LAPORAN TUGAS KECIL II

PENYUSUNAN RENCANA KULIAH DENGAN *TOPOLOGICAL SORT* (PENERAPAN *DECREASE AND CONQUER*)

IF2211 Strategi Algoritma

Oleh:

Alif Bhadrika Parikesit 13519186

Kelas:

K-04

PROGRAM STUDI TEKNIK INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

BANDUNG

2021

BAB I

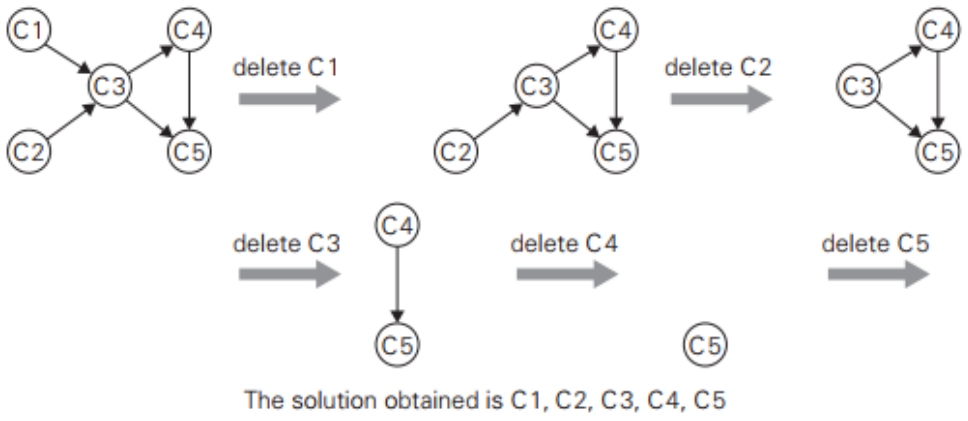
ALGORITMA *DECREASE AND CONQUER*

Algoritma *decrease and conquer* merupakan salah satu metode penyelesaian permasalahan secara komputasional. Algoritma ini bekerja dengan cara mereduksi persoalan menjadi *sub*-persoalan yang lebih kecil, kemudian memproses satu sub-persoalan itu saja. Berbeda dengan *divide and conquer* yang memproses semua *sub*-persoalan dan menggabungkan semua solusi dari setiap sub-persoalan menjadi solusi persoalan secara keseluruhan.

Tahapan pada algoritma *decrease and conquer* seperti namanya, yaitu *decrease* kemudian *conquer*. *Decrease* bekerja dengan mereduksi persoalan menjadi beberapa sub-persoalan yang lebih kecil kemudian *conquer*, menaklukan, yaitu memproses satu sub-persoalan secara rekursif.

Pada tugas kali ini, penulis akan menerapkan algoritma *decrease and conquer* pada permasalahan penyusunan rencana kuliah yang memiliki mata kuliah prasayarat dengan *topological sort*. Metode *topological sort* dapat diimplementasikan secara *decrease and conquer*. Adapun yang harus diperhatikan adalah apabila menggunakan pendekatan *topological sort*, maka harus dipastikan bahwa graf berarah tersebut harus tidak membentuk siklus *Directed Acyclic Graph* (DAG), karena apabila membentuk siklus maka tidak dapat menggunakan pendekatan *topological sort* ini. Kemudian, ada batasan yaitu mata kuliah tidak boleh diambil bersamaan dengan prasyaratnya.

Berikut ini adalah ilustrasi dari penerapa *topological sorting* pada sebuah graf.



Gambar 1. Ilustrasi *Topological Sorting*
Sumber: Levitin, Anany. *Introduction to the design & analysis of algorithms*

Adapun implementasi *topological sort* dengan algoritma *decrease and conquer* yaitu sebagai berikut:

1. Representasikan daftar mata kuliah sebagai simpul pada graf berarah, dengan arah keluar menyatakan mata kuliah tersebut memiliki prasyarat yaitu simpul mata kuliah tujuan sisi berarah tersebut.
2. Identifikasi simpul pada graf yang memiliki derajat masuk = 0, dalam kasus mata kuliah, artinya mata kuliah tersebut tidak memiliki prasyarat.
3. Kemudian, hapus simpul tersebut pada graf dan sisi yang keluar dari simpul tersebut. Dalam kasus ini, misalkan simpul mata kuliah A berderajat masuk 0, maka simpul A akan dihapus dari graf dan seluruh mata kuliah yang tadinya memiliki prasyarat A sekarang tidak memiliki prasyarat A. Dengan demikian ukuran graph berkurang satu.
4. Periksa apakah mata kuliah dapat dimasukkan kedalam satu semester dengan cara memeriksa apakah simpul yang akan dimasukkan ke dalam himpunan solusi memiliki prasyarat berupa mata kuliah yang dihapus pada satu iterasi tepat sebelumnya. Jika ya, maka mata kuliah tersebut tidak dapat ditempatkan pada satu semester yang sama dengan prasyaratnya.
5. Ulangi langkah 2-4 untuk menemukan urutan mata kuliah per semesternya.

