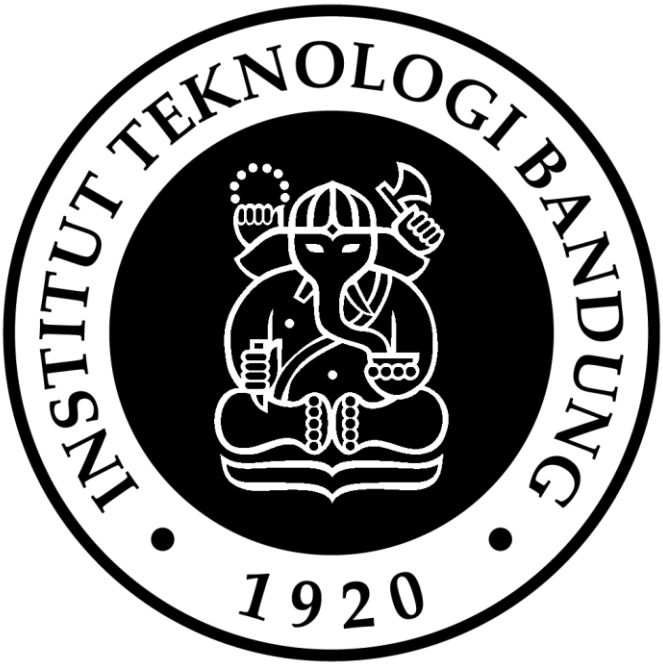
Laporan Tugas Kecil 2 Strategi Algoritma IF2211

Penyusunan Rencana Kuliah dengan Topological Sort

(Penerapan Decrease and Conquer)

Semester II 2020/2021



PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2021

Daftar Isi

BAGIAN I
ALGORITMA TOPOLOGICAL SORT

Berikut algoritma dari topological sorting yang diimplementasikan untuk tugas kecil ini:

1. File txt akan dibaca kemudian dikonversi menjadi graf berarah dengan definisi tiap entitasnya seperti pada source code program (Inisialisasi nilai-nilai graf)
2. Akan ditelusuri untuk semua node pada graf yang tidak memiliki predecessor (atau bisa dikatakan derajat masuknya = 0)
3. Node tanpa predecessor akan dihapus dari graf dan dimasukkan ke dalam sebuah senarai yang berisi node solusi untuk rekursi yang sekarang
4. Untuk node yang tersisa pada graf, hilangkan panah yang masuk yang diakibatkan oleh node yang telah dihapus (dalam program ini yaitu menghilangkan nama node predecessor dari senarai node predecessor pada atribut node)
5. Lakukan langkah 2-4 sampai semua node terpilih dan graf menjadi kosong(hanya berlaku untuk Directed Acyclic Graph)
6. Output solusi dari sorting node-node yang telah dipilih

Pada algoritma ini diterapkan decrease and conquer dengan jenis decrease by variable size, dapat dilihat dari setiap rekursi aka nada node yang dihilangkan di setiap rekursinya (jika graf merupakan DAG).

BAGIAN II
SOURCE CODE

Source code program ditulis dalam Bahasa C++ dibagi menjadi beberapa modul yaitu:

1. Node
2. Graph
3. ScannerGraph

Dapat diakses melalui pranala berikut : https://github.com/rdyzakya/tucil2stima

Berikut adalah source code dari program:

{Modul Node}

```
#ifndef __NODE_HPP__
#define __NODE_HPP__
#include <string>
#include <vector>
#include <algorithm>
#include <iostream>
using namespace std;

class Node {
private:
        string name;
        vector<string> predecessor;
        int num_pred;
public:
        Node(string name);
        ~Node();
        string getName();
        int getNumPred();
        void addPredNode(string pred_name);
        void delPredNode(string pred_name);
        void show();
};
```

```
#endif
```

```
#include "13519061-Node.hpp"


//constructor
Node::Node(string name){
		this->name = name;
		this->num_pred = 0;
	}


//menambah node predecessor berupa namanya saja
void Node::addPredNode(string pred_name){
	predecessor.push_back(pred_name);
	num_pred++;
}


//menghapus node predecessor berupa namanya saja
void Node::delPredNode(string pred_name){
	predecessor.erase(remove(predecessor.begin(),predecessor.end(),pred_name),predecessor.end
());
	num_pred = predecessor.size();
}


//destructor
Node::~Node(){
	predecessor.clear();
}


//dibuat untuk debugging
void Node::show() {
	cout << this->name << endl << "Predesesor: " << endl;
	for (int i = 0; i < num_pred; ++i)
	{
		cout << predecessor[i] << endl;
```

```
        }
}

//getter untuk nama node
string Node::getName() {
        return name;
}

//getter untuk jumlah predecessor dari node
int Node::getNumPred() {
        return num_pred;
}
```

