

PROFESSIONAL TRAINING REPORT
at
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for
the award of B.Tech (Bachelor in Technology) Bachelor
of Technology Degree in Information Technology

By

Balaji Manchi
REG. NO. 39120015



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

**SATHYABAMA INSTITUTE OF SCIENCE AND
TECHNOLOGY**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU**

APRIL 2022



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade “A” by NAAC
(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI
CHENNAI– 600119
www.sathyabama.ac.in

DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Balaji Manchi**(Reg. No: 39120015) who carried out the project entitled “**Web Based Game With Two Players(Dots-And-Boxes)**” under my supervision from February 2022 to April 2022.

Internal Guide

Dr.R.M.Gomathi, M.Tech., P.hD. IT Dept
Dr.Subhashini, M.E., P.hD

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I , **Balaji Manchi** hereby declare that the project report entitled “**Web Based Game With Two Members**” done by me under the guidance of **Dr.R.M.Gomathi, M.Tech., P.hD. IT Dept**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in Information technology.

DATE:

A handwritten signature in dark ink, reading "M. Balaji", with a horizontal line underneath.

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D., Dean**, School of Computing, **Dr.Subhashini, M.E., Ph.D., Heads of the Department of Information Technology** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. R.M.Gomathi, M.Tech., P.hD., IT Dept.**, for her valuable guidance, suggestions, and constant encouragement that paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE

Abstract

As one important computer game, Dots-and-Boxes is always being taken as one project of the international computer Olympic tournament and the Chinese college students' computer game, and has already been listed in the 2015 Chinese intellectual games.

In this paper, firstly, the rules and common terminology of Dots-and-Boxes was introduced, then the chess board representing method, the strategy influencing the game outcome,

And the equilibrium solution to the chess state are analyzed.

Our program adopting the strategies obtained 2015 Students Computer Game Championship.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	6
	LIST OF FIGURES	9
1	INTRODUCTION	10
1.1	WEB BASED GAME WITH TWO PLAYERS	10
2	AIM AND SCOPE OF THE PRESENT INVESTIGATION	11
2.1	BRIEF INTRODUCTION OF GAME RULES	11
2.2	GAME RULES	11
2.3	BASIC CONCEPTS OF DOTS-AND-BOXES	13
3	ANALYSIS ON MAIN TECHNICAL IN DOTS-AND-BOXES	15
3.1	TWO IMPORTANT STARTERGERGIES INFLUENCING THE OUTCOME	15
3.2	EQUILIBRIUM SOLUTION IN DOTS-AND-BOXES	16
4	TECHNOLOGIES	18
4.1	HTML	18
4.2	CSS	18
4.3	JAVASCRIPT	19
5	RESULT	21
5.1	WELCOME SCREEN GUI	21

5.2	GAME PAGE AT START	21
5.3	GAME PAGE AT END	22
6	CONCLUSION	23
7	SOURCE CODE	24
7.1	WELCOME SCREEN HTML	24
7.2	WELCOME SCREEN CSS	25
7.3	DOTS AND BOXES HTML	26
7.4	DOTS AND BOXES CSS	29
7.5	DOTS AND BOXES JS	29

LIST OF FIGURES

FIGURE NO.	FIGURE TITLE	PAGE NO.
2.1	Dots At Start	11
2.2	Board During Game	12
2.3	Board At The End	13
5.1	Welcome-Screen GUI	21
5.2	Game Page At Start	21
5.3	Game Page At End	22

CHAPTER-1

INTRODUCTION:

1.1 WEB BASED GAME WITH TWO PLAYERS

Computer game is an important research subject of artificial intelligence, and the development of artificial intelligence itself mainly profit from the development of computer game research, in which, the notable events in 1997 of Deep Blue beating Garry Kasparov, the world chess champion, became one important milestone.

Invented in 1891 by Edward Lucas, one French mathematician, Dots-and-Boxes[2] is a two-player game that uses paper and pen.

At an earlier time, this game is very popular in European, being taken as the project of international computer Olympic tournament game.

By now, many well-known international institutions have taken part in the development of related software for Dots-and-Boxes, for example, Elmo Timoteus with the Mathematics Department of UCLA (the University of California at Los Angeles) has developed one software for Dots-and-Boxes, which not only of high efficiency but also can adjust the board size at any time during the game.

In recent years, with the development of the Chinese computer game championship, Dots-and-Boxes has gradually been known by computer game enthusiasts.

By now, Dots-and-Boxes has been listed as one of the major computer competition games, and it also is the game that has the most participator number. At present, only a few article have as comprehensively introduce Dots-and-Boxes. This paper will mainly introduce the basic concept of Dots-and-Boxes, study the board representation and winning fact, etc.

CHAPTER-2

AIM AND SCOPE:

2.1. Brief introduction of Game rules and basic concepts:

Dots-and-Boxes is a typical category of Tim games, at present, in the international computer Olympic tournament and the national college computer game competition, the board of 6 x 6, as shown in Figure 2.1, is adopted.

