# **INFSCI 2591: Algorithm Design Project 5**

Guanxiong Ding 4/25/2013

# Problem 1 Three merge sorting

Pseudo-code for P1

function merge\_sort(list) {

if list.length <= 1

return list

}

var left = [], mid = [], right = []

// divide into three

var div1 = int(list.length / 3), div2 = int(list.length\*2 / 3)

foreach list from first to div1

left.push(list[i])

foreach list from div1 to div2

mid.push(list[i])

foreach list from div2 to last

right.push(list[i])

left = merge\_sort(left)

mid = merge\_sort(mid)

right = merge\_sort(right)

return merge(left, mid, right)

}

function merge(left, mid, right) {

var result = []

while(left.length > 0 || mid.length > 0 || right.length > 0) {

result.push(min(left.first(), mid.first(), right.first()))

remove min(left.first(), mid.first(), right.first()) from its list

}

return result

}

Source Code for P1 (JavaScript)

var numbers = [];

function getRandomInt (min, max) {

return Math.floor(Math.random() \* (max - min + 1)) + min;

}

function j(v) {

return JSON.stringify(v);

}

function init() {

var frameOld = $('.frame-old').html('');

var frame = $('.frame').html('');

var numItem = getRandomInt(3,20);

var ranTmp = 0;

for (var i = 0; i < numItem; i++) {

ranTmp = getRandomInt(0,20);

frame.append("<span id="+i+">"+ranTmp+"</span>");

frameOld.append("<span id="+i+">"+ranTmp+"</span>");

numbers.push({i: i, num:ranTmp});

};

var items = $('.frame span');

for (var i = 0 - 1; i < items.length; i++) {

$(items[i]).css({'left': i\*90});

};

var items = $('.frame-old span');

for (var i = 0 - 1; i < items.length; i++) {

$(items[i]).css({'left': i\*90});

};

}

var items = $('.frame span');

function merge\_sort(list) {

if (list.length <= 1) {

return list;

}

var left = [], mid = [], right = [];

// caculate the divider

var div1 = parseInt(list.length / 3), div2 = parseInt(list.length\*2 / 3);

console.log(div1 +': '+ div2);

for (var i=0; i<div1; i++) {

left.push(list[i]);

}

for (var i=div1; i<div2; i++) {

mid.push(list[i]);

}

for (var i=div2; i<list.length; i++) {

right.push(list[i]);

}

left = merge\_sort(left);

mid = merge\_sort(mid);

right = merge\_sort(right);

return merge(left, mid, right);

}

var merge = function(left, mid, right) {

var result = [];

while(left.length > 0 || mid.length > 0 || right.length > 0) {

if (left.length > 0 && mid.length > 0 && right.length > 0) {

if (left[0].num <= right[0].num && left[0].num <= mid[0].num) {

result.push(left[0]);

left.shift();

}else if (mid[0].num <= left[0].num && mid[0].num <= right[0].num) {

result.push(mid[0]);

mid.shift();

}else{

result.push(right[0]);

right.shift();

}

}else if (left.length > 0 && mid.length > 0) {

if (left[0].num <= mid[0].num) {

result.push(left[0]);

left.shift();

}else{

result.push(mid[0]);

mid.shift();

}

}else if (mid.length > 0 && right.length > 0) {

if (right[0].num <= mid[0].num) {

result.push(right[0]);

right.shift();

}else{

result.push(mid[0]);

mid.shift();

}

}else if (right.length > 0 && left.length > 0) {

if (right[0].num <= left[0].num) {

result.push(right[0]);

right.shift();

}else{

result.push(left[0]);

left.shift();

}

}else if(left.length > 0) {

result.push(left[0]);

left.shift();

}else if(mid.length > 0) {

result.push(mid[0]);

mid.shift();

}else if(right.length > 0) {

result.push(right[0]);

right.shift();

}

}

console.log(':'+j(result));

var baseLeft = $('.frame span[id='+result[0].i+']').offset().left;

if(result.length == $('.frame span').length) {

baseLeft = 0;

}

for (var i = 0; i < result.length; i++) {

$('.frame span[id='+result[i].i+']').animate({'left': baseLeft + i\*90}, 200);

};

return result;

}

init();

merge\_sort(numbers);

Big-O analysis for P1

1. O(n log3n)
2. First running is T(n)
3. Next running is T(n) = 3T(n/3) + n
4. The result will be n + 3T(n/3) + 9T(n/9) + … + 3k(N/3k) + … +n/3(3) = nlog3n

Time plot for different n for P1

Results of testing for P1



# Problem 2 Euler Path

Pseudo-code for P2

nodes = a ajacency matrix

path = []

// recursively travel all nodes

function EulerTrail(u) {

foreach neighbor v of u{

remove edge[u][v]

EulerTrail(v)

}

// put traveled node to path

path[count++] = u

}

count = 0

EulerTrail(start)

if path contain all nodes

connectivity = true

else

connectivity = false

// check only two odd

foreach neighbor of each nodes

if neighbor.lenght is odd

oddNum++

if oddNum == 2

onlyTwoOdd = true

else

onlyTwoOdd = false

if onlyTwoOdd and connectivity

graph exist Euler Path

Source Code for P2 (JavaScript)

var LEN = 15;

var nodes = [];

var tmp = [];

for(var i=0; i<LEN; i++){

tmp = [];

for(var j=0; j<LEN; j++){

if(i == j) {

tmp.push(0);

