Implementasi Algoritma *Greedy* Dalam Menjadwalkan Pengerjaan Tugas

Fawwaz Anugrah Wiradhika Dharmasatya - 13520086
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13520086@std.stei.itb.ac.id

***Abstract*—Hampir di semua mata kuliah terdapat tugas, baik individu atau kelompok, yang perlu untuk dikerjakan mahasiswa agar mendapatkan nilai untuk lulus di mata kuliah tersebut. Namun, sering terdapat kasus beberapa mata kuliah yang diikuti mahasiswa menerbitkan tugas di saat yang bersamaan dengan *deadline* yang saling berdekatan sehingga menimbulkan permasalahan tugas mana yang harus dikerjakan terlebih dahulu. Oleh karena itu, pada makalah ini akan dibaha tentang strategi penjadwalan pengerjaan tugas menggunakan algoritma *greedy*.**

Keywords—greedy*, penjadwalan, tugas, *deadline

I. PENDAHULUAN

Mahasiswa sebagai bagian dari perguruan tinggi, memiliki tiga kewajiban yang terdapat dalam tridharma perguruan tinggi, yakni pendidikan dan pengajaran, penelitian dan pengembangan, serta pengabdian kepada masyarakat. Dalam rangka memenuhi unsur pendidikan dan pengajaran, mahasiswa mengikuti proses belajar di lingkungan kampus dengan mengikuti mata kuliah yang disediakan, baik yang wajib maupun pilihan.

Setiap mata kuliah yang ada memiliki bobot SKS nya masing-masing. Satu SKS di perkuliahan berisi 50 menit per minggu untuk pertemuan tatap muka, 60 menit per minggu untuk penugasan terstruktur baik sinkron maupun asinkron, serta 60 menit per minggu bagi mahasiswa untuk melakukan studi mandiri dalam memahami materi mata kuliah tersebut. Pada unsur penugasan ini, dosen pengampu mata kuliah bisa memberikan tugas baik individu maupun berkelompok agar mahasiswa bisa lebih memahami dan melakukan eksplorasi pada materi yang dibahas pada mata kuliah yang diajarkan.

Pada setiap program studi, terdapat beberapa mata kuliah yang wajib diambil oleh mahasiswa di tiap semester. Selain mata kuliah wajib tersebut, mahasiswa juga bisa mengambil mata kuliah lain seperti mata kuliah umum dan mata kuliah pilihan.

Masalah mulai muncul ketika beberapa atau bahkan semua mata kuliah yang diikuti mahasiswa merilis tugas di waktu yang berdekatan serta memiliki tenggat waktu pengumpulan yang berdekatan juga. Hal ini membuat mahasiswa perlu untuk menjadwalkan kapan mengerjakan tiap-tiap tugas agar bisa selesai tepat pada waktunya.

Gambar 1.1 Contoh Kasus Tugas Kuliah yang Menumpuk Serta Memiliki Tenggat Waktu (*Deadline*) yang Berdekatan

sumber: Dokumentasi Pribadi

Setiap tugas memiliki tingkat kesulitannya masing-masing, yang dipengaruhi juga dengan jenis, banyaknya tugas, pemahaman mahasiswa akan materi, serta koordinasi dan kemampuan anggota kelompok pada kasus tugas kelompok. Oleh karena itu, pada makalah ini akan dibahas suatu strategi untuk menyusun jadwal pengerjaan tugas-tugas kuliah menggunakan algoritma *greedy* dengan memperhatikan *deadline* dan tingkat kesulitan pengerjaan tugas.

II. LANDASAN TEORI

A. Tugas Mata Kuliah

Tugas dalam Kamus Besar Bahasa Indonesia adalah suatu hal yang wajib dikerjakan atau yang ditentukan untuk dilakukan.

Mata Kuliah adalah pelajaran yang dipelajari di tingkat universitas. Setiap mata kuliah memiliki bobot nilai yang disebut SKS (Satuan Kredit Semester). Di Institut Teknologi Bandung, satu SKS perkuliahan berisi 50 menit per minggu untuk pertemuan tatap muka, 60 menit per minggu untuk penugasan terstruktur baik sinkron maupun asinkron, serta 60 menit per minggu bagi mahasiswa untuk melakukan studi mandiri dalam memahami materi mata kuliah tersebut.

B. Algoritma *Greedy*

Algoritma *greedy* merupakan suatu algoritma untuk menyelesaikan permasalahan, terutama permasalahan optimasi, dengan cara memecahkan persoalan secara langkah-per-langkah, sehingga pada setiap langkah akan diambil pilihan yang terbaik (optimum lokal) yang dapat diperoleh saat itu dari berbagai pilihan pada langkah tersebut tanpa memperhatikan konsekuensi ke depannya dengan harapan bahwa pilihan tersebut akan mengarah ke solusi optimum global.