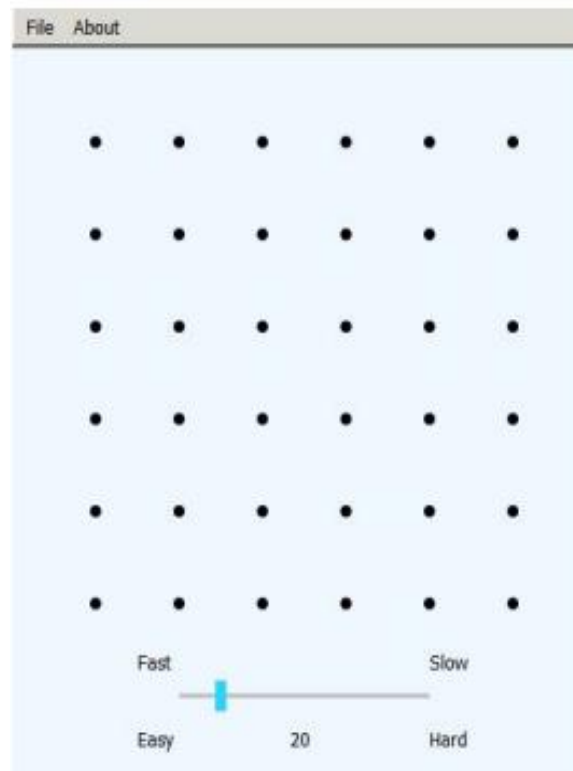


Figure-2.1 Dots At Start

2.2Game rules:

The game rules of Dots-and-Boxes are that.

(1) The sides make move alternately. When one side moves, it connects one line between the two dots near the side, abiding the limitation of

that: the line cannot cross any dot, between two dots there only allows one line, and they're also not allow any diagonal line.

(2) Every box is surrounded by four sides, and the owner of the last edge is the owner of the box.

(3) After owning the box by adding the last line, the winning side can further move, until there is no more unoccupied box.

(4) After all the boxes are occupied, the game ends, and the side owning a larger number of the box wins.

Below Figure 2.1 to 2.3 are three main interfaces during the game.

From Figure 2.3, we can see that in the end, the sides owning blue boxes win.

Generally, for the sake of fairness, each competition will run two times, thus each side will have a chance been the first player or the second player since being the first or the second will have different advantages.

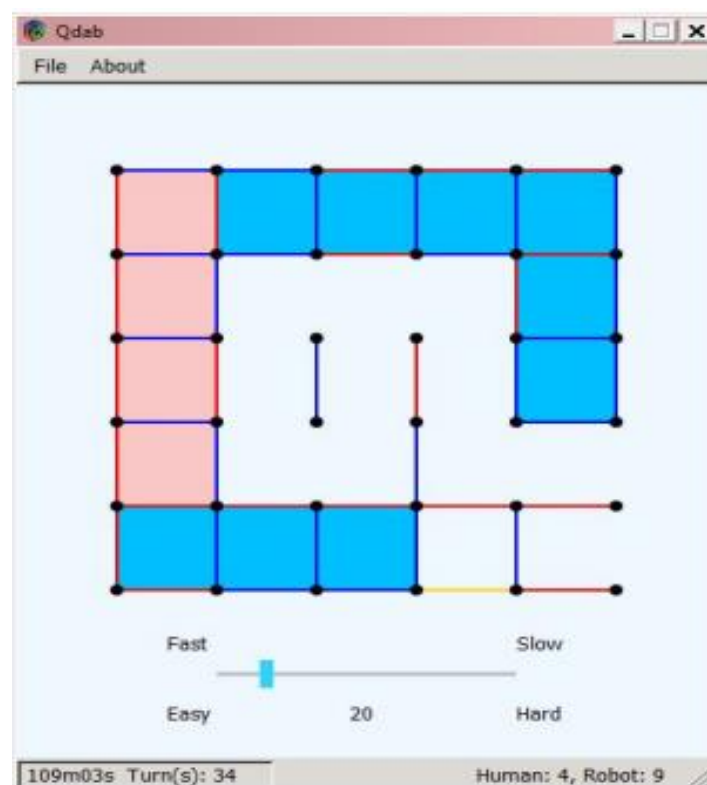


Figure-2.2 Board During Game

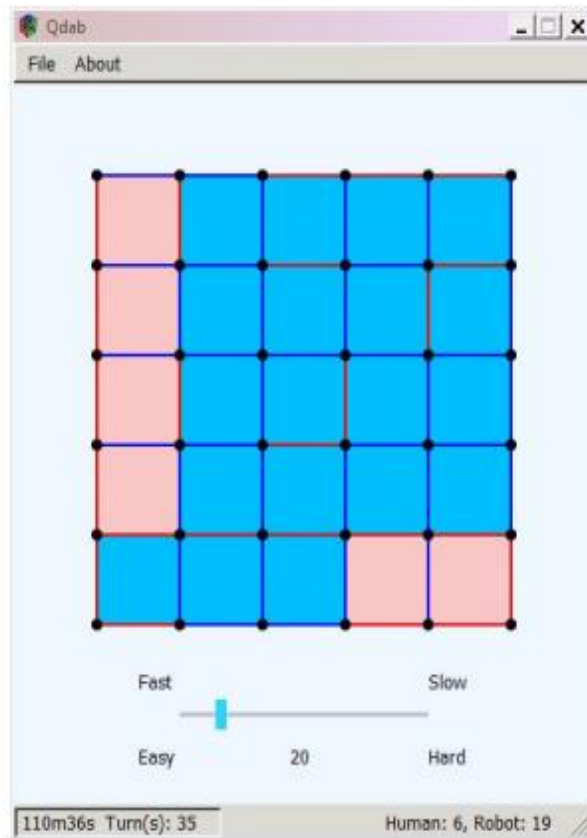


Figure-2.3:Board at the end

2.3 Basic concepts of Dots-and-Boxes:

In the studies of Dots-and-Boxes, some basic concepts are always involved in, which are.

- (1) Degree of freedom. It is the number of unoccupied edges of one box.
- (2) C type box, C type box is the box whose freedom degree is 1.
- (3) Adjacent, refer to two boxes of location (I, j) and (k, l) , only if $|i-k|+|j-l|=1$, and the public side is not occupied.
- (4) One turn, refer to that, after one side moved, the other side start to move.