}else if(j>i){

tmp.push(getRandomInt(0,1))

}else{

tmp.push(nodes[j][i]);

}

}

nodes.push(tmp);

$('.frame-ori').append(tmp+'<br>');

}

console.log(nodes);

// print current nodes

function printNodes() {

$('.frame').html('');

for(var i=0; i<LEN; i++){

for(var j=0; j<LEN; j++){

$('.frame').append(nodes[i][j]+',');

}

$('.frame').append('<br>');

}

}

// generate random number

function getRandomInt (min, max) {

return Math.floor(Math.random() \* (max - min + 1)) + min;

}

var circuit = [];

// start construction Euler path

var makeEuler = function () {

circuPos = 0;

EulerTrail(0);

}

// recursively travel all nodes

var EulerTrail = function(u) {

for(var i = 0; i<nodes[u].length; i++){

if(nodes[u][i] == 1){

nodes[u][i] = 0;

nodes[i][u] = 0;

EulerTrail(i);

}

}

// put traveled node to path

circuit[circuPos++] = u;

}

// check connection number of a nodes

function checkConnNum(array) {

var num = 0;

for (var i = array.length - 1; i >= 0; i--) {

if(array[i] == '1'){

num++;

}

}

return num;

}

var ifOnlyTwoOdd = true, oddNum = 0;

// check odd node only two

for(var i = 0; i < LEN; i++) {

var connNum = checkConnNum(nodes[i]);

if(connNum != parseInt(connNum/2)\*2){

oddNum++;

}

if(oddNum > 2){

ifOnlyTwoOdd = false;

break;

}

}

if(oddNum < 2) {

ifOnlyTwoOdd = false;

}

makeEuler();

console.log(circuit);

var ifConnected = true;

// check array contain item

function arrayContain(array, item) {

for (var i = array.length - 1; i >= 0; i--) {

if(array[i] == item){

return true;

}

}

return false;

}

// check if connected graph by check tantative Euler path contian all nodes

for(var i=0; i<LEN; i++){

if (!arrayContain(circuit, i)) {

ifConnected = false;

}

}

console.log('ifOnlyTwoOdd: '+ifOnlyTwoOdd);

console.log('ifConnected: '+ifConnected);

if(ifOnlyTwoOdd && ifConnected) {

$('.result').append('<h2>Euler Path Found!</h2>');

$('.result').append('<p>'+circuit+'</p>');

}else{

$('.result').append('<h2>Eluer Path Not Exist...</h2>');

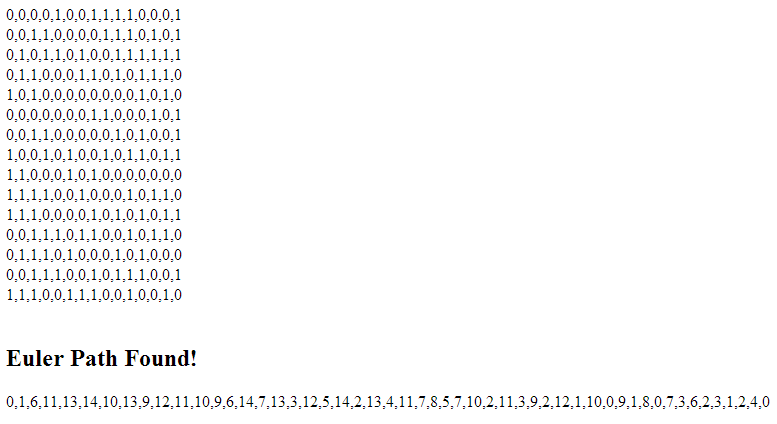
}

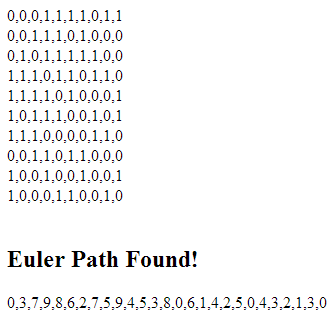
Big-O analysis for P2

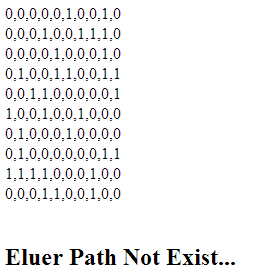
1. O(e2)
2. While the graph traversal in Fleury's algorithm is linear in the number of edges, i.e. O(|E|), we also need to factor in the complexity of detecting bridges. If we are to re-run Tarjan's linear time bridge-finding algorithm after the removal of every edge, Fleury's algorithm will have a time complexity of O(|E|2).