Algoritma *greedy* mempunyai elemen-elemen yang menjadi penyusun dari algoritma ini, antara lain:

1. Himpunan kandidat (C). Himpunan ini berisi kandidat yang dapat dipilih pada setiap langkah algoritma *greedy*. Contoh: Himpunan simpul/sisi dalam graf, *job*, *task*, koin, benda, karakter, dan sebagainya.
2. Himpunan solusi (S). Himpunan ini berisi elemen himpunan kandidat yang sudah dipilih dalam membentuk solusi akhir. Contoh: Himpunan *job* yang dipilih pada permasalahan penjadwalan pekerjaan.
3. Fungsi solusi. Fungsi ini bertujuan untuk memastikan apakah elemen-elemen himpunan kandidat yang sudah dipilih sudah memberikan solusi bagi permasalahan.
4. Fungsi seleksi. Fungsi ini bertujuan untuk memilih elemen himpunan kandidat pada setiap langkah berdasarkan strategi *greedy* tertentu.
5. Fungsi kelayakan. Fungsi ini bertujuan untuk memeriksa apakah kandidat yang dipilih oleh fungsi seleksi dapat dimasukkan ke himpunan solusi.
6. Fungsi objektif. Fungsi ini bertujuan untuk memaksimumkan atau meminimumkan nilai dari himpunan solusi yang akan dibentuk.

C. Permasalahan Penjadwalan Pekerjaan

Salah satu permasalahan yang bisa diselesaikan dengan algoritma *greedy* adalah permasalahan penjadwalan pekerjaan (*job scheduling*). Pada permasalahan ini, terdapat n buah pekerjaan yang akan dikerjakan sebuah sistem/mesin. Mesin hanya bisa mengerjakan satu buah pekerjaan untuk suatu selang waktu dan setiap pekerjaan memiliki tenggat waktu pengerjaan (*deadline*) d, dengan $d \geq 0$. Suatu job i akan memberikan keuntungan sebesar p_i , jika dan hanya jika pekerjaan tersebut diselesaikan tidak melewati tenggat waktunya. Tujuan dari permasalahan ini adalah bagaimana cara memilih pekerjaan-pekerjaan yang akan dikerjakan mesin sehingga keuntungan yang diperoleh dari pengerjaan itu maksimum.

Elemen-elemen algoritma *greedy* dari permasalahan ini adalah:

1. Himpunan kandidat-nya adalah semua perkerjaan yang dapat dipilih oleh mesin.
2. Himpunan solusi-nya adalah daftar pekerjaan yang telah dipilih untuk dikerjakan oleh mesin agar memperoleh keuntungan yang maksimum.
3. Fungsi solusi memeriksa apakah masih ada pekerjaan yang bisa dipilih oleh fungsi seleksi. Jika sudah tidak ada, maka solusi telah ditemukan.
4. Fungsi seleksi akan memilih pekerjaan yang memiliki keuntungan terbesar.
5. Fungsi kelayakan memeriksa apakah kandidat yang dipilih oleh fungsi seleksi dapat dimasukkan ke himpunan solusi. Kandidat yang dipilih tidak boleh memiliki jangka waktu pengerjaan yang tumpeng tindih dengan pekerjaan lain di himpunan solusi serta bisa diselesaikan sebelum *deadline*.
6. Fungsi objektif akan menghitung total pendapatan dari himpunan solusi.

III. ANALISIS DAN PEMBAHASAN

A. Transformasi Permasalahan ke Algoritma Greedy

Untuk menyelesaikan permasalahan penjadwalan pengerjaan tugas, digunakan strategi yang mirip dengan permasalahan penjadwalan pekerjaan. Bedanya adalah fungi objektif yang digunakan adalah sebanyak mungkin tugas yang bisa diselesaikan serta fungsi seleksinya memilih pekerjaan dengan bobot terbesar. Bobot ini diperngaruhi oleh tiga faktor, yakni seberapa dekat dengan *deadline*, tingkat kesulitan tugas, serta kemampuan atau seberapa paham mahasiswa dengan topik dari tugas yang diberikan.

Setiap tugas dibagi kedalam beberapa subtugas. Subtugas ini merepresentasikan bagian dari tugas yang bisa dikerjakan kurang dari sehari. Setiap tugas bisa memiliki jumlah subtugas yang berbeda-beda, tergantung bagaimana mahasiswa membaginya. Sebagai contoh, latihan soal bisa memiliki 1 hingga 2 subtugas sedangkan tugas besar bisa memiliki 5 atau lebih sub tugas. Tujuan pembagian tugas menjadi subtugas di penyelesaian permasalahan ini dikarenakan beberapa tugas sangat kompleks dan membutuhkan waktu lebih dari sehari untuk diselesaikan. Mahasiswa, selaku pihak yang mengerjakan tugas, juga mempunyai kegiatan lain yang perlu dilakukan disamping mengerjakan tugas, seperti makan, ke kamar mandi, tidur, beribadah, kuliah, olahraga, dan sebagainya. Karena itu, dalam sehari, waktu efektif untuk mengerjakan tugas tidak akan tepat 24 jam. Jika tidak dibagi ke dalam subtugas, tugas-tugas kompleks seperti tugas besar tidak

akan diambil oleh fungsi seleksi di algoritma penjadwalan karena tugas tersebut umumnya membutuhkan waktu total pengerjaan lebih dari sehari.