(5) One move, refer to connecting the line between two adjacent dots.

(6) Filling the box, refer to the move that closes one box.

(7) Chain. If there is a set of boxes $BN=\{b(i)\}, 0 \leq i \leq n$, and each $b(i)$ has freedom degree 2 (that is, this box has unoccupied edges), and of $b(0)$ and $b(n)$, there is at most only one box belong to BS , and the left will have two adjacent boxes, is the adjacent boxes also belong to BN , then BN is one Chain. $b(0)$ and $b(n)$ are the terminal Chain, and they can be in the same box. The chain is divided into a long one and a short one. In which, Short-chain, Short-chain are Chain only consist of 1 or two boxes. A long chain, Long-chain are Chain consist of three or more three boxes.

(8) Circle, means one chain $BN=\{b(i)\}, 0 \leq i \leq n$, each $b(i)$ has freedom degree 2 and has two adjacent boxes, and the two adjacent belongs to BN also, then BN is a Circle. A Circle is generally along a Chain, whose beginning and end box is the same box, so in a Circle there are at least four boxes.

(9) Double cross means that the next move will occupy two boxes.

(10) The Final phase, refers to the phase when there is only long Chain.

(11) Joint, refers to the boxes whose degrees of freedom is 3.

CHAPTER-3

Analysis on main technical in Dots-and-Boxes:

3.1.Two important strategies influencing the outcome :

According to the game rules of Dots and Boxes, if the player in his current turn can occupy the box, then after he occupied the box, he will continues to occupy some edge, until the edge lost the box.

And according to the rules, the player who occupies the last long Chain is more likely to win, since he has taken the initiative in the winning of the former chain, and made the right selection from which he was befitted.

Therefore, the strategies should be adopted in the competition are that.

Strategy 1: If the total number of the point in the board is odd, then the first hand should form an odd number long Chains, ad the second hand should form an even number long Chains.

If total number of the point is even, then the first hand should form an even number of long Chains, and the second hand should form an odd long Chains.

In order to gain the initiative in the game, the side which has first entered one long Chain, who will be allowed two choices, one is to take the greedy algorithm, as possible as to occupy more boxes.

Another is the striking method, that is, in the connecting of the last edge, try to give up four or two boxes, thus make the player himself own more boxes than the rival.

Strategy 2: In the end game, if there are more than one Circles and Chains coexisting, then count the numbers cc box num of the boxes closed by the Chains and Circles, where n denotes the number of Chains, and m that of Circles.

If $cc_box_num - 8m - 4n + 4 < 0$, they chose the greedy algorithm, otherwise, select the striking method.

The player who takes the initiative, will form the last edge of each turn into one Circle if there is one, otherwise, he will form this edge in one Chain, because it formed the edge in a Circle in the next turn he can obtain 4 boxes, and form the edge in a Circle can only obtain 2 boxes.

3.2. Equilibrium solution in Dots-and-Boxes:

The board size of Dots-and-Boxes can be changed, we can set a random board of size $n \times m$,

where n and m mean the dots number row-wise and column-wise, respectively, and the usually used are the 3×3 or 6×6 board.

In the 6×6 board, there are 25 boxes, so there must be one winner and one loser.

The purpose of both side is try to inhibit the opponent and to get more benefits.

Since both sides are rational, so the best result is that the absolute difference between the box numbers of the two sides is minimized, thus this is a typical two player zero-sum game, that is, the sum of benefits obtained by the two sides are the zeros, the income of A is just the outcome of B, and vice versa.

Since we only analyze the solution of Dots-and-Boxes, so here we only consider the final phase of the board.

The two players zero-sum game is usually described by a matrix, like this, if A occupied the boxes of $\{0, 1, 2, \dots, 25\}$,

Then B will own the boxes of $\{25, 24, 23, \dots, 0\}$, the difference between the boxes occupied by A and B is used to represent the income.

So, there is $U = \text{diag}\{-25, -23, \dots, -1, 1, \dots, 23, 25\}$ where 0 means the state that can not arrived at.

According to the method of minimax, there is

$$\max(i)\min(j)u(j) = \max\{-25,-23,\dots,-1\} = -1.$$

$$\min(i)\max(j)u(ij) = \min\{1,3,\dots,21,23,25\} = 1.$$

$$\text{Since } \max(i)\min(j)u(ij) \neq \min(j)\max(i)u(ij)$$

It is not in equilibrium stage.

CHAPTER-4

TECHNOLOGIES:

4.1.HTML: Hyper Text Markup Language:

HTML (Hyper Text Markup Language) is the most basic building block of the Web. It defines the meaning and structure of web content.

Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/behavior (JavaScript).

"Hypertext" refers to links that connect web pages to one another, either within a single website or between websites. Links are a fundamental aspect of the Web.

By uploading content to the Internet and linking it to pages created by other people, you become an active participant in the World Wide Web.

HTML uses "markup" to annotate text, images, and other content for display in a Web browser.

HTML markup includes special "elements" such as <head>, <title>, <body>, <header>, <footer>, <article>, <section>, <p>, <div>, , , <aside>, <audio>, <canvas>, <datalist>, <details>, <embed>, <nav>, <output>, <progress>, <video>, , , and many others.

An HTML element is set off from other text in a document by "tags", which consist of the element name surrounded by " " and " ".

The name of an element inside a tag is case insensitive. That is, it can be written in uppercase, lowercase, or a mixture.

4.2.CSS: Cascading Style Sheets:

CSS is a stylesheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG, MathML or XHTML).