Alasan solusi ini merupakan pendekatan algoritma *decrease and conquer* adalah karena metode *topological sort* mengikuti kaidah varian *decrease and conquer* yang pertama yaitu *decrease by constant*, artinya ukuran dari *instance* berkurang secara konstan setiap pengulangannya. Ditunjukkan pada langkah ke-3, setiap simpul berderajat masuk = 0, maka simpul tersebut akan dihapus dari graf dan dimasukkan ke dalam solusi.

BAB II

SOURCE CODE PROGRAM *COURSE PREREQ SCHEDULING*

Source code program selengkapnya dapat diakses melalui tautan berikut:

https://github.com/alifbhadrika/course-prereq-scheduling

```
# February 27 2021, Alif Bhadrika Parikesit
def createGraphfromFile():
    '''
    createGraphfromfile adalahh fungsi parse file kemudian
    merepresentasikan data mata kuliah dari file sebagai
    graf dengan berarah dengan struktur adjacency list
    '''

    inputFile = input("ENTER inputFilename.txt: ")
    print()
    with open('../test/'+inputFile,'r') as file:
        graph = {}
        for line in file:
            line = (line.replace(',','').rstrip('.\n')).split()
            graph[line[0]] = (line[1:])
    return graph

def getStudyPlan(graph, sorted_course, course_per_semester):
    '''
    getStudyPlan melakukan topsort pada graf mata kuliah dan menyeleksi
    apakah mata kuliah dapat diletakkan pada satu semester yang sama
    atau tidak
    '''

    vertex = list(graph.keys())             #list simpul graf
```

```
    inDegree = dict.fromkeys(vertex,0)      #dict derajat masuk simpul
    smt = 1
    forbidden_course = []            #course yang memiliki prasyarat course v
    while len(graph) != 0:
        for v in vertex:
            inDegree[v] = len(graph[v])
            if inDegree[v] == 0:
                vertex.remove(v)
                sorted_course.append(v)

#cek apakah matkul yang baru masuk sorted_course tidak memiliki prasyarat v
#apabila iya, maka pisah kedua course tersebut
#apabila tidak, maka satukan keduanya pada satu semester yang sama

                if v not in forbidden_course:
                    if smt in course_per_semester.keys():
                        course_per_semester[smt].append(v)
                    else:
                        course_per_semester[smt] = [v]
                else:
                    smt += 1
                    course_per_semester[smt] = [v]

                forbidden_course.clear()

                #penghapusan sisi keluar dari simpul yang berderajat 0
                for course,neighbors in graph.items():
                    for neighbor in neighbors:
                        if v == neighbor:
                            forbidden_course.append(course)
                            neighbors.remove(neighbor)

                #penghapusan simpul berderajat masuk 0
```

```
                graph.pop(v, None)
    return

def print_solution(course_per_semester):
    '''
    print_solution mencetak solusi ke layar sesuai dengan
    format SEMESTER : <nama_course> tiap semesternya
    '''
    for semester,courses in course_per_semester.items():
        print("SEMESTER {}: ".format(semester),end="")
        for course in courses:
            print(course," ",end="")
        print()

if __name__ == '__main__':
    sorted_course = []
    course_per_semester = {}
    course_graph = createGraphfromFile()
    getStudyPlan(course_graph,sorted_course,course_per_semester)
    print("==== Sorted Courses List =====")
    print(sorted_course)
    print()
    print("======= Your Study Plan ======")
    print_solution(course_per_semester)
```