Elemen-elemen algoritma *greedy* untuk penyelesaian kasus ini antara lain:

1. Himpunan kandidat, yang berisi daftar subtugas, yang berisi nama subtugas, deadline tugas, tingkat kesulitan subtugas, kemampuan/pemahaman mahasiswa terhadap materi substugas, serta perkiraan alokasi pengerjaan subtugas.
2. Himpunan solusi, yang berisi jadwal pengerjaan subtugas yang telah disusun pada rentang waktu tertentu.
3. Fungsi solusi, yang memeriksa apakah masih ada subtugas yang bisa dipilih oleh fungsi seleksi. Jika sudah tidak ada, maka solusi telah ditemukan.
4. Fungsi seleksi, yang akan memilih tugas dengan bobot terbesar. Bobot tersebut dihitung dari kedekatan dengan *deadline*, tingkat kesulitan, serta kemampuan mahasiswa dengan rumus sebagai berikut:

   Bobot = 100/(*deadline*) + TK + (5-SK) (1)

   Keterangan:

   - Bobot: bobot dari subtugas.
   - *deadline*: waktu menuju deadline dari selang waktu yang sekarang diperiksa.
   - TK: tingkat kesulitan. Tingkat kesulitan pengerjaan subtugas menurut mahasiswa. Berada dalam rentang 1-10.
   - SK: *skill*, kemampuan/pemahaman mahasiswa dalam mengerjakan subtugas, Berada pada skala 0-5.
5. Fungsi kelayakan, fungsi ini memeriksa apakah subtugas yang dipilih oleh fungsi seleksi bisa dikerjakan pada selang waktu yang tersedia.
6. Fungsi objektif akan menghitung total subtugas yang bisa dikerjakan.

Pseudocode dari algoritma ini adalah:

```
type Subtask : <nama:String,deadline: Date,tingkat_kesulitan:integer,skill:integer,
waktu_pengerjaan:integer,bobot:float>
type Date: < Tahun: integer, Bulan: integer[1..12], hari: integer[1..31]. jam: integer[0..23], menit:
integer[0..23], detik: integer[0..23] >
type IntegerDay: <tanggal: Date, alokasi_waktu[0..24]: integer>

function scheduleTask(Himpunan_Kandidat: Array of Subtask, Alokasi: Array of IntegerDay)→
Array of Array of Subtask
{ Fungsi yang menerima masukan berupa himpunan kandidat dan alokasi waktu efektif
mahasiswa dalam mengerjakan tugas tiap hari dan mengembalikan jadwal pengerjaan tugas }
Deklarasi
HS: array of array of Subtask
i , j: integer
exist: boolean
Candidate: Subtask
JadwalHari: array of Subtask
Algoritma
HS ← { }
i traversal [0..Himpunan_Kandidat.length]
        Himpunan_Kandidat[i].bobot ← 0
while(i< Alokasi.length) do
        exist ← True
        JadwalHari←{}
        { cari bobot untuk setiap subtask}
        j traversal [0..Himpunan_Kandidat.length]
                Himpunan_Kandidat[j].bobot ←getScore(Himpunan_Kandidat[j],Alokasi[i])
        { Pada iterasi pertama, urutkan himpunan kandidat dari yang terbesar}
        if(i=0)then
                sortTask(Himpunan_Kandidat, False )
        while(exist)do
                { Pilih yang bobotnya terbesar dan memenuhi syarat}
                Candidate ← chooseCandidate(Himpunan_Kandidat,Alokasi[i].alokasi_waktu)
                if(Candidate = nil)then
                        { Sudah tidak ada kandidat yang memenuhi}
                        exist ← False
                else {Ada kandidat yang memenuhi}
                        JadwalHari← JadwalHari U { Candidate }
                        Alokasi[i].alokasi_waktu ← Alokasi[i].alokasi_waktu -
Candidate.waktu_pengerjaan
                        Himpunan_Kandidat ← Himpunan_Kandidat - Candidate
        Endwhile
        HS ← HS U {JadwalHari}
        i ← i+1
endwhile
→ HS
```

Gambar 3.1. *Pseudocode* Algoritma Utama Program

```
function getScore(subtask: Subtask, alokasi: IntegerDay)→float
{ Menghitung bobot dari subtask }
Deklarasi
sisa_waktu: integer
Algoritma
Sisa_waktu← getDuration(subtask.deadline,alokasi.tanggal)
→(100 / (sisa_waktu)) + subtask.tingkat_kesulitan + subtask.skill

function chooseCandidate(HK:Array of Subtask,alokasi_waktu:integer)→ Subtask
{ Memilih subtask dalam HK yang bobotnya terbesar serta nilai waktu_pengerjaan nya lebih
kecil atau sama dengan alokasi_waktu, Asumsi HK sudah terurut mengecil }
Deklarasi
i : integer
candidate: Subtask
Algoritma
i ← 0
while(i<HK.length)do
        candidate ← HK[i]
        if(candidate.waktu_pengerjaan ≤ alokasi_waktu)then
                →candidate
        else
                i ← i+1
endwhile
→nil

function getDuration(date1:Date,date2: Date)→integer
{ Fungsi bawaan bahasa pemrograman yang menghitung selisih waktu antara dua buah
tanggal, implementasi tergantung bahasa pemrograman }

procedure sortTask(Himpunan_Kandidat: array of Subtask )
{ Mengurutkan Himpunan_Kandidat berdasarkan atribut bobot, diurutkan secara mengecil
menggunakan quick sort }
```