CSS describes how elements should be rendered on screen, on paper, in speech, or on other media.

CSS is among the core languages of the [open web](#) and is standardized across Web browsers according to W3C specifications.

Previously, development of various parts of CSS specification was done synchronously, which allowed versioning of the latest recommendations.

You might have heard about CSS1, CSS2.1, CSS3. However, CSS4 has never become an official version.

From CSS3, the scope of the specification increased significantly and the progress on different CSS modules started to differ so much, that it became more effective to develop and release recommendations separately per module.

Instead of versioning the CSS specification, W3C now periodically takes a snapshot of the latest stable state of the CSS specification.

4.3.JavaScript(JS):

JS is a lightweight, interpreted, or just-in-time compiled programming language with first-class functions.

While it is most well-known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js, Apache CouchDB and Adobe Acrobat.

JavaScript is a prototype-based, multi-paradigm, single-threaded, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles.

[Read more about JavaScript.](#)

This section is dedicated to the JavaScript language itself, and not the parts that are specific to Web pages or other host environments.

For information about API specifics to Web pages, please see [Web APIs](#) and [DOM](#).

The standards for JavaScript are the [ECMAScript Language Specification \(ECMA-262\)](#) and the [ECMAScript Internationalization API specification \(ECMA-402\)](#).

The JavaScript documentation throughout MDN is based on the latest draft versions of [ECMA-262](#) and [ECMA-402](#).

And in cases where some proposals for new ECMAScript features have already been implemented in browsers, documentation and examples in MDN articles may use some of those new features.

Do not confuse JavaScript with the Java programming language.

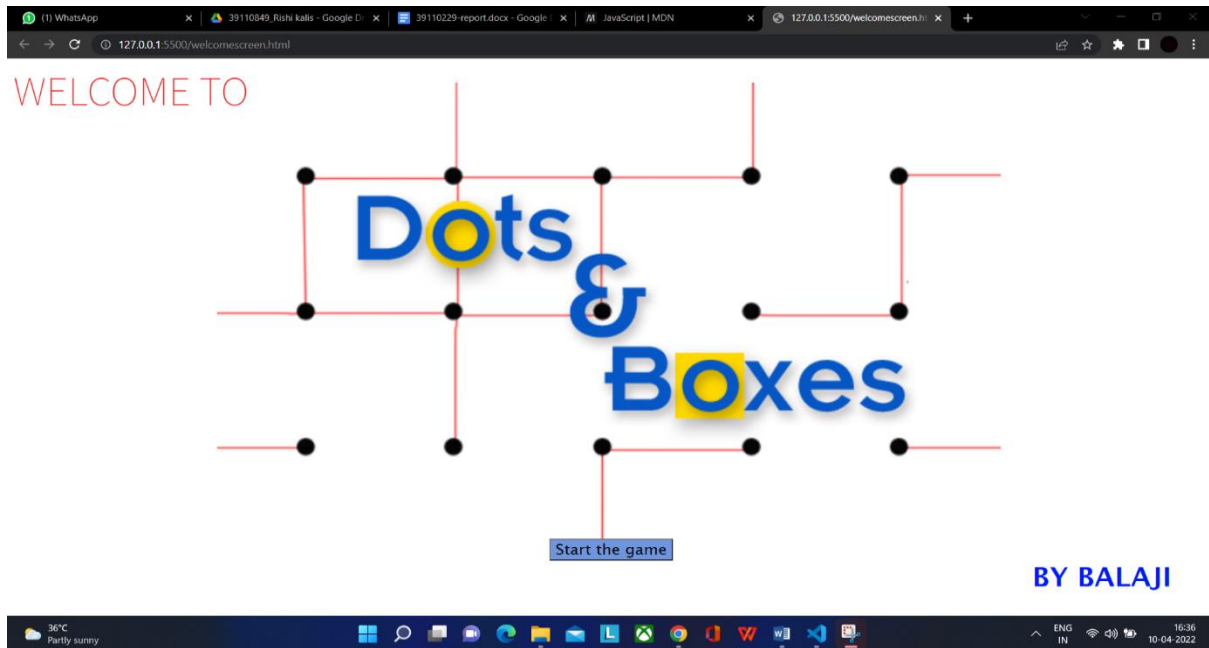
Both "Java" and "JavaScript" are trademarks or registered trademarks of Oracle in the U.S. and other countries.

However, the two programming languages have very different syntax, semantics, and use.