Time plot for different n for P2

Results of testing for P2







# Problem 3 Three-dimensional tictactoe

Pseudo-code for P3

nodes = a ajacency matrix

path = []

// recursively travel all nodes

function EulerTrail(u) {

foreach neighbor v of u{

remove edge[u][v]

EulerTrail(v)

}

// put traveled node to path

path[count++] = u

}

count = 0

EulerTrail(start)

if path contain all nodes

connectivity = true

else

connectivity = false

// check only two odd

foreach neighbor of each nodes

if neighbor.lenght is odd

oddNum++

if oddNum == 2

onlyTwoOdd = true

else

onlyTwoOdd = false

if onlyTwoOdd and connectivity

graph exist Euler Path

// q3

board = 4 x 4 x 4 matrix

lastHand = {

p: 2, // player of lastest hand

x: 1, // position of the hand

y: 3,

z: 0

}

function checkWin(lastHand){

// init win on each row to true

ifRowOnX = true

ifRowOnY = true

ifRowOnZ = true

ifRowOnXY = true

ifRowOnXZ = true

ifRowOnYZ = true

ifRowOnXYZ = true

foreach node in board

if node on X axis of lastHand and node != lastHand.p

ifRowOnX = false

if node on Y axis of lastHand and node != lastHand.p

ifRowOnY = false

if node on Z axis of lastHand and node != lastHand.p

ifRowOnZ = false

if node on XY axis of lastHand and node != lastHand.p

ifRowOnXY = false

if node on XZ axis of lastHand and node != lastHand.p

ifRowOnXZ = false

if node on YZ axis of lastHand and node != lastHand.p

ifRowOnYZ = false

if node on XYZ axis of lastHand and node != lastHand.p

ifRowOnXYZ = false

// if one of direction in a row, this hand win

if ifRowOnX = true or ifRowOnY = true or ifRowOnZ = true or ifRowOnXY = true or ifRowOnXZ = true or ifRowOnYZ = true or ifRowOnXYZ = true

lastHand.p win

}

Source Code for P3 (JavaScript)

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Problem 5</title>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js"></script>

<script src="three.min.js"></script>

<script src="stats.min.js"></script>

<style type="text/css">

body {

-webkit-touch-callout: none;

-webkit-user-select: none;

-khtml-user-select: none;

-moz-user-select: none;

-ms-user-select: none;

user-select: none;

font-family: "Lucida Grande","Lucida Sans Unicode","DejaVu Sans",Lucida,Arial,Helvetica,sans-serif;

margin: 0;

}

h1{

font-size: 20px;

margin: 0;

padding: 20px 0;

}

div#canvas-frame{

border: none;

cursor: default;

width: 800px;

height: 800px;

background-color: #EEEEEE;

float: left;

border: 1px solid #ccc;

}

.info{

margin-left: 850px;

}

.who-win.red {

color: #ff0000;

}

.who-win.green {

color: #00ff00;

}

.cp {

clear: both;

font-size: 12px;

}

.cp a {

color: #00f;

}

</style>

</head>

<body onload="threeStart();">

<div id="canvas-frame"></div>

<div class="info">

<div class='btn'>

<h1>Algorithm Project 5 Problem 3<br>Three Dimentioanl Tictactoe</h1>

<button class="restart">Restart</button>

<h2 class="who-win"></h2>

</div>

<div class='board-2d'></div>

<div class='last'></div>

<div class='win'></div>

</div>

<div class="cp">4/24/2013 Designed by Neil Ding, Powered by <a href="http://threejs.org/">Three.js</a></div>

<script>

// first hand use 1, second hand use 2.

var l = function(val){

console.log(val);

};

var board = [[[0,0,0,0],[0,0,0,0],[0,0,0,0],[0,0,0,0]],

[[0,0,0,0],[0,0,0,0],[0,0,0,0],[0,0,0,0]],

[[0,0,0,0],[0,0,0,0],[0,0,0,0],[0,0,0,0]],

[[0,0,0,0],[0,0,0,0],[0,0,0,0],[0,0,0,0]]];

var printBoard = function() {

var bdDom = $('.board-2d').html('');

var tmp = '';

for(var i = 0; i < 4; i++){

for(var j = 0; j < 4; j++){

for(var k = 0; k < 4; k++) {

tmp += board[i][j][k]+' ';

}

tmp += "<br>";

}

tmp += "<br>";

}

bdDom.html(tmp);

};

// hand type include player, position

var lastHand = {

p: 2,

x: 0,

y: 0,

z: 0

};

// checkWinner check one hand if win

// return ifWin:boolean, p:player, d:winRowPostion

var checkWinner = function(hand){

var checkLineX = function(point){

return point[0] == hand.z && point[1] == hand.y ? true : false;

};

var checkLineY = function(point){

return point[0] == hand.z && point[2] == hand.x ? true : false;

};

var checkLineZ = function(point){

return point[1] == hand.y && point[2] == hand.x ? true : false;

};

var checkLineXY = function(point){

if(hand.x == hand.y){

return point[0] == hand.z && point[1] == point[2] ? true : false;

}else if(hand.x == 3-hand.y){

return point[0] == hand.z && point[1] == 3-point[2] ? true : false;

}else{

return true;

}

};

var checkLineXZ = function(point){

if(hand.x == hand.z){

return point[1] == hand.y && point[0] == point[2] ? true : false;