BAB III

PENGUJIAN PROGRAM

NAMA FILE	INPUT	OUTPUT
1.txt	C1, C3. C2, C1, C4. C3. C4, C1, C3. C5, C2, C4.	C:\Users\ASUS\Documents\course-prereq-scheduling\src>main.py ENTER inputFilename.txt: 1.txt ==== Sorted Courses List ===== ['C3', 'C1', 'C4', 'C2', 'C5'] ======= Your Study Plan ====== SEMESTER 1: C3 SEMESTER 2: C1 SEMESTER 3: C4 SEMESTER 4: C2 SEMESTER 5: C5
2.txt	C5. C4. C2, C5. C0, C5, C4. C1, C4, C3. C3, C2.	C:\Users\ASUS\Documents\course-prereq-scheduling\src>main.py ENTER inputFilename.txt: 2.txt ==== Sorted Courses List ===== ['C5', 'C2', 'C3', 'C4', 'C1', 'C0'] ======= Your Study Plan ====== SEMESTER 1: C5 SEMESTER 2: C2 SEMESTER 3: C3 C4 SEMESTER 4: C1 C0
3.txt	A. B. C, A. D, A, B. J, C, D. E. H, E, D. F, B. I, F.	C:\Users\ASUS\Documents\course-prereq-scheduling\src>main.py ENTER inputFilename.txt: 3.txt ==== Sorted Courses List ===== ['A', 'C', 'E', 'B', 'F', 'D', 'H', 'J', 'I'] ======= Your Study Plan ====== SEMESTER 1: A SEMESTER 2: C E B SEMESTER 3: F D SEMESTER 4: H J I
4.txt	C1. C2. C3, C1, C2. C4, C3. C5, C3, C4.	C:\Users\ASUS\Documents\course-prereq-scheduling\src>main.py ENTER inputFilename.txt: 4.txt ==== Sorted Courses List ===== ['C1', 'C2', 'C3', 'C4', 'C5'] ======= Your Study Plan ====== SEMESTER 1: C1 C2 SEMESTER 2: C3 SEMESTER 3: C4 SEMESTER 4: C5
5.txt	IF1111. IF1122, IF1111. IF1133, IF1111. IF1144, IF1122, IF1133. IF1155, IF1122, IF1144, IF1166, IF1144, IF1133.	C:\Users\ASUS\Documents\course-prereq-scheduling\src>main.py ENTER inputFilename.txt: 5.txt ==== Sorted Courses List ===== ['CS01', 'CS06', 'CS02', 'CS05', 'CS04', 'CS03', 'CS07'] ======= Your Study Plan ====== SEMESTER 1: CS01 SEMESTER 2: CS06 CS02 SEMESTER 3: CS05 SEMESTER 4: CS04 SEMESTER 5: CS03 SEMESTER 6: CS07

6.txt	C1. C2. C3. C4. C5, C1. C6, C2. C7, C3. C8, C4. C9, C1. C10, C2. C11, C3. C12, C4. C13, C5, C6, C7, C8. C14, C9, C10, C11, C12.	C:\Users\ASUS\Documents\course-prereq-scheduling\src>main.py ENTER inputFilename.txt: 6.txt ==== Sorted Courses List ===== ['C1', 'C3', 'C5', 'C7', 'C9', 'C11', 'C2', 'C6', 'C10', 'C4', 'C12', 'C14', 'C8', 'C13'] ======= Your Study Plan ====== SEMESTER 1: C1 C3 C5 C7 C9 C11 C2 SEMESTER 2: C6 C10 C4 SEMESTER 3: C12 SEMESTER 4: C14 C8 SEMESTER 5: C13
7.txt	A. B, A. D, A. C, B. E, D. F, E, C.	C:\Users\ASUS\Documents\course-prereq-scheduling\src>main.py ENTER inputFilename.txt: 7.txt ==== Sorted Courses List ===== ['A', 'D', 'E', 'B', 'C', 'F'] ======= Your Study Plan ====== SEMESTER 1: A SEMESTER 2: D SEMESTER 3: E B SEMESTER 4: C SEMESTER 5: F
8.txt	IF1111. IF1122, IF1111. IF1133, IF1111. IF1144, IF1122, IF1133. IF1155, IF1122, IF1144, IF1166, IF1144, IF1133.	C:\Users\ASUS\Documents\course-prereq-scheduling\src>main.py ENTER inputFilename.txt: 8.txt ==== Sorted Courses List ===== ['IF1111', 'IF1133', 'IF1122', 'IF1144', 'IF1166', 'IF1155'] ======= Your Study Plan ====== SEMESTER 1: IF1111 SEMESTER 2: IF1133 IF1122 SEMESTER 3: IF1144 SEMESTER 4: IF1166 IF1155

TABEL PENILAIAN

Poin	Kriteria	Ya	Tidak
1.	Program berhasil dikompilasi	**V**	
2.	Program berhasil running	**V**	
3.	Program dapat menerima berkas input dan menuliskan output.	**V**	
4.	Luaran sudah benar untuk semua kasus input.	**V**	

DAFTAR PUSTAKA

Levitin, Anany. 2003. *Introduction to the design & analysis of algorithms -3rd ed.* USA: Adison-Wesley.

Munir, Rinaldi. Slide kuliah "Algoritma *Decrease & Conquer*". *http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Decrease-and-Conquer-2021-Bagian1.pdf*. Diakses *online* pada 27 Februari 2021.

Munir, Rinaldi. Slide kuliah "Tugas Kecil 1 IF2211". *https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Tugas-Kecil-2-(2021).pdf*. Diakses *online* pada 27 Februari 2021.