

LAPORAN TUGAS KECIL 2 IF2211 STRATEGI ALGORITMA

Penyusunan Rencana Kuliah dengan *Topological Sort*

(Penerapan *Decrease and Conquer*)



Disusun Oleh:

Delisha Azza Naadira

13519133

**PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG**

2021

A. Algoritma *Decrease and Conquer*

Decrease and conquer merupakan algoritma yang dirancang dengan mereduksi suatu persoalan menjadi dua sub-persoalan yang lebih kecil, tetapi selanjutnya hanya memproses satu sub-persoalan saja. Algoritma *decrease and conquer* terdiri dari dua tahapan, yaitu *decrease* yang mereduksi persoalan menjadi beberapa persoalan yang lebih kecil (biasanya dua sub-persoalan), dan *conquer* yang memproses salah satu sub-persoalan secara rekursif.

Terdapat tiga jenis algoritma *decrease and conquer*, yaitu:

1. *Decrease by a constant*: ukuran instans persoalan direduksi sebesar konstanta yang sama setiap iterasi algoritma. Biasanya konstanta 1.
2. *Decrease by a constant factor*: ukuran instans persoalan direduksi sebesar faktor konstanta yang sama setiap iterasi algoritma. Biasanya faktor konstanta 2.
3. *Decrease by a variable size*: ukuran instans persoalan direduksi bervariasi pada setiap iterasi algoritma.

B. *Topological Sort*

Topological sort merupakan pengurutan simpul-simpul suatu graf asiklik berarah (Directed Acyclic Graph/DAG) secara linier sehingga setiap ada sisi yang menghubungkan simpul A dan B, A akan terurut sebelum B. Dalam kehidupan sehari-hari, *topological sorting* digunakan untuk penjadwalan urutan pekerjaan atau tugas berdasarkan ketergantungan masing-masing tugas/pekerjaan, menentukan urutan kompilasi untuk dilakukan di makefiles, dan untuk tugas kecil ini, menyusun rencana kuliah.

Program untuk melakukan *topological sort* dibuat dengan menggunakan bahasa Python. Program utama meng-*import module* `os` dan *module-module* lain untuk menjalankan program. Pada masing-masing *module* terdapat beberapa fungsi, yang nantinya akan digunakan pada program utama.

Topological sorting dilakukan pada program utama. Pertama-tama, baca file input, kemudian berdasarkan input tersebut buat list yang berisikan list simpul mata kuliah dan semua simpul mata kuliah *prerequisite*.

Setelah itu, dibuat list untuk mengurutkan simpul-simpul dengan *topological sort*. Dicari mata kuliah yang tidak memiliki *prerequisite* atau mata kuliah yang semua *prerequisite*-nya sudah terpenuhi, kemudian dimasukkan ke suatu list yang kemudian dimasukkan ke list urutan simpul. Matkul-matkul tersebut kemudian dihapus dari semua elemen list simpul mata kuliah dan *prerequisite*. Tahap ini diulang terus menerus hingga semua list mata kuliah kosong. Mata kuliah tidak langsung dimasukkan ke dalam list *topological sorting* jadi satu karena mahasiswa bisa saja mengambil dua atau lebih mata kuliah yang tidak memiliki *prerequisite* atau semua *prerequisite* sudah terpenuhi dalam satu semester.

Setelah terbentuk urutan mata kuliah, program membentuk string susunan rencana kuliah. Susunan tersebut ditampilkan ke layar dan juga dikeluarkan dalam bentuk `output.txt`.

Algoritma *decrease and conquer* yang digunakan pada program ini adalah ketika mencari mata kuliah yang tidak memiliki *prerequisite* atau yang *prerequisite*-nya sudah terpenuhi semua, kemudian persoalan direduksi dengan menghapus mata kuliah tersebut dari list. Hal ini dilakukan terus-menerus hingga semua mata kuliah terpilih.