{Modul Graph}

```
#ifndef __GRAPH_HPP__
#define __GRAPH_HPP__
#include "13519061-Node.hpp"
#include <iostream>
using namespace std;

class Graph {
private:
        vector<Node> nodes;
        int num_node;
public:
        Graph();
        void addNode(Node add_node);
        void deleteNode(string del_nodename);
        ~Graph();
        vector<string> noPredecessor();
        bool isAcyclic();
        void topologicalSort(vector<vector<string>> * result, bool * solved);
        void result();
        void show();
};
```

```
#endif
```

```
#include "13519061-Graph.hpp"

//constructor
Graph::Graph() {
	num_node = 0;
}

//menambah node beserta atributnya
void Graph::addNode(Node add_node) {
	nodes.push_back(add_node);
	num_node++;
}

//menghapus node beserta atributnya
void Graph::deleteNode(string del_nodename) {
	for (int i = 0; i < nodes.size(); ++i)
	{
		if (del_nodename.compare(nodes[i].getName()) == 0)
		{
			nodes.erase(nodes.begin()+i);
			num_node--;
		}
	}
}

//destructor
Graph::~Graph(){
	nodes.clear();
}

//mengembalikan list nama node yang tidak memiliki predecessor
vector<string> Graph::noPredecessor() {
	vector<string> result;
```

```
	for (int i = 0; i < num_node; ++i)
	{
		if (nodes[i].getNumPred() == 0)
		{
			result.push_back(nodes[i].getName());
		}
	}
	return result;
}

//mengembalikan keadaan graf asiklik atau siklik
bool Graph::isAcyclic() {
	return noPredecessor().size() != 0;
}

//pemilihan matkul dengan topological sort
void Graph::topologicalSort(vector<vector<string>> * result , bool * solved){
	if (isAcyclic() && num_node > 0)
	{
		vector<string> sub_result = noPredecessor();
		//nama node yang tidak memiliki predecessor atau matkul yang tidak memiliki
		//prerequisite
		result->push_back(sub_result);
		for (int i = 0; i < num_node; ++i)
		//menghilangkan nama node pada list di atas untuk setiap node yang memiliki
		//predecessor dengan nama node terkait
		{
			for (int j = 0; j < sub_result.size(); ++j)
			{
				nodes[i].delPredNode(sub_result[j]);
			}
		}
		for (int k = 0; k < sub_result.size(); ++k)
		//delete node dari graph, ini bagian decreasenya
		{
			deleteNode(sub_result[k]);
```

```
                }
                topologicalSort(result , solved);//rekursi
        }else if (!isAcyclic() && num_node > 0){
        //jika graph bukan merupakan DAG
                *solved = false;
        }
        else{
        //jika sorting telah selesai dilaksanakan
                *solved = true;
        }
}

//output hasil dari topological sort
void Graph::result() {
        vector<vector<string>> my_result;
        bool isSolved;
        topologicalSort(&my_result,&isSolved);
        if (!isSolved)
        {
                cout << "Rencana matakuliah tidak dapat dibuat (Graf matakuliah bukan berupa DAG)!"
<< endl;
        }else{
                cout << "Hasil penjadwalan matakuliah :" << endl;
                for (int i = 0; i < my_result.size(); ++i)
                {
                        cout << "Semester " << i+1 << " : ";
                        for (int j = 0; j < my_result[i].size(); ++j)
                        {
                                if (j != 0)
                                {
                                        cout << ", ";
                                }
                                cout << my_result[i][j];
                        }
                        cout << endl;
                }
```

```
        }
}

//digunakan untuk debugging
void Graph::show() {
        for (int i = 0; i < num_node; ++i)
        {
                nodes[i].show();
                cout << endl;
        }
}
```

{Modul ScannerGraph}

```
#ifndef __SCANNERGRAPH_HPP__
#define __SCANNERGRAPH_HPP__
#include "13519061-Graph.hpp"
#include <fstream>
#include <sstream>
#include <algorithm>
#include <unistd.h>

using namespace std;

class ScannerGraph {
public:
        Graph readTxt(string file_name);
};

#endif
```

```
#include "13519061-ScannerGraph.hpp"

//membaca file dan mengubahnya menjadi sebuah graf
Graph ScannerGraph::readTxt(string file_name){
```

```
	Graph my_graph;
	ifstream file("..\\test\\" + file_name + ".txt");
	string line;
	while(getline(file,line)){
		line.erase(remove(line.begin(),line.end(),' '), line.end());
		line.erase(remove(line.begin(),line.end(),'.'), line.end());
		stringstream my_line(line);
		string node_name;
		vector<string> node_line;
		while(getline(my_line,node_name, ',')){
			node_line.push_back(node_name);
		}
		Node main_node(node_line[0]);
		for (int i = 1; i < node_line.size(); ++i)
		{
			main_node.addPredNode(node_line[i]);
		}
		my_graph.addNode(main_node);
	}
	file.close();
	return my_graph;
}
```