}else if(hand.x == 3 - hand.z){

return point[1] == hand.y && point[0] == 3-point[2] ? true : false;

}else{

return true;

}

};

var checkLineYZ = function(point){

if(hand.y == hand.z){

return point[2] == hand.x && point[0] == point[1] ? true : false;

}else if(hand.x == 3 - hand.z){

return point[2] == hand.x && point[0] == 3-point[1] ? true : false;

}else{

return true;

}

};

var checkLineXYZ = function(point){

if(hand.x == hand.y && hand.y == hand.z){

return point[0] == point[1] && point[1] == point[2] ? true : false;

}else if(hand.x == hand.y && hand.y == 3-hand.z){

return point[0] == 3-point[1] && point[1] == point[2] ? true : false;

}else if(hand.x == 3-hand.y && hand.y == hand.z){

return point[0] == point[1] && point[1] == 3-point[2] ? true : false;

}else if(hand.x == 3-hand.y && hand.y == 3-hand.z){

return point[0] == 3-point[1] && point[1] == 3-point[2] ? true : false;

}else {

return true;

}

};

// x, y, z, xyCross, yzCross, xzCross, zyxCross

var winPosi = {

x: true,

y: true,

z: true,

xyCross: true,

xzCross: true,

yzCross: true,

zyxCross: true

};

var point = [];

for(var i = 0; i < 4; i++){

for(var j = 0; j < 4; j++){

for(var k = 0; k < 4; k++) {

point = [i, j, k];

if(checkLineX(point) && board[i][j][k] != hand.p){

winPosi.x = false;

}

if(checkLineY(point) && board[i][j][k] != hand.p){

winPosi.y = false;

}

if(checkLineZ(point) && board[i][j][k] != hand.p){

winPosi.z = false;

}

if(checkLineXY(point) && board[i][j][k] != hand.p){

winPosi.xyCross = false;

}

if(checkLineXZ(point) && board[i][j][k] != hand.p){

winPosi.xzCross = false;

}

if(checkLineYZ(point) && board[i][j][k] != hand.p){

winPosi.yzCross = false;

}

if(checkLineXYZ(point) && board[i][j][k] != hand.p){

winPosi.zyxCross = false;

}

}

}

}

if(winPosi.x || winPosi.y || winPosi.z || winPosi.xyCross || winPosi.xzCross || winPosi.yzCross || winPosi.zyxCross ) {

$('.who-win').text('Player ' + lastHand.p + 'Win');

$('.who-win').removeClass('red').removeClass('green').addClass(lastHand.p == 1 ? 'red' : 'green');

}

$('.last').html("My Last Hand:<br>"+JSON.stringify(hand)+"<br><br>Machine Last Hand:<br>"+JSON.stringify(lastHand));

$('.win').html("<br>Win Status:<br>"+JSON.stringify(winPosi));

}

function getRandomInt (min, max) {

return Math.floor(Math.random() \* (max - min + 1)) + min;

}

var aiHand = {

p: 2,

x: 2,

y: 1,

z: 3

};

var randomAi = function() {

aiHand.x = getRandomInt(0, 3);

aiHand.y = getRandomInt(0, 3);

aiHand.z = getRandomInt(0, 3);

}

var aiGo = function(aiHnad){

lastHand.p = aiHand.p;

lastHand.x = aiHand.x;

lastHand.y = aiHand.y;

lastHand.z = aiHand.z;

board[lastHand.z][lastHand.y][lastHand.x] = lastHand.p;

cube[lastHand.z][lastHand.y][lastHand.x].material.color.setHex( 0x00ff00 );

cube[lastHand.z][lastHand.y][lastHand.x].material.ambient.setHex( 0x00ff00 );

cube[lastHand.z][lastHand.y][lastHand.x].ifDown = true;

checkWinner(lastHand);

};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\*

\* Interface

\*

\*

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

var width, height;

var renderer, projector;

function initThree() {

width = document.getElementById('canvas-frame').clientWidth;

height = document.getElementById('canvas-frame').clientHeight;

renderer = new THREE.WebGLRenderer({antialias: true});

renderer.setSize(width, height );

document.getElementById('canvas-frame').appendChild(renderer.domElement);

renderer.setClearColor(0xFFFFFF, 1.0);

renderer.shadowMapEnabled = true;

projector = new THREE.Projector();

// init stats

stats = new Stats();

stats.domElement.style.position = 'absolute';

stats.domElement.style.top = '0px';

$('body').append( stats.domElement );

}

var camera;

function initCamera() {

camera = new THREE.PerspectiveCamera( 65 , width / height , 1 , 10000 );

camera.position.x = 500;

camera.position.y = -400;

camera.position.z = 300;

camera.up.x = 0;

camera.up.y = 0;

camera.up.z = 1;

camera.lookAt( {x:0, y:0, z:0 } );

}

var scene;

function initScene() {

scene = new THREE.Scene();

}

var light, light2;

function initLight() {

light = new THREE.DirectionalLight(0xffffff, 1.0, 0);

light.position.set( 50, 50, 200 );

scene.add(light);

light2 = new THREE.AmbientLight(0x555555);

scene.add(light2);

light.castShadow = true;