C. Source Program

```
# File: graph_13519133.py
# File berisi fungsi-
fungsi untuk membuat DAG/Directed Acyclic Graph dari masukan file

# Fungsi untuk membuat list yang berisi simpul mata kuliah dan prerequisitenya
berdasarkan bacaan line
# Elemen pertama list merupakan nama simpul, sedangkan elemen-
elemen berikutnya merupakan semua simpul masuk
# Jika simpul tidak memiliki simpul masuk, list hanya terdiri dari satu elemen
def list_of_vertices(line):
    vertices = [] # list berisi simpul mata kuliah dan prerequisitenya
    namelist = [] # list of char tiap nama mata kuliah yang terbaca
    for c in line:
        if c == ' ':
            continue
        elif c == ',' or c == '.':
            name = ''.join(namelist) # mengubah list menjadi string nama mata
                                    # kuliah
            vertices.append(name)
            namelist = []
        else:
            namelist.append(c)
    return vertices

# Fungsi untuk membuat list DAG, yaitu list yang berisi list simpul-
simpul mata kuliah dan prerequisitenya
def make_dag_list(input):
    dag = []
    for line in input:
        dag.append(list_of_vertices(line))
    return dag
```

```
# File: list_13519133.py
# File berisi fungsi-
fungsi untuk membuat dan mengubah list yang berhubungan dengan DAG

# Fungsi untuk membuat list semua matkul yang tidak memiliki prerequisitenya
ngga dapat langsung diambil dalam satu semester
def make_matkul_list(list):
    matkul_list = []
    for l in list:
        if len(l) == 1: # List matkul yang tidak memiliki prerequisitenya akan
                        # hanya memiliki satu elemen yaitu matkul itu sendiri
            matkul_list.append(l[0])
    return matkul_list
```

```

# Fungsi untuk menghapus matkul yang sudah diambil dari list graph
def del_matkul_from_list(mlist, glist):
    i = 0
    j = len(glist)
    while (i != j):
        for m in mlist:
            glist[i] = del_from_list(m, glist[i])
            if len(glist[i]) == 0: # Jika list kosong setelah matkul dihapus, list
                                # tersebut dihapus dari list graf
                glist.pop(i)
                j -= 1
        else:
            i += 1
    return glist

# Fungsi untuk delete elemen dari list
# Jika pada list terdapat elemen n, maka n di-delete dari list
def del_from_list(n, list):
    i = 0
    while (i < len(list)):
        if list[i] == n:
            list.pop(i)
        else:
            i += 1
    return list

```

```

# File: others_13519133.py
# File berisi fungsi-fungsi lain untuk menjalankan program

# Fungsi untuk mengubah integer menjadi angka romawi
# Diasumsikan pengambilan mata kuliah hanya 8 semester sehingga angka romawi y
ang digunakan hanya I, IV dan V
def roman(n):
    ints = (5, 4, 1)
    nums = ('V', 'IV', 'I')
    res = []

    for i in range(len(ints)):
        count = int(n / ints[i])
        res.append(nums[i] * count)
        n -= ints[i] * count
    return ''.join(res)

# Fungsi untuk mengubah array maupun tuple menjadi string
def to_string(l):
    return ''.join(l)

```

```

# Fungsi untuk membuat string susunan rencana kuliah berdasarkan topological sort yang telah didapat
def output(list):
    string = ["----- Susunan Rencana Kuliah -----\\n\\n"]
    for i in range(len(list)):
        tup = "Semester ", roman(i+1), ": ", ', '.join(list[i])
        string.append(to_string(tup) + '\\n')
    return to_string(string)