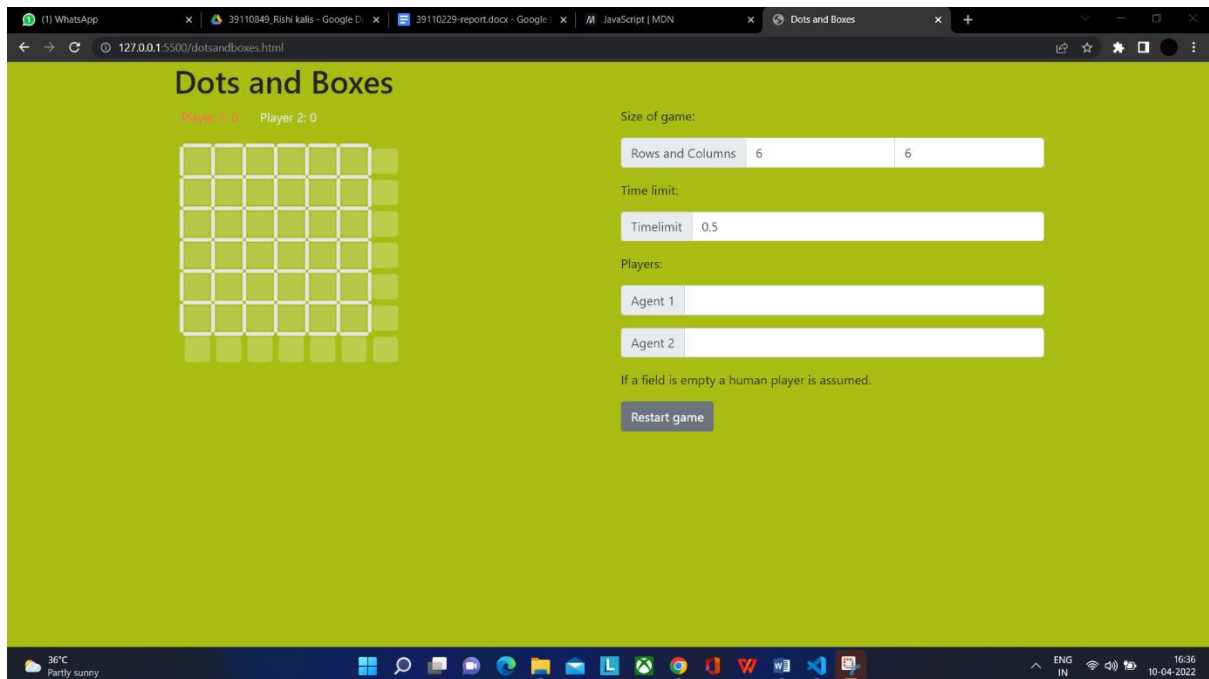
CHAPTER-5

Result:

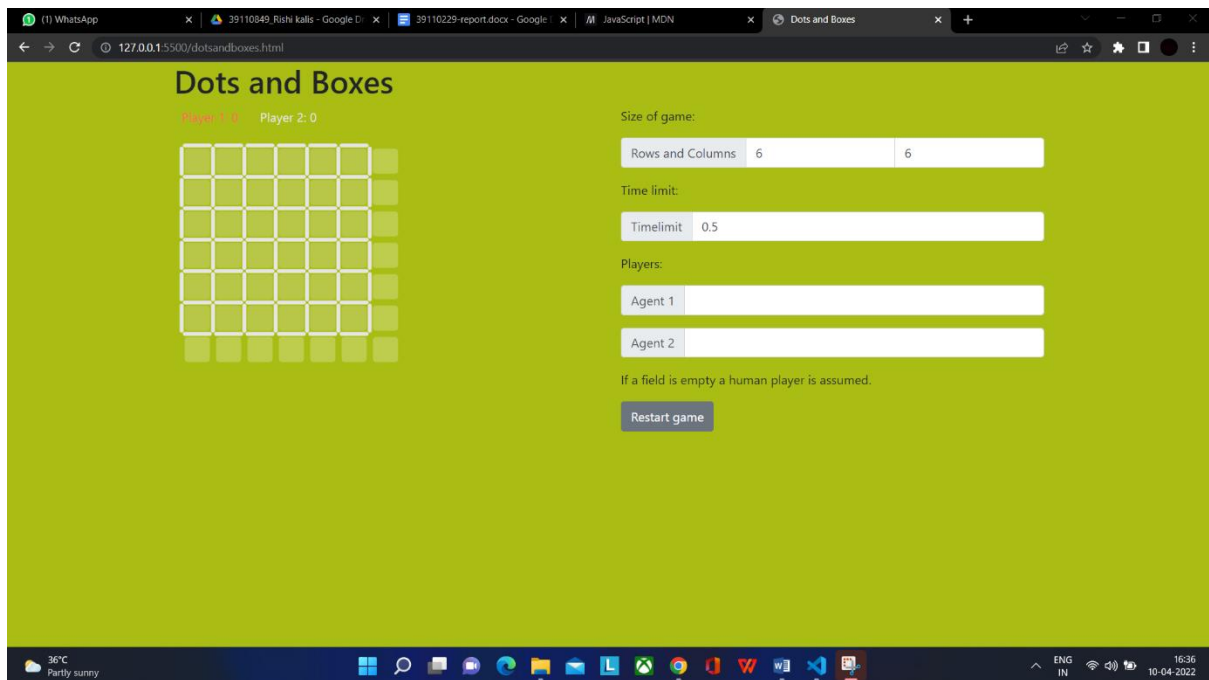
5.1.Welcome Screen GUI:



5.2.Game Page At start:



5.3.Game Page At End:



CHAPTER-6

Conclusion:

As one crystallization of human wisdom, intelligence movements of chess and cards have attracted many a few participators, by its special cultural attraction, and its profound cultural heritage and educational function have also been widely appreciated by current society.

As the competition project of the International Computer Olympic tournament and of the Chinese college students' computer game, it was listed in the Chinese intellectual sports meeting in 2015.

This paper has introduced the origin, the basic concept, the board, as well as the winning factors of Dots-and-Boxes, which will give some lights for the future researches and studies.