}

var cube = [[[],[],[],[]],[[],[],[],[]],[[],[],[],[]],[[],[],[],[]]];

var objects = [];

var plane;

function initObject(){

for(var i=0; i<4; i++){

for(var j=0; j<4; j++){

for(var k=0; k<4; k++){

cube[i][j][k] = new THREE.Mesh(

new THREE.CubeGeometry(40,40,40),

new THREE.MeshLambertMaterial({color: 0xAAAAAA, ambient:0xAAAAAA})

);

scene.add(cube[i][j][k]);

cube[i][j][k].position.set(-100+100\*k,-100+100\*i,-100+100\*j);

cube[i][j][k].castShadow = true;

cube[i][j][k].ifDown = false;

cube[i][j][k].x = k;

cube[i][j][k].y = j;

cube[i][j][k].z = i;

objects.push(cube[i][j][k]);

}

}

}

l(cube);

plane = new THREE.Mesh(

new THREE.PlaneGeometry(600, 600),

new THREE.MeshLambertMaterial({color: 0xcccccc, ambient:0x00FF00})

);

scene.add(plane);

plane.position.set(0,20,-200);

plane.receiveShadow = true;

}

var t=0;

function loop() {

t++;

// cube[0].rotation.set( t/100, 0, 0 );

// cube[1].rotation.set( 0, t/100, 0 );

// cube[2].rotation.set( 0, 0, t/100 );

// cube[3].rotation.set( t/100, 0, 0 );

// cube[4].rotation.set( 0, t/100, 0 );

// cube[5].rotation.set( 0, 0, t/100 );

renderer.clear();

//camera.position.set( 400\*Math.cos(t/100), 400\*Math.sin(t/200), 50\*Math.cos(t/50));

camera.lookAt( {x:0, y:0, z:0 } );

renderer.render(scene, camera);

window.requestAnimationFrame(loop);

}

var down = false;

var sy = 0, sz = 0;

var cx = 0;

var cy = 0;

window.onmousedown = function (ev){

ev.preventDefault();

if (ev.target == renderer.domElement) {

down = true;

// for mousemove

sy = ev.clientX; sz = ev.clientY;

// cx = cube[0].position.x;

// cy = cube[0].position.y;

var vector = new THREE.Vector3( ( ev.clientX / 800 ) \* 2 - 1, - ( ev.clientY / 800 ) \* 2 + 1, 0.5 );

projector.unprojectVector( vector, camera );

var raycaster = new THREE.Raycaster( camera.position, vector.sub( camera.position ).normalize() );

var intersects = raycaster.intersectObjects( objects );

if ( intersects.length > 0 ) {

if(!intersects[ 0 ].object.ifDown){

l(intersects[ 0 ].object.z+','+intersects[ 0 ].object.y+','+intersects[ 0 ].object.x);

if(lastHand.p === 2){

intersects[ 0 ].object.material.color.setHex( 0xff0000 );

intersects[ 0 ].object.material.ambient.setHex( 0xff0000 );

board[intersects[ 0 ].object.z][intersects[ 0 ].object.y][intersects[ 0 ].object.x] = 1;

lastHand = {

p: 1,

x: intersects[ 0 ].object.x,

y: intersects[ 0 ].object.y,

z: intersects[ 0 ].object.z

};

}else{

intersects[ 0 ].object.material.color.setHex( 0x00ff00 );

intersects[ 0 ].object.material.ambient.setHex( 0x00ff00 );

board[intersects[ 0 ].object.z][intersects[ 0 ].object.y][intersects[ 0 ].object.x] = 2;

lastHand = {

p: 2,

x: intersects[ 0 ].object.x,

y: intersects[ 0 ].object.y,

z: intersects[ 0 ].object.z

};

}

checkWinner(lastHand);

printBoard();

intersects[ 0 ].object.ifDown = true;

while(1){

randomAi();

if(cube[aiHand.z][aiHand.y][aiHand.x].ifDown != true) {

l(cube[aiHand.z][aiHand.y][aiHand.x]);

aiGo(aiHand);

break;

}

}

}

}

}

};

window.onmouseup = function(){ //¥Þ¥¦¥¹¥¢¥Ã¥×

down = false;

};

window.onmousemove = function(ev) { //¥Þ¥¦¥¹¥à©`¥Ö

ev.preventDefault();

var speed = 2;

if (down) {

stats.update();

if (ev.target == renderer.domElement) {

var dy = -(ev.clientX - sy);

var dz = -(ev.clientY - sz);

//console.log(dy+','+dz);

//console.log(cx+','+cy);

camera.position.y += dy\*speed;

camera.position.z -= dz\*speed;

//cube[0].position.set(-dz+cx,-dy+cy,0);

sy -= dy;

sz -= dz;

//renderer.clear();

camera.lookAt( {x:0, y:0, z:0 } );

renderer.render(scene, camera);