{Main Program RSA}

```
#include "13519061-ScannerGraph.hpp"
#include <iostream>
using namespace std;

//penambahan fitur seperti loading
void isengStart() {
	cout << "RSA Preparing Things";
	for (int i = 0; i < 3; ++i)
	{
		sleep(1);
		cout << ".";
```

```
        }
        cout << endl << endl;
}

int main() {
        isengStart();
        cout << "Selamat datang di RSA (Randy's Scheduling Algorithm)" << endl;
        string file_name;
        cout << "Masukkan nama file matakuliah: ";
        cin >> file_name;
        ScannerGraph my_scanner;
        Graph my_graph = my_scanner.readTxt(file_name);
        my_graph.result();
        cout << endl << "RSA Shutting Down..." << endl;
        return 0;
}
```

BAGIAN III
HASIL PERCOBAAN

Berikut merupakan percobaan program pada beberapa test case:

1. Test case 1

   Kriptografi, Matdis.

   Kalkulus.

   TBFO, Matdis.

   Fisika.

   Stima, Matdis, Kalkulus.

   Matdis, Kalkulus.

```
D:\Documents\Kuliah semester 4\stima\tucil2stima\bin>RSA
RSA Preparing Things...

Selamat datang di RSA (Randy's Scheduling Algorithm)
Masukkan nama file matakuliah: grafTest1
Hasil penjadwalan matakuliah :
Semester 1 : Kalkulus, Fisika
Semester 2 : Matdis
Semester 3 : Kriptografi, TBFO, Stima

RSA Shutting Down...
```

2. Test case 2

   MA1201, MA1101.

   FI1201, FI1101.

   IF1210, KU1102.

   KU1202, KU1102.

   KI1002, KU1011.

   EL1200, FI1101.

   KU1102.

   MA1101.

   FI1101.

KU1011.

```
D:\Documents\Kuliah semester 4\stima\tucil2stima\bin>RSA
RSA Preparing Things...

Selamat datang di RSA (Randy's Scheduling Algorithm)
Masukkan nama file matakuliah: grafTest2
Hasil penjadwalan matakuliah :
Semester 1 : KU1102, MA1101, FI1101, KU1011
Semester 2 : MA1201, FI1201, IF1210, KU1202, KI1002, EL1200

RSA Shutting Down...
```

3. Test case 3

   MA1101.

   FI1101.

   KU1001.

   KU1102.

   KU1011.

   KU1024.

   MA1201, MA1101.

   FI1201, FI1101.

   IF1210.

   KU1202.

   EL1200, MA1101.

   IF2121.

   IF2110.

   IF2120.

   IF2124.

   IF2123, MA1101.

   IF2130.

   IF2210, IF2110.

   IF2211.

IF2220, MA1101, MA1201, IF2120.

IF2230.

IF2240.

IF2250.

IF3170, IF2121, IF2124, IF2220, IF2211.

IF3110, IF2210, IF2110.

IF3130, IF2230.

IF3141, IF2240, IF2250.

IF3150, IF2250.

IF3140.

IF3151, IF2250.

IF3210, IF2130, IF2110.

IF3270, IF3170, IF2110.

IF3230, IF3130.

IF3250, IF3150, IF2250.

IF3260, IF2130, IF2110, IF2123.

IF3280.

IF4090, IF3280.

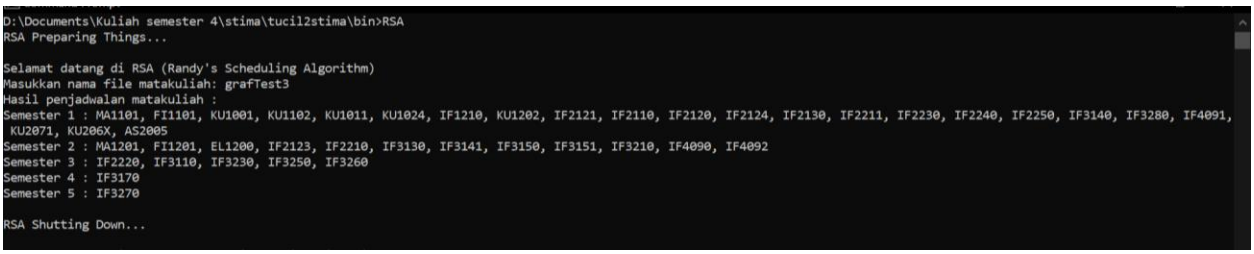
IF4091.

KU2071.

IF4092, IF4091.

KU206X.

AS2005.



4. Test case 4

Flask, Python, Pip.

Pip, Python.

Python, C.