Gambar 3.2. *Pseudocode* Fungsi Pembantu

Kompleksitas algoritma ini ditentukan oleh banyaknya subtugas dan banyaknya hari yang diperiksa. Misalkan banyaknya subtugas adalah **n** dan banyaknya hari adalah **m**. Pada *loop* luar akan dilakukan iterasi sebanyak m kali. Pada *loop* dalam akan dilakukan beberapa aksi sebagai berikut:

- Pemberian bobot pada setiap elemen himpunan kandidat. Pada kasus terburuk, pada setiap hari yang

diperiksa, akan dilakukan n kali pengulangan untuk pemberian bobot. Fungsi untuk memberikan bobot memiliki kompleksitas O(1), tidak terpengaruh oleh jumlah subtugas.

- Pengurutan himpunan kandidat berdasarkan bobotnya secara mengecil. Karena algoritma yang digunakan adalah quicksort, pada kasus terburuk akan memiliki kompleksitas O(n log n). Khusus untuk bagian ini hanya dilakukan pada iterasi pertama saja karena variabel yang berubah pada tiap iterasi dalam perhitungan bobot hanya selisih menuju *deadline*, yang mana bagian ini untuk setiap elemen akan saling berkurang secara linier dan tidak mengubah urutan bobot.
- Kalang untuk mencari subtugas yang bisa dikerjakan di suatu hari. Pada setiap iterasi akan dipanggil fungsi untuk mencari kandidat untuk dipilih, pada kasus terburuk akan memiliki kompleksitas O(n). Jika semua subtugas bisa dijadwalkan pada suatu rentang waktu m, maka total operasi pemanggilan fungsi yang dilakukan sebanyak (n+m) kali. n kali dari pemilihan subtugas yang akan dimasukkan, dan m kali dari pemanggilan fungsi yang Menghasilkan nil (tidak ada lagi subtugas yang bisa dikerjakan di hari tersebut). Total kompleksitas algoritmanya menjadi $O(n^2 + mn)$

Kompleksitas algoritma totalnya menjadi:

$=O(m)(O(n)+O(n \log n)+ O(n^2 + mn))$

$=O(m)O(n^2 + mn)$

$= O(mn^2+m^2n)$ (3)

Karena secara umum m lebih kecil dari n, maka variabel yang signifikan adalah n sehingga komplekitas algoritma akhirnya menjadi:

$O(n^2)$ (4)

B. Contoh Penyelesaian Kasus

Misalkan ada daftar subtugas seperti berikut:

Nama	*deadline*	kesulitan	*skill*	durasi(jam)
Task 1	24.05.2022	1	4	3
Task 2	23.05.2022	2	5	4
Task 3	26.05.2022	3	3	5
Task 4	01.06.2022	9	1	8
Task 5	02.06.2022	10	2	11
Task 6	29.05.2022	2	3	2
Task 7	26.05.2022	1	4	1
Task 8	25.05.2022	8	5	12
Task 9	27.05.2022	7	4	13
Task 10	28.05.2022	4	3	2
Task 11	29.05.2022	2	2	1
Task 12	30.05.2022	1	1	2
Task 13	31.05.2022	1	2	2
Task 14	31.05.2022	2	3	3

Tabel 3.1 Daftar Tugas yang Harus Dikerjakan

Alokasi waktu pengerjaan harian mulai dari tanggal 21 Mei 2022 hingga 31 Mei 2022:

Tanggal	Alokasi Pengerjaan (jam)
21 Mei 2022	10
22 Mei 2022	8
23 Mei 2022	16
24 Mei 2022	11
25 Mei 2022	13
26 Mei 2022	0
27 Mei 2022	2
28 Mei 2022	5
29 Mei 2022	4
30 Mei 2022	18
31 Mei 2022	3

Tabel 3.2 Alokasi Waktu Harian Untuk Mengerjakan Tugas

Langkah pemecahan masalah dengan algoritma *greedy*:

1. Langkah 1: Periksa untuk tanggal 21 Mei 2022
   - Alokasi Waktu: 10 Jam
   - Cari bobot awal

Nama	Bobot
Task 1	(100/(24-21))+1+(5-4)= 35.33
Task 2	(100/(23-21))+2+(5-5)= 52
Task 3	(100/(26-21))+3+(5-3)=25
Task 4	(100/((31+1)-21))+9+(5-1)=22.09
Task 5	(100/((31+2)-21))+10+(5-2)=21.33
Task 6	(100/(29-21))+2+(5-3)=16.5
Task 7	(100/(26-21))+1+(5-4)=22
Task 8	(100/(25-21))+8+(5-5)=33
Task 9	(100/(27-21))+7+(5-4)=24.67
Task 10	(100/(28-21))+4+(5-3)=20.29
Task 11	(100/(29-21))+2+(5-2)=17.5
Task 12	(100/(30-21))+1+(5-1)=16.11
Task 13	(100/(31-21))+1+(5-2)=14
Task 14	(100/(31-21))+2+(5-3)=14

Tabel 3.3 Bobot Awal Tiap Subtugas

   - Urutkan dari yang terbesar hingga terkecil

Nama	Bobot	durasi(jam)
Task 2	52	4
Task 1	35.33	3
Task 8	33	12
Task 3	25	5
Task 9	24.67	13
Task 4	22.09	8
Task 7	22	1

Task 5	21.33	11
Task 10	20.29	2
Task 11	17.5	1
Task 6	16.5	2
Task 12	16.11	2
Task 13	14	2
Task 14	14	3

Tabel 3.4 Subtugas yang sudah diurutkan menurun

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 10 jam. Dipilihlah Task 2 dengan bobot 52 dan durasi pengerjaan 4 jam. Task 2 dimasukkan ke himpunan solusi untuk tanggal 21 Mei 2022 (variabel JadwalHari). Task 2 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 4. Hasil akhir menjadi:

  Alokasi Waktu = 10-4 = 6

  JadwalHari = { Task 2}

- Cari lagi bobot tertinggi yang memiliki waktu pengerjaan ≤ 6 jam. Dipilihlah Task 1 dengan bobot 35.33 dan durasi pengerjaan 3 jam. Task 1 dimasukkan ke himpunan solusi untuk tanggal 21 Mei 2022(variabel JadwalHari). Task 1 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 3. Hasil akhir menjadi:

  Alokasi Waktu = 6-3 = 3

  JadwalHari = { Task 2, Task 1}

- Cari lagi bobot tertinggi yang memiliki waktu pengerjaan ≤ 3 jam. Dipilihlah Task 7 dengan bobot 22 dan durasi pengerjaan 1 jam. Task 7 dimasukkan ke himpunan solusi untuk tanggal 21 Mei 2022(variabel JadwalHari). Task 7 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 1. Hasil akhir menjadi:

  Alokasi Waktu = 3-1 = 2

  JadwalHari = { Task 2, Task 1, Task 7}

- Cari lagi bobot tertinggi yang memiliki waktu pengerjaan ≤ 2 jam. Dipilihlah Task 10 dengan bobot 20.29 dan durasi pengerjaan 2 jam. Task 10 dimasukkan ke himpunan solusi untuk tanggal 21 Mei 2022(variabel JadwalHari). Task 2 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 2. Hasil akhir menjadi:

  Alokasi Waktu = 2-2 = 0

  JadwalHari = { Task 2, Task 1, Task 7, Task 10}

- Karena waktu alokasi pengerjaan tersisa untuk tanggal 21 Mei 2022 sudah 0. Maka iterasi untuk tanggal ini sudah selesai dan dilanjutkan ke tanggal berikutnya.

2. Langkah 2: Periksa untuk tanggal 22 Mei 2022

- Alokasi Waktu: 8 Jam
- Cari bobot awal

Nama	Bobot
Task 3	(100/(26-22))+3+(5-3)=30
Task 4	(100/((31+1)-22))+9+(5-1)=23
Task 5	(100/((31+2)-22))+10+(5-2)=22.09
Task 6	(100/(29-22))+2+(5-3)=18.29
Task 8	(100/(25-22))+8+(5-5)=41.33
Task 9	(100/(27-22))+7+(5-4)=28
Task 11	(100/(29-22))+2+(5-2)=19.29
Task 12	(100/(30-22))+1+(5-1)=17.5
Task 13	(100/(31-22))+1+(5-2)=15.11
Task 14	(100/(31-22))+2+(5-3)=15.11

Tabel 3.5 Bobot Tiap Subtugas Pada Tanggal 22 Mei 2022

Nama	Bobot	durasi(jam)
Task 8	41.33	12
Task 3	30	5
Task 9	28	13
Task 4	23	8
Task 5	22.09	11
Task 11	19.29	1
Task 6	18.29	2
Task 12	17.5	2
Task 13	15.11	2
Task 14	15.11	3

Tabel 3.6 Data Subtugas Pada Tanggal 22 Mei 2022

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 8 jam. Dipilihlah Task 3 dengan bobot 30 dan durasi pengerjaan 5 jam. Task 3 dimasukkan ke himpunan solusi untuk tanggal 22 Mei 2022 (variabel JadwalHari). Task 3 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 5. Hasil akhir menjadi:

  Alokasi Waktu = 8-5 = 3

  JadwalHari = { Task 3}

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 3 jam. Dipilihlah Task 11 dengan bobot 19.29 dan durasi pengerjaan 1 jam. Task 11 dimasukkan ke himpunan solusi untuk tanggal 22 Mei 2022 (variabel JadwalHari). Task 11 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 1. Hasil akhir menjadi:

