**TUGAS KECIL 2 IF2211 STRATEGI ALGORITMA**

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

Oleh

Muhammad Fawwaz Naabigh - 13519206



**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2020/2021**

**Algoritma *Topological Sort* dan kaitannya dengan pendekatan *Decrease and Conquer***

*Topological Sort* pada *Directed Acyclic Graph* (DAG) adalah algoritma pengurutan linear dari beberapa simpul, sehingga setiap busur berarah u v, dengan simpul u muncul sebelum simpul v berdasarkan urutan. *Topological Sort* hanya dapat digunakan pada DAG, selain itu, tidak dimungkinkan digunakan *Topological Sort.* Simpul pertama dalam algoritma ini adalah simpul dengan derajat masuk berjumlah 0, v1. Kemudian, derajat simpul lain yang memiliki busur masuk berupa simpul yang berderjat 0 tadi (v1) akan dikurangi sebanyak 1. Setelah pengurangan tersebut, akan didapat derajat baru untuk setiap simpul yang belum dipilih. Lalu, ulangi lagi langkah pertama, yaitu pilih simpul berderajat 0 dan kurangi derajat simpul lain yang memiliki busur masuk berupa simpul yang terpilih hingga semua simpul terpilih. Urutan pemilihan simpul tersebutlah yang menjadi dasar pengurutan algoritma ini.

Di sisi lain, *Decrease and Conquer* adalah algoritma dengan mereduksi persoalan menjadi beberapa sub-persoalan lain yang lebih kecil dengan syarat hanya satu sub-persoalan yang akan diproses selanjutnya. Hal tersebut berbeda denan *Divide and Conquer* yang memproses seluruh sub-persoalan dan kemudian setiap solusi dari sub-persoalan akan digabung. Algoritma *Decrease and Conquer* terdiri dari dua tahap. Sesuai dengan namanya, yaitu tahap *decrease* dan tahap *conquer.* Pada tahapan *decrease* permasalahan akan direduksi menjadi sub-persoalan dan pada tahap *conquer* persoalan tersebut kemudian diproses satu persatu.

Dalam *topological sort*¸ proses *decrease* terjadi pada saat penghapusan simpul terpilih dan *conquer* terjadi pada saat pemilihan simpul dengan derajat masuk = 0. Proses tersebut dilakukan seluruh simpul memiliki derajat masuk = 0, yang berarti semua simpul telah ditaklukkan (*conquered*). Pada program yang ada di bawah ini, graf direpresentasikan dalam sebuat data bertipe *dictionary*, dengan key yang menjadi sebuah simpul dan value merupakan sebuah list yang beranggotakan setiap simpul masuk yang sesuai dengan simpul key. Proses *decrease* terjadi saat sebuah anggota list dari sebuah value milik key tertentu dihapus, yaitu berupa simpul yang dipilih pada proses *conquer*. Sedangkan, proses *conquer* terjadi saat pemilhan key yang memiliki value berupa list kosong dari *dictionary*, key dengan value list kosong menandakan bahwa simpul tersebut tidak memiliki simpul masuk atau dengan kata lain berderajat nol.