}

}else{

var vector = new THREE.Vector3( ( ev.clientX / 800 ) \* 2 - 1, - ( ev.clientY / 800 ) \* 2 + 1, 0.5 );

projector.unprojectVector( vector, camera );

var raycaster = new THREE.Raycaster( camera.position, vector.sub( camera.position ).normalize() );

var intersects = raycaster.intersectObjects( objects );

var frame = document.getElementById('canvas-frame');

for (var i = objects.length - 1; i >= 0; i--) {

if(!objects[i].ifDown) {

objects[i].material.color.setHex( 0xAAAAAA );

objects[i].material.ambient.setHex( 0xAAAAAA );

}

};

if ( intersects.length > 0 ) {

if(!intersects[ 0 ].object.ifDown) {

if(lastHand.p === 2){

intersects[ 0 ].object.material.color.setHex( 0xff0000 );

intersects[ 0 ].object.material.ambient.setHex( 0xff0000 );

}else{

intersects[ 0 ].object.material.color.setHex( 0x00ff00 );

intersects[ 0 ].object.material.ambient.setHex( 0x00ff00 );

}

}

frame.style.cursor = 'pointer';

}else{

frame.style.cursor = 'move';

}

}

// change cursor

}

window.onmousewheel = function(ev){//¥Þ¥¦¥¹¥Û¥¤©`¥ë

var speed = 0.2;

camera.position.y += ev.wheelDelta \* speed ;

}

function threeStart() {

initThree();

initCamera();

initScene();

initLight();

initObject();

loop();

}

$('.restart').click(function(){

location.reload();;

});

</script>

</body>

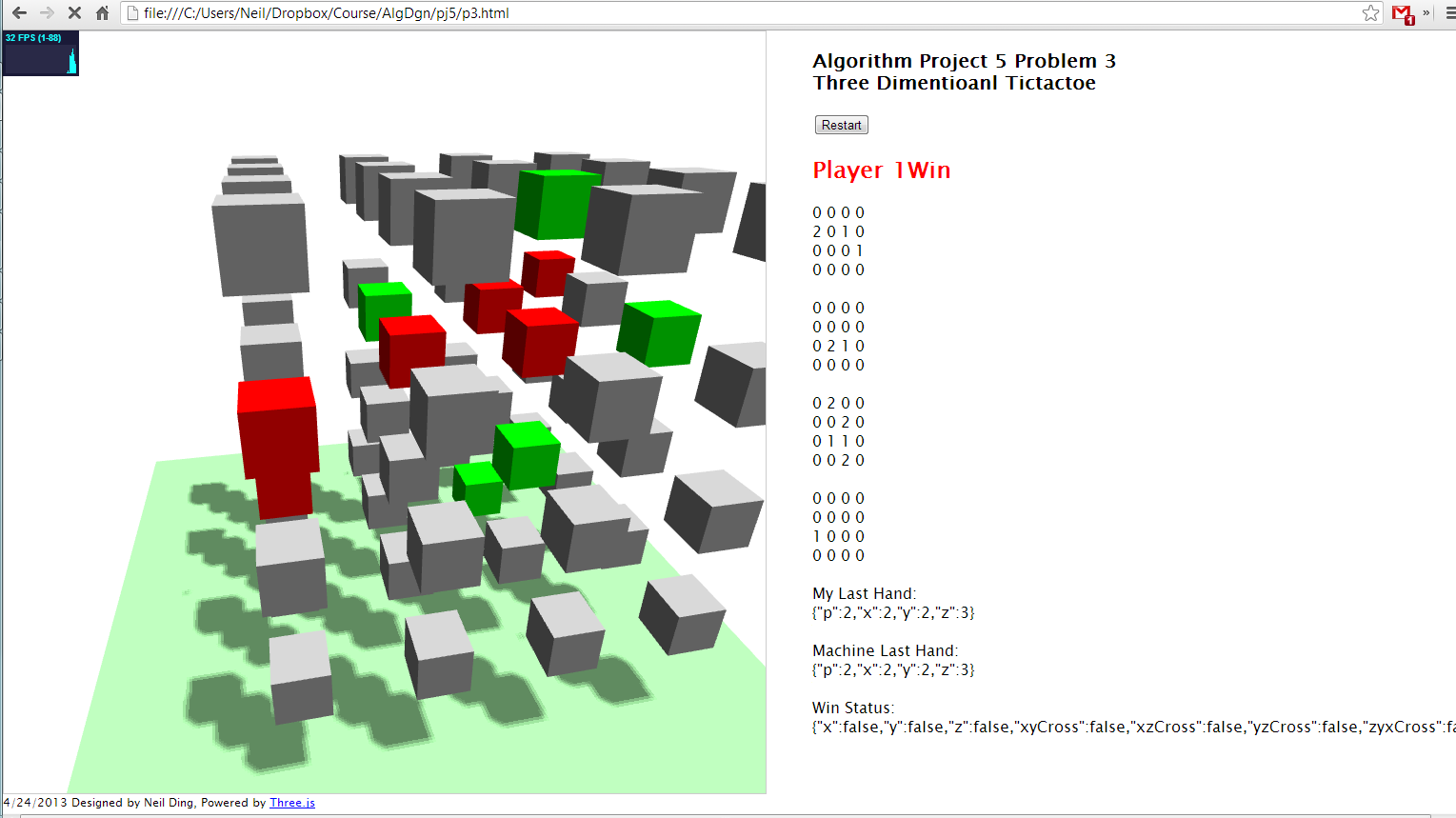
</html>

Big-O analysis for P3

1. O(64)
2. The machine make move randomly, so the time consume is spend on checking if win of a hand, which loop over each node, namely 64.

