Penerapan Algoritma Greedy dalam Pembuatan Rekomendasi Jadwal Belajar Mandiri Mahasiswa

Gregorius Dimas Baskara - 13519190
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13519190@std.stei.itb.ac.id

***Abstrak*—Kegiatan perkuliahan mahasiswa, khususnya di Indonesia, tidak hanya meliputi pembelajaran pada jam yang telah ditentukan saja, namun juga meliputi kegiatan belajar mandiri di luar jam perkuliahan dengan durasi yang ditetapkan pada Sistem Kredit Semester (SKS) mata kuliah terkait. Meskipun demikian, SKS terkadang hanya berupa teori saja dan praktisnya, terdapat berbagai macam dinamika dalam masa perkuliahan yang menyebabkan tidak setaranya "tingkat kesibukan" dari waktu ke waktu. Akibatnya, sering terdapat beberapa tugas yang memiliki tenggat waktu yang sama (harus dikumpulkan pada waktu yang sama/berdekatan) ataupun kuis/ujian yang diselenggarakan pada hari yang sama. Meskipun seharusnya berbagai penilaian tersebut dapat dicicil, namun praktisnya terdapat banyak hal yang menyebabkan mencicil tersebut tidak atau tidak bisa dilakukan. Oleh karena itu, dibutuhkan suatu rekomendasi penjadwalan tertentu sehingga mahasiswa tetap mampu memperoleh nilai yang optimal di sela keterbatasan waktu belajar mandiri yang ada. Salah satu strategi yang dapat digunakan untuk membuat penjadwalan yang dimaksud adalah strategi Greedy. Melalui strategi greedy, maka dapat dibentuk suatu penjadwalan yang memaksimalkan nilai total yang didapat pada suatu semester dengan kendala/*constraint* waktu yang dimiliki mahasiswa.**

Keywords—jadwal, greedy, mahasiswa, perkuliahan, SKS

I. PENDAHULUAN

Kegiatan pendidikan dalam kancah perguruan tinggi Indonesia terdiri atas dua jenis kegiatan, kegiatan akademik dan kegiatan non akademik. Di dalam kegiatan akademik, dikenal konsep dan istilah Sistem Kredit Semester (SKS) yang digunakan untuk mengukur beban akademik yang diambil oleh mahasiswa pada suatu semester. Berdasarkan LTPB ITB, Satu SKS setara dengan upaya mahasiswa sebanyak 3 jam per minggu dalam 1 semester reguler yang meliputi 1 (satu) jam kegiatan interaksi akademik terjadwal dengan staf pengajar, 1 (satu) jam kegiatan terstruktur yang dilakukan dalam rangka kegiatan kuliah, dan minimal 1 (satu) jam kegiatan mahasiswa secara mandiri. Dalam realitasnya, pelaksanaan SKS tersebut juga terkendala oleh faktor-faktor tertentu yang menyebabkan kegiatan mahasiswa secara mandiri, baik itu mengerjakan tugas ataupun belajar untuk kuis atau ujian, menjadi terhambat.

Permasalahan ini salah satunya disebabkan dari tidak homogennya "tingkat kesibukan" dari waktu ke waktu. Pada saat "longgar", mahasiswa cenderung terlena dan bersantai-santai. Hal ini mengakibatkan kesibukan yang luar biasa pada saat mendekati masa pengumpulan tugas atau masa kuis/ujian. Di lain sisi, pemberitahuan tugas atau pelaksanaan kuis/ujian juga terkadang dilakukan pada saat yang bersamaan, sehingga waktu mulai persiapan mahasiswa untuk semua penilaian tersebut harus dilakukan pada saat bersamaan.

Selain itu, terkadang beban yang diberikan untuk suatu mata kuliah pun juga dinilai melebihi SKS yang telah ditetapkan. Dalam pelaksanaannya, "minimal 1 jam" untuk belajar mandiri dapat berarti lebih dari 3 jam waktu mahasiswa yang harus diluangkan untuk mengerjakan tugas kecil ataupun lebih dari 5 jam untuk tugas besar. Hal ini lah yang sering dikeluhkan oleh mahasiswa. Bukan saja membutuhkan waktu lebih untuk mengerjakan, namun waktu lebih yang digunakan untuk fokus pada suatu mata kuliah juga menyebabkan berkurangnya waktu untuk mata kuliah lain, sehingga menyebabkan performa yang tidak maksimal untuk setiap mata kuliah terdampak. Hal ini juga lantas berujung pada terdapatnya beberapa penilaian, baik tugas dan kuis yang terpaksa dimampatkan pada suatu waktu agar dapat selesai sebelum semester berakhir.

Terakhir, faktor padatnya kegiatan mahasiswa juga ikut menjadi penyebab dibutuhkannya penyusunan prioritas akademik. Sebagai peserta didik di dalam suatu institusi perguruan tinggi, mahasiswa juga dituntut agar dapat menjalankan pendidikan secara akademik dan non akademik secara seimbang. Pihak institusi pun selalu mendorong mahasiswanya untuk aktif berorganisasi, mengikuti kegiatan unit kemahasiswaan, dan mencari kegiatan magang. Hal ini lah kadang yang membuat mahasiswa terpaksa untuk menunda pekerjaan sehingga beban pekerjaan terasa tertumpuk pada suatu waktu. Meskipun pihak instansi mendorong mahasiswa untuk aktif, tidak diberikan kompensasi khusus untuk mahasiswa yang sedang menjalankan kegiatan non akademik tersebut sehingga mahasiswa harus cakap dalam menyeimbangkan kegiatan akademik dan non akademik agar sesuai dengan bobot kurikulum yang ada.

