Universidade Estadual de Maringá - Ciência Da Computação (DIN)
Aluno: Otávio Hideki Gonçalves Kochi RA:107635
Aluno: Thiago Issao Yasunaka RA:103069

Trabalho1 – Algoritmo em Grafos 6898
Árvores Aleatórias

Parte 3 – Função randomTreePrim(T)
e
mst_prim()

IMPLEMENTACÃO UTILIZANDO A LINGUAGEM JAVASCRIPT

Repositório no GitHub:
https://github.com/thiagoissao/arvore_aleatorias_grafos/blob/master/prim.js

Código:

```
const assert = require('assert')
const BRANCO = 'branco'
const CINZA = 'cinza'
const PRETO = 'preto'

const extract_min = (Q, G) => {
    let menor = Number.POSITIVE_INFINITY
    let index = -1
    Q.forEach( (v, i) => {
        if(G[v].chave < menor ){
            menor = v
            index = i
        }
    })
    Q.splice(index, 1)
    return menor
}

const belongsTo = (v, Q) => {
    for(let i=0; i<Q.length; i++)
        if(Q[i] == v) return true
    return false
}

const mst_prim = (G, w, r) => {
    Object.keys(G).forEach(u => {
        G[u] = {
            adj: G[u],
```

```
            chave: Number.POSITIVE_INFINITY,
            pi: null
        }
    })
    G[r].chave = 0
    let Q = Object.keys(G)
    while(Q.length != 0){
        const u = extract_min(Q, G)
        G[u].adj.forEach((v, index) => {
            if(belongsTo(v, Q) && w[u][index] < G[v].chave){
                G[v].pi = u
                G[v].chave = w[u][index]
            }
        })
    }
    let A = []
    for(let i = 0; i<G.length; i++) A[i] = []
    G.map( (v, index) => {
        if(v.pi != null){
            A[v.pi].push(index)
            A[index].push(Number(v.pi))
        }
    })
    return A
}

const createEdges = (current, n) => {
    let e = Array()
    for(let i=0; i < n; i++)
        if(i != current) e.push(i)
    return e
}

const createWeights = (end) => {
    let w = Array()
    for(let i=0; i < end - 1; i++)
        w.push(Math.random())
    return w
}

const random_tree_prim = n => {
    const G = Array(n).fill(null)
    G.forEach((element,i) => G[i] = createEdges(i, n))
    const w = Array(n).fill(null)
    w.forEach((element, i) => w[i] = createWeights(n))
    let u = Math.round(Math.random() * (n - 1))
    return mst_prim(G, w, u)
}
```

```
//*********************************************************
const diametro = (G) => {
    const s = Math.round(Math.random() * (G.length - 1))
    const a = bfs(G, s).reduce((number1, number2) =>
Math.max(number1, number2))
    const b = bfs(G,a).reduce((number1, number2) => Math.max(number1,
number2))
    return b > a ? b:a
}

const numero_arestas = (G) => {
    let contador = 0;
    if(G.length > 1)
        for(let i = 0; i < G.length; i ++){
            for(let j = 0; j < G[i].length; j ++){
                if(G[i][j] <= i)
                    contador++;
            }
        }
    return contador;
}
const bfs = (G, s) => {
    let GAux = Object.assign({}, G)
    Object.keys(GAux).forEach( index => {
        let adj = G[index]
        GAux[index] = {
            'd': Number.POSITIVE_INFINITY,
            'pi': null,
            'cor': BRANCO,
            'adj':  adj
        }
    })
    GAux[s].d = 0
    GAux[s].pi = null
    GAux[s].cor = CINZA
    let fila = [] //Inicialização da fila
    fila.push(GAux[s])
    indexU = 0
    while (indexU != G.length) {
        let u = fila[indexU]
        indexU++
        u.adj.forEach( v => {
            if(GAux[v].cor === BRANCO) {
                GAux[v].d = u.d + 1
                GAux[v].pi = G.indexOf(u.adj)
                GAux[v].cor = CINZA
                fila.push(GAux[v])
```

```
            }
        })
        u.cor = PRETO
    }
    return Object.keys(GAux).map(index => GAux[index].d)
}

const eh_arvore = G => {
    const arestas = numero_arestas(G)
    if(arestas != (G.length - 1)){
        return false
    }
    const s = Math.floor(Math.random() * (G.length - 1))
    const aux = bfs(G, s)
    for(let i = 0; i < G.length; i ++){
        if(aux[i] === Number.POSITIVE_INFINITY)
            return false
    }
    return true
}

const teste_arvore = () => {
    const n = [250, 500, 750, 1000, 1250, 1500, 1750, 2000]
    n.forEach(number => {
        let soma_diametro = 0
        for(let i=0; i<500; i++) {
            let G = random_tree_prim(number)
            assert(eh_arvore(G))
            soma_diametro = soma_diametro + diametro(G)
        }
        let media = soma_diametro/500
        console.log(number + ' ' + media)
    })
}
//************************

const test_mst_prim = () => {
    const a = 0
    const b = 1
    const c = 2
    const d = 3
    const e = 4
    const f = 5
    const g = 6
    const h = 7
    const i = 8

    const G = [
```

```
        [b, h],         //a
        [a, c, h],      //b
        [b, d, i ,f],   //c
        [c, e, f],      //d
        [d, f],         //e
        [c, d, e, g],   //f
        [f, i, h],      //g
        [a, b, i, g],   //h
        [c, g, h],      //i
    ]

    const w = [
        [4, 8],
        [4, 8, 11],
        [8, 7, 2, 4],
        [7, 9, 14],
        [9, 10],
        [4, 14, 10, 2],
        [2, 6, 1],
        [8, 11, 7, 1],
        [2, 6, 7]
    ]
    // console.log(mst_prim(G, w, 0))
    console.log(mst_prim(G, w, Math.round(Math.random()*(w.length -
1))))
}
// test_mst_prim()
teste_arvore()
```