C.

```
D:\Documents\Kuliah semester 4\stima\tucil2stima\bin>RSA
RSA Preparing Things...

Selamat datang di RSA (Randy's Scheduling Algorithm)
Masukkan nama file matakuliah: grafTest4
Hasil penjadwalan matakuliah :
Semester 1 : C
Semester 2 : Python
Semester 3 : Pip
Semester 4 : Flask

RSA Shutting Down...
```

5. Test case 5

   MA1101.

   FI1101.

   KU1001.

   KU1102.

   KU1011.

   KU1024.

   MA1201, MA1101.

   FI1201, FI1101.

   IF1210, KU1102.

   KU1202, KU1102.

   KI1002, KU1011.

   EL1200, FI1101.

   IF2121, IF1210, MA1101, MA1201.

   IF2110, KU1102, IF1210.

   IF2120, MA1201, MA1101.

   IF2124, EL1200.

IF2123, MA1201.

IF2130, KU1202.

IF2210, IF2110.

IF2211, IF2110.

IF2220, MA1101, MA1201, IF2120.

IF2230, IF2130.

IF2240, IF2121, IF2120.

IF2250, KU1202, IF2110.

IF3170, IF2121, IF2124, IF2220, IF2211.

IF3110, IF2210, IF2110.

IF3130, IF2230.

IF3141, IF2240, IF2250.

IF3150, IF2250.

IF3140, IF2240.

IF3151, IF2250.

IF3210, IF2110, IF2130, IF3110.

IF3270, IF2210, IF3170.

IF3230, IF3130.

IF3250, IF2250, IF3150.

IF3260, IF2123, IF2110, IF2130, IF3151.

IF3280, IF3151, IF3150.

IF4090, IF3280.

IF4091, IF3280.

IF4092, IF4091.

```
D:\Documents\Kuliah semester 4\stima\tucil2stima\bin>RSA
RSA Preparing Things...

Selamat datang di RSA (Randy's Scheduling Algorithm)
Masukkan nama file matakuliah: grafTest5
Hasil penjadwalan matakuliah :
Semester 1 : MA1101, FI1101, KU1001, KU1102, KU1011, KU1024
Semester 2 : MA1201, FI1201, IF1210, KU1202, KI1002, EL1200
Semester 3 : IF2121, IF2110, IF2120, IF2124, IF2123, IF2130
Semester 4 : IF2210, IF2211, IF2220, IF2230, IF2240, IF2250
Semester 5 : IF3170, IF3110, IF3130, IF3141, IF3150, IF3140, IF3151
Semester 6 : IF3210, IF3270, IF3230, IF3250, IF3260, IF3280
Semester 7 : IF4090, IF4091
Semester 8 : IF4092

RSA Shutting Down...
```

6. Taste case 6

   C1,C3

   C2,C1,C4.

   C3.

   C4,C1,C3.

   C5,C2,C4.

```
D:\Documents\Kuliah semester 4\stima\tucil2stima\bin>RSA
RSA Preparing Things...

Selamat datang di RSA (Randy's Scheduling Algorithm)
Masukkan nama file matakuliah: tc1
Hasil penjadwalan matakuliah :
Semester 1 : C3
Semester 2 : C1
Semester 3 : C4
Semester 4 : C2
Semester 5 : C5

RSA Shutting Down...
```

7. Test case 7

   C1, C2.

C2.

C3.

C4, C2, C3.

C5, C4.

C6, C1, C4, C5.

C7, C5.

```
D:\Documents\Kuliah semester 4\stima\tucil2stima\bin>RSA
RSA Preparing Things...

Selamat datang di RSA (Randy's Scheduling Algorithm)
Masukkan nama file matakuliah: tc2
Hasil penjadwalan matakuliah :
Semester 1 : C2, C3
Semester 2 : C1, C4
Semester 3 : C5
Semester 4 : C6, C7

RSA Shutting Down...
```

8. Test case 8

   C3.

   C2, C3, C1, C4.

   C1, C3.

   C4.

   C5, C1,C2, C6.

   C6, C4.

   C8, C5,C6.

   C9.

   C7, C8, C9.

```
D:\Documents\Kuliah semester 4\stima\tucil2stima\bin>RSA
RSA Preparing Things...

Selamat datang di RSA (Randy's Scheduling Algorithm)
Masukkan nama file matakuliah: tc3
Hasil penjadwalan matakuliah :
Semester 1 : C3, C4, C9
Semester 2 : C1, C6
Semester 3 : C2
Semester 4 : C5
Semester 5 : C8
Semester 6 : C7

RSA Shutting Down...
```

TABEL PENILAIAN

Poin	Ya	Tidak
Program berhasil dikompilasi	✓	
Program berhasil running	✓	
Program dapat menerima berkas input dan menuliskan output	✓	
Luaran sudah benar untuk semua kasus input	✓	