```

```

# File: main_13519133.py
# Program Utama

import list_13519133
import graph_13519133
import others_13519133
import os

path = os.path.dirname(__file__)
input = os.path.relpath('../test/3.txt', path)
# file '1.txt' dapat diganti dengan file apapun yang ada di dalam folder test

# Membaca file dan membuat list DAG berdasarkan bacaan file
f = open(input, "r")
graphList = graph_13519133.make_dag_list(f)
f.close()

# Melakukan algoritma Topological Sort
# Membuat list topSort, yaitu list yang berisi list semua mata kuliah yang dapat diambil dalam satu semester
# Banyak indeks list topSort merupakan banyak semester yang dibutuhkan untuk mengambil semua mata kuliah
# Iterasi hingga graphList kosong, yaitu hingga semua mata kuliah terambil
topSort = []
while(len(graphList) != 0):
    matkulList = list_13519133.make_matkul_list(graphList)
    graphList = list_13519133.del_matkul_from_list(matkulList, graphList)
    topSort.append(matkulList)

# Write dan print output
savePath = '../test'
fileName = "output.txt"
dirFile = os.path.join(savePath, fileName)
outputFile = open(dirFile, "r+") # membuat file output susunan rencana kuliah
outText = others_13519133.output(topSort)
print(outText) # print susunan rencana kuliah ke layar
outputFile.write(outText)
outputFile.close()

```

GitHub: https://github.com/delishaandr/Tucil2_13519133

D. Input dan Output

Percobaan	Input	Output
1	C1, C3. C2, C1, C4. C3. C4, C1, C3. C5, C2, C4.	----- Susunan Rencana Kuliah ----- Semester I: C3 Semester II: C1 Semester III: C4 Semester IV: C2 Semester V: C5
2	C1, C8. C2. C3, C2. C4, C2, C3, C5. C5, C1, C2. C6, C1, C7, C8. C7, C8. C8.	----- Susunan Rencana Kuliah ----- Semester I: C2, C8 Semester II: C1, C3, C7 Semester III: C5, C6 Semester IV: C4
3	C1, C2, C3, C7. C2, C7. C3, C2, C5. C4, C1, C3, C5, C8. C5. C6, C1, C4. C7. C8, C5.	----- Susunan Rencana Kuliah ----- Semester I: C5, C7 Semester II: C2, C8 Semester III: C3 Semester IV: C1 Semester V: C4 Semester VI: C6
4	MK1, MK2. MK2. MK3, MK1, MK4, MK6. MK4, MK1. MK5, MK2, MK3. MK6, MK4.	----- Susunan Rencana Kuliah ----- Semester I: MK2 Semester II: MK1 Semester III: MK4 Semester IV: MK6 Semester V: MK3 Semester VI: MK5
5	Stima. PBO, Stima. OS, PBO. Algeo, PBO, Probststat. TBFO, PBO, OS, Algeo. RPL, TBFO. Probststat.	----- Susunan Rencana Kuliah ----- Semester I: Stima, Probststat Semester II: PBO Semester III: OS, Algeo Semester IV: TBFO Semester V: RPL
6	AR114, MT126. MT126. CE133, AR114, TD149, GC167. TD149, ES150. ES150. GC167, ES150, MT126. KS178, AR114, CE133, LN181. LN181, MT126, ES150, GC167.	----- Susunan Rencana Kuliah ----- Semester I: MT126, ES150 Semester II: AR114, TD149, GC167 Semester III: CE133, LN181 Semester IV: KS178

7	K1. K2, K1. K3, K1. K4, K2. K5, K2, K6. K6, K3. K7, K4, K5, K6.	----- Susunan Rencana Kuliah ----- Semester I: K1 Semester II: K2, K3 Semester III: K4, K6 Semester IV: K5 Semester V: K7
8	A501. A202, A113. A113, A501, A704. A704. A905, A113, A806. A806, A704, A307. A307. A108, A113, A307.	----- Susunan Rencana Kuliah ----- Semester I: A501, A704, A307 Semester II: A113, A806 Semester III: A202, A905, A108

Tabel Penilaian

Poin	Ya	Tidak
1. Program berhasil dikompilasi.	√	
2. Program berhasil <i>running</i> .	√	
3. Program dapat menerima berkas input dan menuliskan output.	√	
4. Luaran sudah benar untuk semua kasus input.	√	