Dots-and-Boxes the 2nd National Intelligence Sports Meeting

CHAPTER-7

Source code:

7.1.Welcome Screen HTML:

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta http-equiv="X-UA-Compatible" content="IE=edge">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <link rel="preconnect" href="https://fonts.googleapis.com">

  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

  <link
href="https://fonts.googleapis.com/css2?family=Source+Sans+Pro:wght@200&display=swap"
rel="stylesheet">

  <link rel="stylesheet" href="welcomescreen.css">

  <div><span class="wlc">WELCOME TO</span> <span class="ping"></span> </div>

<h3>

  BY BALAJI

</h3>

  <a href="/dotsandboxes.html"><button>Start the game</button> </a>

</head>

<body>
```

```
</body>
```

```
</html>
```

7.2.Welcome Screen CSS:

```
div {  
    font-family: 'Source Sans Pro', sans-serif;  
    text-align: left;  
    font-size: 55px;  
    margin-bottom: 20%;  
    border-color: rgb(81, 0, 255) 15px;  
}
```

```
.wlc {  
    color: rgb(231, 31, 31);  
}
```

```
.ping {  
    color: rgb(8, 0, 255);  
}
```

```
h3 {  
    position: absolute;  
    font-size: 35px;  
    margin-top: 290px;  
    margin-left: 1300px;  
    color: rgb(0, 26, 255);
```

```
    font-family: 'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida Sans Unicode', Geneva,
    Verdana, sans-serif;
}
```

```
button {

    margin-left: 45%;

    margin-top: 260px;

    color: black;

    font-family: 'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida Sans Unicode', Geneva,
    Verdana, sans-serif;

    font-size: 20px;

    background-color: rgb(111, 150, 223);

}
```

```
body {

    background-image: url("./qwertasdfgh.png");

    background-repeat: no-repeat;

    background-size: 1000px;

    background-position: center;

}
```

7.3.Dots And Boxes HTML:

```
<!DOCTYPE html>

<html>

<html lang="en">

<title>Dots and Boxes</title>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">

<link rel="stylesheet" href="dotsandboxes.css">
```

```
</head>
```

```
<body>
```

```
  <div class="container">
```

```
    <h1>Dots and Boxes</h1>
```

```
    <div class="row">
```

```
      <div class="col-md">
```

```
        <div id="playing-area"></div>
```

```
      </div>
```

```
      <div class="col-md">
```

```
        <div class="form-group">
```

```
          <p>Size of game:</p>
```

```
          <div class="input-group">
```

```
            <div class="input-group-prepend">
```

```
              <span class="input-group-text">Rows and Columns</span>
```

```
            </div>
```

```
            <input type="number" class="form-control" id="nb-rows" value=6 min=1>
```

```
            <input type="number" class="form-control" id="nb-cols" value=6 min=1>
```

```
          </div>
```

```
        </div>
```

```
        <div class="form-group">
```

```
          <p>Time limit:</p>
```

```
          <div class="input-group">
```

```
            <div class="input-group-prepend">
```

```
              <span class="input-group-text">Timelimit</span>
```

```
            </div>
```

```
            <input type="number" class="form-control" id="timelimit" value=0.5 step=0.5 min=0>
```

```
          </div>
```

```

</div>

<div class="form-group">

  <p>Players:</p>

  <div class="input-group mb-3">

    <div class="input-group-prepend"><span class="input-group-text" id="basic-addon3">Agent 1</span></div>

    <input type="text" class="form-control" id="agent1" aria-describedby="basic-addon3">

  </div>

  <div class="input-group mb-3">

    <div class="input-group-prepend"><span class="input-group-text" id="basic-addon3">Agent 2</span></div>

    <input type="text" class="form-control" id="agent2" aria-describedby="basic-addon3">

  </div>

  <p>If a field is empty a human player is assumed.</p>

  <p></p>

  <button type="button" class="btn btn-secondary" id="restart-btn">Restart game</button>

</div>

</div>

</div>

<div class="footer">

  <small>&copy; <a href="https://github.com/Balaji8403/dots-and-boxes/tree/main/DAB">Source</a></small>

</div>

</div>

<script src="https://d3js.org/d3.v4.min.js"></script>

<script src="dotsandboxes.js"></script>

</body>

</html>

```

7.4Dots And Boxes CSS:

```
body {  
    background-color: rgb(255, 255, 255);  
}
```

```
.footer {  
    color: #bb4747;  
    margin-bottom: 1ex;  
}
```

```
.footer a {  
    color: #000000;  
}
```

7.5Dots And Boxes JS:

```
function generateGuid() {  
    var result, i, j;  
    result = "";  
    for(j=0; j<32; j++) {  
        if( j == 8 || j == 12 || j == 16 || j == 20)  
            result = result + '-';  
        i = Math.floor(Math.random()*16).toString(16).toUpperCase();  
        result = result + i;  
    }  
    return result;  
}
```

```
// GAME LOGIC
```

```
var cur_game = generateGuid();  
var cur_player = 1;  
var cur_ended = false;  
var points = [0, 0, 0];  
var timelimit = 0.5;  
var nb_cols = 6;  
var nb_rows = 6;  
var data = new Array(0);  
var alertstring = "";
```

```
function restart_game() {  
    //console.log("Restarting game");
```

```

cur_game = generateGuid();
alertstring = "";
nb_cols = parseInt(document.getElementById('nb-cols').value);
if (nb_cols == "" || isNaN(nb_cols) || nb_cols < 1) {
    nb_cols = 6;
    alertstring += "Invalid number of columns, the default value of " + nb_cols.toString() + " is
applied. ";
}
nb_rows = parseInt(document.getElementById('nb-rows').value);
if (nb_rows == "" || isNaN(nb_rows) || nb_rows < 1) {
    nb_rows = 6;
    alertstring += "Invalid number of rows, the default value of " + nb_rows.toString() + " is
applied. ";
}
timelimit = parseFloat(document.getElementById('timelimit').value);
if (timelimit == "" || isNaN(timelimit) || timelimit < 0) {
    timelimit = 0.5 ;
    alertstring += "Invalid time limit, the default value of " + timelimit.toString() + " is applied. ";
}
if (alertstring != ""){
    alert(alertstring);
}
cur_ended = false;
console.log("Starting game", cur_game);
points = [0, 0, 0];
cur_player = 1;
var old_length = 0;
for (var ri=0; ri<nb_rows + 1; ri++) {
    if (ri >= data.length) {
        data.push(new Array(0));
    }
    var row = data[ri];
    for (var ci=0; ci<nb_cols + 1; ci++) {
        if (ci >= row.length) {
            row.push({l:0, t:0, p:0, r:0, c:0});
        }
        var l = 0;
        var t = 0;
        var p = 0;
        if (ri == nb_rows) {
            l = undefined;
            p = undefined;
        }
        if (ci == nb_cols) {
            t = undefined;
            p = undefined
        }
        var cell = row[ci];
        cell.l = l;
        cell.t = t;
        cell.p = p;
        cell.r = ri;
        cell.c = ci;
    }
}

```