***Source Code***

|  |
| --- |
| # 13519206 - Muhammad Fawwaz Naabigh  import re, time, os  def bacaFile(namafile):  '''  Membaca masukan dari file.txt lalu membaca tiap baris  dan mengembalikan array matakuliah yang juga berisi sekumpulan array (sub-array)  dengan elemen pertama adalah kode kuliah dan sisanya adalah kuliah prasyarat.  Sub-array tersebut adalah data per baris yang dibaca dari file.  [['kode kuliah1', 'prasyarat1', 'prasyarat2'], ['kode kuliah2']]  '''  try:  directory = os.path.abspath(f'../test/{namafile}.txt')  f = open(directory, "r")  lines = f.readlines()  courses = [] # list yang menampung sub-array matkul  for line in lines:  course = [] # list yang akan ditampung (sub-array) pada list courses. berisi matkul dan prasyarat  # Menghapus karakter yang tidak diperlukan/yang dapat menganggu program. seperti quote ('') dsb.  x = re.sub(r'[^\w\s]', '', line)  y = re.sub(r'\n', '', x)  if y!="":  y = y.split(' ')  for matkul in y:  course.append(matkul)  courses.append(course)  return courses  except:  # File gagal dibaca  printFailed()  return quit()  def hitungPrasyarat(arrayOfCourses):  '''  Nilai yang dikembalikan adalah dictionary dengan key berupa nama course (simpul)  dan value adalah prasyarat (busur masuk).  {'kode kuliah1': ['prasyarat1', 'prasyarat2'], 'kode kuliah2': []}    Bila direpresentasikan dengan graf.  - key -> simpul  - value -> simpul yang masuk (busur) ke simpul key. berupa array of course ['C1', 'C2', 'C6'].  value berupa array kosong ([]) berarti course tersebut tidak memiliki prasyarat.  '''  courses = {}  for course in arrayOfCourses:  matkul = course[0]  prasyarat = course[1:]  courses[matkul] = prasyarat  return courses  def topSort(coursesGraph):  '''  Nilai yang dikembalikan adalah array yang berisi subarray  dengan index array adalah semester - 1 dan subarray berisi course yang dapat diambil pada semester tersebut.    [['C1', 'C2'], ['C3']]  Misal, index ke-0 berisi ['C1', 'C2'],  berarti pada semester 1 mata kuliah yang dapat diambil adalah course C1 dan C2.  Begitu seterusnya.  '''  coursePerSemester = [] # Berisi course dengan semester = index+1  while(len(coursesGraph)!=0):  selectedCourse = [] # Menampung course (simpul) dengan busur masuk = 0  for course, prereq in coursesGraph.items():  if(len(prereq)==0):  selectedCourse.append(course)  coursePerSemester.append(selectedCourse)  # Menghapus course (simpul) dengan prasyarat (busur masuk) = 0  for c in selectedCourse:  del coursesGraph[c]  # Menghapus busur yang keluar dari selectedCourse (course (simpul) dengan busur masuk = 0)  for course, prereq in coursesGraph.items():  for matkul in selectedCourse:  if matkul in prereq:  prereq.remove(matkul)  # print(dictOfCourses)  return coursePerSemester  def printFailed():  '''  Dipanggil jika masukan salah,  bisa disebabkan jika file tidak terbaca (entah user salah input dan sebagainya)  atau jika isi file kosong.  '''  print("\n\tMasukan tidak terbaca.")  print("\tPastikan file Anda:")  print("\t - Berekstensi .txt")  print("\t - Memiliki nama file yang sesuai dengan yang Anda input. Nama file ditulis tanpa .txt")  print("\t - Berada pada directory yang sama dengan program ini")  print("\t - Tidak berisi file kosong dan memiliki format yang benar")  print("\t - Contoh format yang benar adalah")  print('''\t\tC1, C3.  C2, C1, C4.  C3.  C4, C1, C3.  C5, C2, C4.  ''')  print("\t Tiap baris berisi course dan diikuti prerequisite dari course tersebut")  print()  input("Press ENTER to exit")  def printHasil(arrayOfCourses):  '''  Dipanggil untuk menampilkan hasil ke layar  '''  if(len(arrayOfCourses) == 0):  printFailed()  else:  print("\n\tMata kuliah yang dapat Anda ambil tiap semester adalah:")  for i in range(len(arrayOfCourses)):  print(f'\tSemester {i+1} : {", ".join(arrayOfCourses [i])}')  print()  input("Press ENTER to exit")  a = input("Masukkan nama file Anda (tanpa ekstensi .txt): ")    topsort = topSort(hitungPrasyarat(bacaFile(a)))  printHasil(topsort) |

***Screenshot* hasil percobaan**

|  |  |
| --- | --- |
| Masukan | Luaran |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

*Link Source Code*

[*https://github.com/fwznbg/strategiAlgoritma/tree/main/Tucil%202\_13519206*](https://github.com/fwznbg/strategiAlgoritma/tree/main/Tucil%202_13519206)

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