Time plot for different n for P3

Results of testing for P3



# Problem 4 Tournament Scheduling

Pseudo-code for P4 a

// fix the first item, rotate other clock-wise

function rotate(arr){

var tmpArr1 =[], tmpArr2 = []

for(var i = 0 i < arr[0].length i++){

if i == 0

tmpArr1[i] = arr[0][i]

tmpArr2[i] = arr[1][i+1]

// set exception for len 2

else if i == 1 && arr[0].length <= 2

tmpArr1[i] = arr[1][i-1]

tmpArr2[i] = arr[0][i]

else if i == 1

tmpArr1[i] = arr[1][i-1]

tmpArr2[i] = arr[1][i+1]

else if i == arr[0].length - 1

tmpArr1[i] = arr[0][i-1]

tmpArr2[i] = arr[0][i]

else

tmpArr1[i] = arr[0][i-1]

tmpArr2[i] = arr[1][i+1]

}

return [tmpArr1, tmpArr2]

}

// Schedule A

function scheduleA(N) {

players = pow(2,N) players array

n = players.length

row1 = first half of player array

row2 = second half of players array

for i from 0 to n-1

for j from 0 to n

if row1 contain j

day[i].push(players[row2.indexOf(j)])

else if row2 contain j

day[i].push(players[row1.indexOf(j)])

tmpArr = rotate([row1, row2]);

row1 = tmpArr[0];

row2 = tmpArr[1];

return day;

}

Pseudo-code for P4 b

// fix the first item, rotate other clock-wise

function rotate(arr){

var tmpArr1 =[], tmpArr2 = []

for(var i = 0 i < arr[0].length i++){

if i == 0

tmpArr1[i] = arr[0][i]

tmpArr2[i] = arr[1][i+1]

// set exception for len 2

else if i == 1 && arr[0].length <= 2

tmpArr1[i] = arr[1][i-1]

tmpArr2[i] = arr[0][i]

else if i == 1

tmpArr1[i] = arr[1][i-1]

tmpArr2[i] = arr[1][i+1]

else if i == arr[0].length - 1

tmpArr1[i] = arr[0][i-1]

tmpArr2[i] = arr[0][i]

else

tmpArr1[i] = arr[0][i-1]

tmpArr2[i] = arr[1][i+1]

}

return [tmpArr1, tmpArr2]

}

// Schedule B

function scheduleB(N) {

players = N players array

n = players.length

row1 = first half of player array

row2 = second half of players array

if N is odd

row.push(n)

players.push('-');

for i from 0 to n-1

for j from 0 to n (if N is odd to n-1)

if row1 contain j

day[i].push(players[row2.indexOf(j)])

else if row2 contain j

day[i].push(players[row1.indexOf(j)])

tmpArr = rotate([row1, row2]);

row1 = tmpArr[0];

row2 = tmpArr[1];

return day;

}

Source Code for P4 a (JavaScript)

var rotate = function(arr){

var tmpArr1 =[], tmpArr2 = [];

for(var i = 0; i < arr[0].length; i++){

if (i == 0){

tmpArr1[i] = arr[0][i];

tmpArr2[i] = arr[1][i+1];

// set exception for len 2

}else if (i == 1 && arr[0].length <= 2){

tmpArr1[i] = arr[1][i-1];

tmpArr2[i] = arr[0][i];

}else if (i == 1){

tmpArr1[i] = arr[1][i-1];

tmpArr2[i] = arr[1][i+1];

}else if (i == arr[0].length - 1) {

tmpArr1[i] = arr[0][i-1];

tmpArr2[i] = arr[0][i];

}else{

tmpArr1[i] = arr[0][i-1];

tmpArr2[i] = arr[1][i+1];

}

}

return [tmpArr1, tmpArr2];

};

var getIndexOfArr = function(arr, item){

for(var i = 0; i < arr.length; i++){

if (item == arr[i]){

return i;

}

}

return -1;

}

var q1 = function(power){

var power = power;

var players = [], row1 = [], row2 = [];

for(var i = 0; i < Math.pow(2, power); i++){

if (i < Math.pow(2, power) / 2) {

row1.push(i);

}else{

row2.push(i);

}

players.push('p'+(i+1));

}

var thead = $('<tr class="thead"><td>DAY</td></tr>'), tbody = $('<tbody></tbody>');

for (var i = 0; i < players.length; i++) {

thead.append("<td>"+players[i]+"</td>");

}

for (var i = 0; i < players.length-1; i++) {

var tmp = $('<tr><td class="day">day'+(i+1)+'</td></tr>');

for (var j = 0; j < players.length; j++) {

if(getIndexOfArr(row1, j) != -1){

tmp.append("<td>"+players[row2[getIndexOfArr(row1, j)]]+"</td>");

