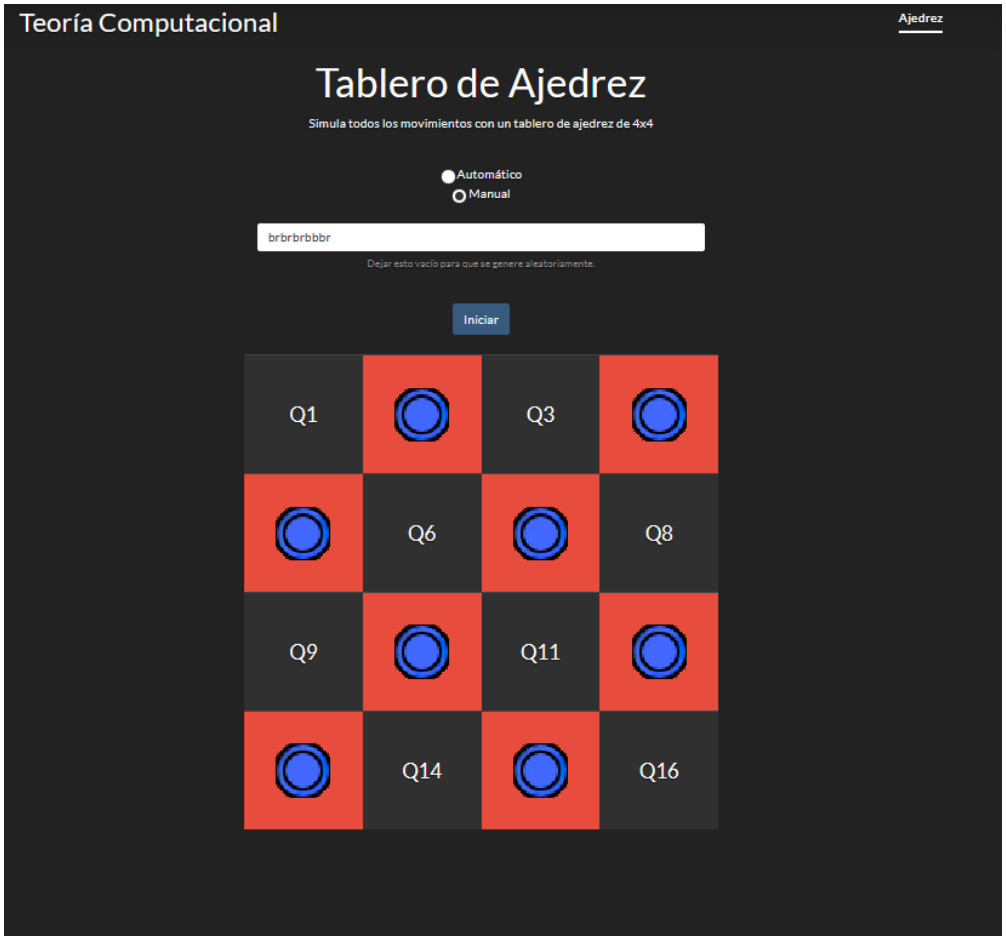

Figura 1: Tabla de Conversión

3. Resultados

Autómata ejecutándose:

Figura 2: Tabla de Conversión