  Alokasi Waktu = 3-1 = 2

  JadwalHari = { Task 3, Task 11}

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 2 jam. Dipilihlah Task 6 dengan bobot 18.29 dan durasi pengerjaan 2 jam. Task 6 dimasukkan ke himpunan solusi untuk tanggal 22 Mei 2022 (variabel JadwalHari). Task 6 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 2. Hasil akhir menjadi:

  Alokasi Waktu = 2-2 = 0

  JadwalHari = { Task 3, Task 11, Task 6}

- Karena waktu alokasi pengerjaan tersisa untuk tanggal 22 Mei 2022 sudah 0. Maka iterasi untuk tanggal ini sudah selesai dan dilanjutkan ke tanggal berikutnya.

3. Langkah 3: Periksa untuk tanggal 23 Mei 2022
   - Alokasi Waktu: 16 Jam
   - Cari bobot awal

Nama	Bobot
Task 4	(100/((31+1)-23))+9+(5-1)=24.11
Task 5	(100/((31+2)-23))+10+(5-2)=23
Task 8	(100/(25-23))+8+(5-5)=58
Task 9	(100/(27-23))+7+(5-4)=33
Task 12	(100/(30-23))+1+(5-1)=19.29
Task 13	(100/(31-23))+1+(5-2)=16.5
Task 14	(100/(31-23))+2+(5-3)=16.5

Tabel 3.7 Bobot Tiap Subtugas Pada Tanggal 23 Mei 2022

Nama	Bobot	durasi(jam)
Task 8	58	12
Task 9	33	13
Task 4	24.11	8
Task 5	23	11
Task 12	19.29	2
Task 13	16.5	2
Task 14	16.5	3

Tabel 3.8 Data Subtugas Pada Tanggal 23 Mei 2022

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 16 jam. Dipilihlah Task 8 dengan bobot 58 dan durasi pengerjaan 12 jam. Task 8 dimasukkan ke himpunan solusi untuk tanggal 23 Mei 2022 (variabel JadwalHari). Task 8 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 12. Hasil akhir menjadi:

  Alokasi Waktu = 16-12 = 4

  JadwalHari = { Task 8}

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 4 jam. Dipilihlah Task 12 dengan bobot 19.29 dan durasi pengerjaan 2 jam. Task 12 dimasukkan ke himpunan solusi untuk tanggal 23 Mei 2022 (variabel JadwalHari). Task 12 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 2. Hasil akhir menjadi:

  Alokasi Waktu = 4-2 = 2

  JadwalHari = { Task 8, Task 12}

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 2 jam. Dipilihlah Task 13 dengan bobot 16.5 dan durasi pengerjaan 2 jam. Task 13 dimasukkan ke himpunan solusi untuk tanggal 23 Mei 2022 (variabel JadwalHari). Task 13 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 2. Hasil akhir menjadi:

  Alokasi Waktu = 2-2 = 0

  JadwalHari = { Task 8, Task 12, Task 13}

- Karena waktu alokasi pengerjaan tersisa untuk tanggal 23 Mei 2022 sudah 0. Maka iterasi untuk tanggal ini sudah selesai dan dilanjutkan ke tanggal berikutnya.

4. Langkah 4: Periksa untuk tanggal 24 Mei 2022
   - Alokasi Waktu: 11 Jam
   - Cari bobot awal

Nama	Bobot
Task 4	(100/((31+1)-24))+9+(5-1)=25.5
Task 5	(100/((31+2)-24))+10+(5-2)=24.11
Task 9	(100/(27-24))+7+(5-4)=41.33
Task 14	(100/(31-24))+2+(5-3)=18.29

Tabel 3.9 Bobot Tiap Subtugas Pada Tanggal 24 Mei 2022

Nama	Bobot	durasi(jam)
Task 9	41.33	13
Task 4	25.5	8
Task 5	24.11	11
Task 14	18.29	3

Tabel 3.10 Data Subtugas Pada Tanggal 24 Mei 2022

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 11 jam. Dipilihlah Task 4 dengan bobot 25.5 dan durasi pengerjaan 8 jam. Task 4 dimasukkan ke himpunan solusi untuk tanggal 24 Mei 2022 (variabel JadwalHari). Task 4 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 8. Hasil akhir menjadi:

  Alokasi Waktu = 11-8 = 3

  JadwalHari = { Task 4}

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 3 jam. Dipilihlah Task 14 dengan bobot 18.29 dan durasi pengerjaan 3 jam. Task 14 dimasukkan ke himpunan solusi untuk tanggal 24 Mei 2022 (variabel JadwalHari). Task 14 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 3. Hasil akhir menjadi:

  Alokasi Waktu = 3-3 = 0

  JadwalHari = { Task 4, Task 14}

- Karena waktu alokasi pengerjaan tersisa untuk tanggal 24 Mei 2022 sudah 0. Maka iterasi untuk tanggal ini sudah selesai dan dilanjutkan ke tanggal berikutnya.