```

    old_length = row.length;
    for (var ci=nb_cols + 1; ci<old_length; ci++) {
        row.pop();
    }
}
old_length = data.length;
for (var ri=nb_rows + 1; ri<old_length; ri++) {
    data.pop();
}
}

```

```

function user_click(cell, o) {
    if (cur_ended) {
        //console.log('Game ended, ignoring click');
        return;
    }
    console.log('User click', cell, o);
    var won_cell = false;
    var c = cell.c;
    var r = cell.r;
    var msg = {
        type: "action",
        game: cur_game,
        player: cur_player,
        nextplayer: cur_player,
        score: [points[1], points[2]],
        location: [r, c],
        orientation: o
    };
    if (o == "h") {
        if (cell.t != 0) {
            return;
        }
        cell.t = cur_player;
        // Above
        if (r > 0) {
            if (data[r - 1][c].l != 0
                && data[r - 1][c + 1].l != 0
                && data[r - 1][c].t != 0
                && data[r][c].t != 0) {
                won_cell = true;
                points[cur_player] += 1;
                data[r - 1][c].p = cur_player;
            }
        }
        // Below
        if (r < nb_rows) {
            if (data[r][c].l != 0
                && data[r][c + 1].l != 0
                && data[r][c].t != 0
                && data[r + 1][c].t != 0) {
                won_cell = true;
                points[cur_player] += 1;
                data[r][c].p = cur_player;
            }
        }
    }
}

```

```

    }
}

if (o == "v") {
    if (cell.l != 0) {
        return;
    }
    cell.l = cur_player;
    // Left
    if (c > 0) {
        if (data[r][c - 1].l != 0
            && data[r][c].l != 0
            && data[r][c - 1].t != 0
            && data[r + 1][c - 1].t != 0) {
            won_cell = true;
            points[cur_player] += 1;
            data[r][c - 1].p = cur_player;
        }
    }
    // Right
    if (c < nb_cols) {
        if (data[r][c].l != 0
            && data[r][c + 1].l != 0
            && data[r][c].t != 0
            && data[r + 1][c].t != 0) {
            won_cell = true;
            points[cur_player] += 1;
            data[r][c].p = cur_player;
        }
    }
}

msg["score"] = [points[1], points[2]];

if (!won_cell) {
    cur_player = 3 - cur_player;
    msg.nextplayer = cur_player;
}
update_board();
if (points[1] + points[2] == nb_cols * nb_rows) {
    // Game over
    var winner = 1
    if (points[2] == points[1]) {
        winner = 0;
    }
    if (points[2] > points[1]) {
        winner = 2;
    }
    cur_ended = true;
    msg.type = "end";
    msg.nextplayer = 0;
    msg.winner = winner;
}
send_to_agents(msg);
}

```

```

var field_margin = 10;
var cell_width = 40;
var cell_margin = 4;
var player_height = 40;
var width = 400;
var height = 600;
var line_width = 5;

var player_color = [
  "#E6E6E6",
  "#FC6666",
  "#0F80FF"
];

var svg = d3.select("#playing-area").append("svg")
  .attr("width", width)
  .attr("height", height)
  .append("g")
  .attr("transform", "translate("+field_margin+","+field_margin+)");

var player = svg.append("g")
  .attr("class", "player")
  .attr("transform", "translate(0,10)");

var field = svg.append("g")
  .attr("class", "field")
  .attr("transform", "translate(0,"+player_height+)");

function update_board() {
  // PLAYERS - enter & update
  var player_text = player.selectAll("text")
    .data([cur_player, cur_player]);

  player_text = player_text.enter().append("text")
    .attr("x", function(c, i) { return i * 100;})
    .merge(player_text)
    .text(function(c, i) {return "Player " + (i + 1) + ": " + points[i + 1];})
    .attr("fill", function(c, i) {
      if (c == i + 1) {
        return player_color[c];
      } else {
        return player_color[0];
      }
    });

  // ROWS - enter & update
  var rows = field.selectAll(".row")
    .data(data)
    .attr("fill", function() {return null;});

  rows.exit().remove();

  rows = rows.enter().append("g")

```

```

        .attr("class", "row")
        .attr("transform", function(row, i) {return "translate(0," + cell_width * i + ")";})
        .merge(rows);

// COLS - enter & update
var cols = rows.selectAll(".col")
    .data(function(col) {return col;});

cols.exit().remove();

var cols_enter = cols.enter().append("g")
    .attr("class", "col")
    .attr("transform", function(col, ri) {return "translate("+cell_width * ri+",0)";});

// CELL - enter
cols_enter.append("rect")
    .attr("class", "cell")
    .attr("rx", cell_margin)
    .attr("ry", cell_margin)
    .attr("opacity", 0.25)
    .attr("x", cell_margin)
    .attr("y", cell_margin)
    .attr("width", cell_width - 2*cell_margin)
    .attr("height", cell_width - 2*cell_margin);