}else if(getIndexOfArr(row2, j) != -1){

tmp.append("<td>"+players[row1[getIndexOfArr(row2, j)]]+"</td>");

}

}

// do rotation

tmpArr = rotate([row1, row2]);

row1 = tmpArr[0];

row2 = tmpArr[1];

tbody.append(tmp);

}

// load to HTML

thead.appendTo($('.t1'));

tbody.appendTo($('.t1'));

}

q1(3);

Source Code for P4 b (JavaScript)

var rotate = function(arr){

var tmpArr1 =[], tmpArr2 = [];

for(var i = 0; i < arr[0].length; i++){

if (i == 0){

tmpArr1[i] = arr[0][i];

tmpArr2[i] = arr[1][i+1];

// set exception for len 2

}else if (i == 1 && arr[0].length <= 2){

tmpArr1[i] = arr[1][i-1];

tmpArr2[i] = arr[0][i];

}else if (i == 1){

tmpArr1[i] = arr[1][i-1];

tmpArr2[i] = arr[1][i+1];

}else if (i == arr[0].length - 1) {

tmpArr1[i] = arr[0][i-1];

tmpArr2[i] = arr[0][i];

}else{

tmpArr1[i] = arr[0][i-1];

tmpArr2[i] = arr[1][i+1];

}

}

return [tmpArr1, tmpArr2];

};

var getIndexOfArr = function(arr, item){

for(var i = 0; i < arr.length; i++){

if (item == arr[i]){

return i;

}

}

return -1;

}

// Q2 - 1

var q21 = function(number) {

var power = power;

var players = [], row1 = [], row2 = [];

for(var i = 0; i < number; i++){

if (i < number / 2) {

row1.push(i);

}else{

row2.push(i);

}

players.push('p'+(i+1));

}

row2.push(number);

players.push('-');

var thead = $('<tr class="thead"><td>DAY</td></tr>'), tbody = $('<tbody></tbody>');

for (var i = 0; i < players.length-1; i++) {

thead.append("<td>"+players[i]+"</td>");

}

for (var i = 0; i < players.length-1; i++) {

var tmp = $('<tr><td class="day">day'+(i+1)+'</td></tr>');

for (var j = 0; j < players.length-1; j++) {

if(getIndexOfArr(row1, j) != -1){

tmp.append("<td>"+players[row2[getIndexOfArr(row1, j)]]+"</td>");

}else if(getIndexOfArr(row2, j) != -1){

tmp.append("<td>"+players[row1[getIndexOfArr(row2, j)]]+"</td>");

}

}

// do rotation

tmpArr = rotate([row1, row2]);

row1 = tmpArr[0];

row2 = tmpArr[1];

tbody.append(tmp);

}

thead.appendTo($('.t2-1'));

tbody.appendTo($('.t2-1'));

}

var q22 = function(number) {

var power = power;

var players = [], row1 = [], row2 = [];

for(var i = 0; i < number; i++){

if (i < number / 2) {

row1.push(i);

}else{

row2.push(i);

}

players.push('p'+(i+1));

}

var thead = $('<tr class="thead"><td>DAY</td></tr>'), tbody = $('<tbody></tbody>');

for (var i = 0; i < players.length; i++) {

thead.append("<td>"+players[i]+"</td>");

}

for (var i = 0; i < players.length-1; i++) {

var tmp = $('<tr><td class="day">day'+(i+1)+'</td></tr>');

for (var j = 0; j < players.length; j++) {

if(getIndexOfArr(row1, j) != -1){

tmp.append("<td>"+players[row2[getIndexOfArr(row1, j)]]+"</td>");

}else if(getIndexOfArr(row2, j) != -1){

tmp.append("<td>"+players[row1[getIndexOfArr(row2, j)]]+"</td>");

}

}

// do rotation

tmpArr = rotate([row1, row2]);

row1 = tmpArr[0];

row2 = tmpArr[1];

tbody.append(tmp);

}

thead.appendTo($('.t2-2'));

tbody.appendTo($('.t2-2'));

}

q21(9);

q22(10);

Big-O analysis for P4 a

1. O(23n)
2. Number of player will be 2n, and there will be double nested loop over players and a function to find item’s index of array, which loop n/2 times. So big o will be 2n \*2n \*2n = 23n

Big-O analysis for P4 b

1. O(n3)
2. There is a double nested for loop with n length. In the inner loop, there is function to find item’s index of array. It will loop n/2 times. So the big o will be n\*n\*n = n3

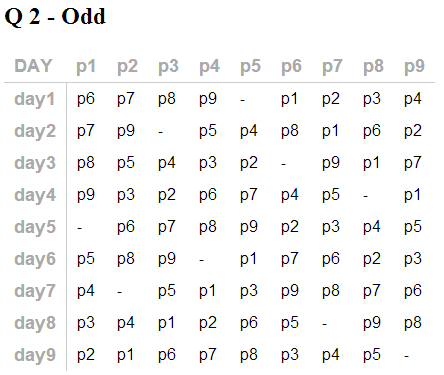
Time plot for different n for P4 a

Time plot for different n for P4 b

Results of testing for P4 a

# 

Results of testing for P4 B Odd



Results of testing for P4 B Even