5. Langkah 5: Periksa untuk tanggal 25 Mei 2022
   - Alokasi Waktu: 13 Jam
   - Cari bobot awal

Nama	Bobot
Task 5	(100/((31+2)-25))+10+(5-2)=25.5
Task 9	(100/(27-25))+7+(5-4)=58

Tabel 3.11 Bobot Tiap Subtugas Pada Tanggal 25 Mei 2022

Nama	Bobot	durasi(jam)
Task 9	58	13
Task 5	25.5	11

Tabel 3.12 Data Subtugas Pada Tanggal 25 Mei 2022

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 13 jam. Dipilihlah Task 9 dengan bobot 58 dan durasi pengerjaan 13 jam. Task 9 dimasukkan ke himpunan solusi untuk tanggal 25 Mei 2022 (variabel JadwalHari). Task 9 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 13. Hasil akhir menjadi:

Alokasi Waktu = 13-13 = 0

JadwalHari = { Task 9}

- Karena waktu alokasi pengerjaan tersisa untuk tanggal 25 Mei 2022 sudah 0. Maka iterasi untuk tanggal ini sudah selesai dan dilanjutkan ke tanggal berikutnya.

6. Langkah 6: Periksa untuk tanggal 26 Mei 2022
   - Alokasi Waktu: 0 Jam
   - Cari bobot awal

Nama	Bobot
Task 5	(100/((31+2)-26))+10+(5-2)=27.29

Tabel 3.13 Bobot Tiap Subtugas Pada Tanggal 26 Mei 2022

Nama	Bobot	durasi(jam)
Task 5	27.29	11

Tabel 3.14 Data Subtugas Pada Tanggal 26 Mei 2022

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 0 jam. Karena waktu alokasi pengerjaan tersisa untuk tanggal 26 Mei 2022 adalah 0. Maka iterasi untuk tanggal ini sudah selesai dan dilanjutkan ke tanggal berikutnya.

JadwalHari = { }

7. Langkah 7: Periksa untuk tanggal 27 Mei 2022
   - Alokasi Waktu: 2 Jam
   - Cari bobot awal

Nama	Bobot
Task 5	(100/((31+2)-27))+10+(5-2)=29.67

Tabel 3.15 Bobot Tiap Subtugas Pada Tanggal 27 Mei 2022

Nama	Bobot	durasi(jam)
Task 5	29.67	11

Tabel 3.16 Data Subtugas Pada Tanggal 27 Mei 2022

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 2 jam. Karena tidak ada subtugas tersisa yang memenuhi, maka iterasi untuk tanggal ini sudah selesai dan dilanjutkan ke tanggal berikutnya.

JadwalHari = { }

8. Langkah 8: Periksa untuk tanggal 28 Mei 2022
   - Alokasi Waktu: 5 Jam
   - Cari bobot awal

Nama	Bobot
Task 5	(100/((31+2)-28))+10+(5-2)=33

Tabel 3.17 Bobot Tiap Subtugas Pada Tanggal 28 Mei 2022

Nama	Bobot	durasi(jam)
Task 5	33	11

Tabel 3.18 Data Subtugas Pada Tanggal 28 Mei 2022

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 5 jam. Karena tidak ada subtugas tersisa yang memenuhi, maka iterasi untuk tanggal ini sudah selesai dan dilanjutkan ke tanggal berikutnya.

JadwalHari = { }

9. Langkah 9: Periksa untuk tanggal 29 Mei 2022
   - Alokasi Waktu: 4 Jam
   - Cari bobot awal

Nama	Bobot
Task 5	(100/((31+2)-29))+10+(5-2)=38

Tabel 3.19 Bobot Tiap Subtugas Pada Tanggal 29 Mei 2022

Nama	Bobot	durasi(jam)
Task 5	38	11

Tabel 3.20 Data Subtugas Pada Tanggal 29 Mei 2022

Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 4 jam. Karena tidak ada subtugas tersisa yang memenuhi, maka iterasi untuk tanggal ini sudah selesai dan dilanjutkan ke tanggal berikutnya.

JadwalHari = { }

10. Langkah 10: Periksa untuk tanggal 30 Mei 2022
    - Alokasi Waktu: 18 Jam
    - Cari bobot awal

Nama	Bobot
Task 5	(100/((31+2)-30))+10+(5-2)=46.33

Tabel 3.21 Bobot Tiap Subtugas Pada Tanggal 30 Mei 2022

Nama	Bobot	durasi(jam)
Task 5	46.33	11

Tabel 3.22 Data Subtugas Pada Tanggal 30 Mei 2022

- Pilih kandidat dengan bobot tertinggi yang memiliki waktu pengerjaan ≤ 18 jam. Dipilihlah Task 5 dengan bobot 46.33 dan durasi pengerjaan 11 jam. Task 5 dimasukkan ke himpunan solusi untuk tanggal 30 Mei 2022 (variabel JadwalHari). Task 5 dibuang dari daftar task tersedia. Kurangi alokasi pengerjaan sebesar 11. Hasil akhir menjadi:

Alokasi Waktu = 18-11=7

JadwalHari = { Task 5}

- Karena sudah tidak ada lagi subtugas yang tersedia, maka iterasi di tanggal ini selesai.