// HLINE - enter
cols_enter.append("line")
    .attr("class", "hline")
    .attr("x1", function(cell, ci) {return cell_margin;})
    .attr("x2", function(cell, ci) {return cell_width - cell_margin;})
    .attr("y1", 0)
    .attr("y2", 0)
    .attr("stroke-linecap", "round")
    .attr("stroke", function(cell) {return player_color[cell.t];});

cols_enter.append("path")
    .attr("d", "M"+cell_margin+",0"+
        "L"+(cell_width/2)+",-"+(cell_width/3)+
        "L"+(cell_width-cell_margin)+",0"+
        "L"+(cell_width/2)+", "+(cell_width/3)+"Z")
    .attr("stroke", "black")
    .attr("stroke-width", 2)
    .attr("opacity", "0")
    .on("click", function(cell) {
        if (agents[cur_player].active == true) {
            console.log("Ignoring click, automated agent")
        } else {
            user_click(cell, "h");
        }
    });

// VLINE - enter
cols_enter.append("line")
    .attr("class", "vline")
    .attr("y1", function(cell, ci) {return cell_margin;})

```

```

.attr("y2", function(cell, ci) {return cell_width - cell_margin;})
.attr("x1", 0)
.attr("x2", 0)
.attr("stroke-linecap", "round")
.attr("stroke", function(cell) {return player_color[cell.l];});

cols_enter.append("path")
.attr("d", "M0,"+cell_margin+
    "L-"+(cell_width/3)+","+(cell_width/2)+
    "L0,"+(cell_width-cell_margin)+
    "L"+(cell_width/3)+","+(cell_width/2)+"Z")
.attr("stroke", "black")
.attr("stroke-width", 2)
.attr("opacity", "0")
.on("click", function(cell) {
    if (agents[cur_player].active == true) {
        console.log("Ignoring click, automated agent");
    } else {
        user_click(cell, "v");
    }
});

cols = cols_enter
    .merge(cols);

// HLINE - update
cols.selectAll(".hline")
.attr("stroke-width", function(cell) {
    if (typeof(cell.t) == "undefined") {
        return 0;
    }
    return line_width;
}))
.attr("stroke", function(cell) {return player_color[cell.t];});

// VLINE - update
cols.selectAll(".vline")
.attr("stroke-width", function(cell, ci) {
    if (typeof(cell.l) == "undefined") {
        return 0;
    }
    return line_width;
}))
.attr("stroke", function(cell) {return player_color[cell.l];});

// CELL - update
cols.selectAll(".cell")
.attr("fill", function(cell) {
    if (cell.p == undefined) {
        return "white";
    }
    return player_color[cell.p];
}));
}

```

```
// AGENT CONNECTIONS
```

```
var agents = [  
  {},  
  {address: undefined, active: false, socket: undefined},  
  {address: undefined, active: false, socket: undefined}  
];
```

```
var msg_queue = [];
```

```
function start_connections() {  
  for (var i=1; i<3; i++) {  
    agents[i] = {address:undefined, active: false, socket: undefined};  
    var address = document.getElementById('agent'+i).value;  
    console.log('Address agent'+i+": "+address);  
    if (address != "") {  
      //console.log("Starting websocket for agent "+i+" on address "+address);  
      var agent = agents[i];  
      agent.address = address;  
      agent.socket = new WebSocket(address);  
      agent.socket.onopen = (function (ii, iagent) { return function(event) {  
        console.log("Agent "+ii+" connected")  
        iagent.active = true;  
        iagent.socket.onmessage = function(event) {  
          var msg = JSON.parse(event.data);  
          //console.log("Get msg from agent "+ii, msg);  
          if (msg.type == "action") {  
            if (cur_player == ii) {  
              console.log("Received action from ACTIVE player "+ii, msg);  
              user_click(data[msg.location[0]][msg.location[1]], msg.orientation);  
            } else {  
              console.log("Received action from NON-ACTIVE player "+ii, msg);  
            }  
          }  
        }  
      }  
      return false;  
    };  
    iagent.socket.onclose = function(event) {  
      console.log("Closing connection to agent "+ii);  
    };  
    iagent.socket.onerror = function(event) {  
      console.log("Error on connection to agent "+ii, event);  
    };  
    msg = {  
      "type": "start",  
      "player": ii,  
      "timelimit": timelimit,  
      "game": cur_game,  
      "grid": [nb_rows, nb_cols]  
    };  
    iagent.socket.send(JSON.stringify(msg));  
  }  
  }(i, agent));  
}
```

```

}

function send_to_agents(msg) {
  msg_queue.push(JSON.stringify(msg));
  try_sending_to_agents();
}

function try_sending_to_agents() {
  var all_connected = true;
  for (var i=1; i<3; i++) {
    if (agents[i].address !== undefined && agents[i].active == false) {
      all_connected = false;
      break;
    }
  }
  if (!all_connected) {
    // Wait until all are connected
    setTimeout(try_sending_to_agents, 100);
  } else {
    if (msg_queue.length == 0 ) {
      return;
    }
    var msg = msg_queue.shift();
    console.log("Send msg to agents", msg);
    for (var i=1; i<3; i++) {
      if (agents[i].active == true) {
        agents[i].socket.send(msg);
      }
    }
  }
}

```

// STARTUP

```

function restart() {
  restart_game();
  update_board();
  start_connections();
}

var restartbtn = document.getElementById("restart-btn");
restartbtn.onclick = function() {
  console.log("Restart game");
  restart();
};

restart();

```