11. Langkah 11: Periksa untuk tanggal 31 Mei 2022
    - Alokasi Waktu: 3 Jam
    - Karena sudah tidak ada lagi subtugas yang belum dijadwalkan, maka himpunan solusi untuk tanggal 31 Mei 2022 kosong.

JadwalHari = { }

Hasil akhir penjadwalan:

Tanggal	Subtugas yang Dikerjakan
21 Mei 2022	Task 1, Task 2, Task 7, Task 10
22 Mei 2022	Task 3, Task 6, Task 11
23 Mei 2022	Task 8, Task 12, Task 13
24 Mei 2022	Task 4, Task 14
25 Mei 2022	Task 9
26 Mei 2022	-
27 Mei 2022	-
28 Mei 2022	-
29 Mei 2022	-
30 Mei 2022	Task 5
31 Mei 2022	-

Tabel 3.23 Jadwal Pengerjaan Tugas Pada Rentang Waktu 21 Mei 2022 Hingga 31 Mei 2022

IV. KESIMPULAN

Algoritma *greedy* merupakan suatu algoritma untuk memecahkan masalah optimasi dengan memilih opsi terbaik di tiap langkah sebagai optimum lokal dengan harapan pilihan tersebut akan mengarah ke optimum global. Masalah penjadwalan merupakan contoh implementasi algoritma *greedy*. Tugas kuliah yang banyak dengan *deadline* yang berdekatan membuat perlunya suatu strategi untuk menyusun jadwal pengerjaan tugas yang efisien. Algoritma *greedy* menjadi strategi yang tepat dalam menyusun jadwal pengerjaan tugas ini karena algoritma ini akan memilih tugas dengan melihat prioritasnya, dengan prioritas ini dilihat dari jarak ke *deadline*, tingkat kesulitan, serta pemahaman mahasiswa terhadap materi, sehingga memberikan prioritas tinggi untuk tugas yang sudah dekat dengan *deadline*, namun juga memberikan prioritas pengerjaan yang lumayan tidak terlalu jauh dari *deadline* untuk tugas yang sulit atau membutuhkan waktu untuk eksplorasi materi.

Meski sudah cukup bagus, algoritma penjadwalan tugas ini memiliki kekurangan yakni bergantung pada alokasi waktu yang disediakan mahasiswa setiap hari. Akibatnya, ada kasus tugas yang sulit dan membutuhkan waktu banyak namun baru mendapat jadwal pengerjaan dekat dengan *deadline* karena ada tugas lain yang jauh lebih dekat *deadline* nya serta alokasi waktu yang ada baru cukup untuk mengerjakan tugas saat mendekati *deadline*. Untuk kedepannya, program bisa ditingkatkan dengan memungkinkan adanya pengerjaan parsial, dimana subtugas yang ada bisa dipecah pengerjaannya di hari yang berbeda.

UCAPAN TERIMA KASIH

Puji syukur kepada Tuhan Yang Maha Esa, karena atas rahmat dan petunjuk-Nya penulis bisa menulis makalah ini. Penulis berterima kasih kepada semua pihak yang secara langsung maupun tidak langsung membantu dalam keberjalanan penulisan makalah ini. Penulis juga berterima kasih yang sebesar-besarnya kepada Bu Ulfa, Bu Masayu, serta Pak Rinaldi selaku dosen pengampu Mata Kuliah Strategi Algoritma atas bimbingan dan pengajaran selama satu semester ini yang sangat membantu penulis dalam menyelesaikan makalah ini.

REFERENSI

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag1.pdf

[2] https://ft.untidar.ac.id/tri-dharma/ , diakses pada tanggal 20 Mei 2022 Pukul 10.00 WIB

[3] KODE ETIK DAN PERATURAN KEMAHASISWAAN ITB, https://fi.itb.ac.id/wp-content/uploads/sites/298/2022/02/Peraturan-Rektor-ITB-tentang-Peraturan-Akademik-ITB-PTNBH-2021.pdf, diakses pada tanggal 20 Mei 2022 Pukul 10.08 WIB

[4] Saiful A. 2017. PERILAKU MAHASISWA JURUSAN ILMU PERPUSTAKAAN DALAM MENYELESAIKAN TUGAS-TUGAS MATA KULIAH. Skripsi. Fakultas Adab dan Humaniora. UIN Alauddin Makassar: Makassar.

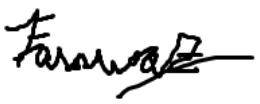
[5] https://kbbi.kemdikbud.go.id/ , diakses pada 20 Mei 2022 Pukul 17.45 WIB.

[6] https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Kompleksitas-Algoritma-2020-Bagian2.pdf

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 22 Mei 2022



Fawwaz Anugrah Wiradhika Dharmasatya 13520086