Untuk mengatasi persoalan yang dapat datang dari ketiga faktor tersebut, maka mahasiswa harus menyusun prioritas dengan baik. Salah satu strategi yang dapat digunakan dalam penyusunan rekomendasi jadwal belajar adalah strategi greedy. Strategi greedy dipilih karena efisien dalam menyelesaikan persoalan optimasi secara indikator waktu dan juga cukup efektif dalam pembentukan solusi yang sekurang-kurangnya

mendekati optimal (Pada beberapa kasus persoalan, solusi yang dihasilkan dapat dibuktikan selalu optimal).

Suatu mata kuliah dapat memiliki jumlah SKS yang berbeda dengan mata kuliah lainnya. Mata kuliah dengan SKS yang lebih besar akan memiliki pengaruh lebih besar pula pada perhitungan Indeks Prestasi (IP) pada suatu semester. Selain itu, masing-masing mata kuliah juga memiliki bobot penilaian masing-masing. Mata kuliah strategi algoritma misalkan, terdiri atas bobot penilaian tugas kecil (tucil), tugas besar (tubes), makalah, Ujian Tengah Semester (UTS), dan Ujian Akhir Semester (UAS). Bobot tugas besar, contohnya, akan lebih besar dibandingkan bobot tugas kecil, sehingga fokus pada pengerjaan tugas besar akan lebih "bernilai" ketimbang fokus pada pengerjaan tugas kecil. Berkaitan dengan itu pula, terdapat aspek penilaian yang dapat diselesaikan dengan lebih cepat dibanding penilaian yang lain. Semua faktor tersebut perlu dipertimbangkan agar nilai akhir yang didapatkan oleh mahasiswa dapat maksimal.

II. LANDASAN TEORI

A. *Persoalan Optimasi*

Persoalan optimasi merupakan suatu tugas atau pekerjaan untuk menentukan nilai-nilai variabel keputusan demi mendapatkan nilai tujuan yang paling bagus dengan tetap memenuhi kendala-kendala yang ada. Di dalam optimasi, hanya terdapat dua persoalan utama, yaitu maksimasi (maximization) dan minimasi (minimization). Salah satu contoh persoalan maksimasi adalah persoalan memilih aktivitas (activity selection problem) dan salah satu contoh persoalan minimasi adalah persoalan penukaran uang (coin exchange problem).

B. *Algoritma Greedy*

Algoritma greedy merupakan salah satu metode yang paling populer dalam menyelesaikan persoalan optimasi. Dari arti katanya sendiri, 'greedy' berarti 'rakus' atau 'tamak'. Hal ini selaras dengan prinsip algoritma greedy itu sendiri yang kurang lebih berbunyi "take what you can get now!". Beberapa sifat-sifat utama dari algoritma greedy dapat ditulis sebagai berikut:

1. Algoritma greedy merupakan algoritma yang memberikan solusi langkah per langkah atau *step by step* secara prosedural.
2. Pada setiap langkah, terdapat banyak opsi nilai-nilai variabel keputusan yang dapat diambil.
3. Pada setiap langkah, hanya dapat diambil satu keputusan yang dinilai terbaik berdasarkan pertimbangan saat itu dan tidak dapat melangkah mundur kembali ke langkah sebelumnya.
4. Keputusan terbaik yang diambil pada setiap langkah disebut sebagai optimum lokal (*local optimum*).
5. Dengan mengambil optimum lokal pada setiap langkah, diharapkan bahwa pada akhir persoalan, dapat dihasilkan solusi yang bersifat optimum global (meskipun juga dimungkinkan solusi yang tidak optimum global meski menggunakan algoritma greedy).

Selain sifat-sifat yang disebutkan sebelumnya, algoritma greedy juga memiliki elemen-elemen berikut ini yang merupakan ciri khas dari algoritma greedy:

1. Himpunan kandidat (C), yang berisi kandidat nilai variabel keputusan yang akan dipilih pada setiap langkah. Contohnya, himpunan koin-koin atau pecahan uang yang merepresentasikan nilai 1, 5, 10, 25, di mana paling sedikit mengandung satu koin untuk setiap nilai pada persoalan *coin exchange problem*.
2. Himpunan solusi (S), yang berisi kandidat yang sudah dipilih sebagai solusi pada setiap langkah. Contoh: himpunan koin-koin yang dipilih pada persoalan *coin exchange problem*.
3. Fungsi solusi yang berfungsi untuk memeriksa apakah himpunan kandidat yang dipilih sudah memberikan solusi yang diinginkan. Contoh: fungsi yang memeriksa apakah total nilai koin yang dipilih tepat sama jumlahnya dengan uang yang ditukarkan pada persoalan *coin exchange problem*.
4. Fungsi seleksi (*selection function*), yaitu fungsi yang bertugas untuk memilih kandidat berdasarkan strategi greedy tertentu (strategi ini bersifat heuristik). Contoh: fungsi yang bertugas memilih koin yang bernilai paling besar dari himpunan koin yang tersisa pada persoalan *coin exchange problem*.
5. Fungsi kelayakan (*feasible function*), yaitu fungsi yang bertugas untuk memeriksa apakah kandidat yang telah dipilih dapat dimasukkan ke dalam himpunan solusi. Contoh: fungsi yang memeriksa apakah uang hasil pertukaran melebihi jumlah uang awal atau tidak pada persoalan koin.
6. Fungsi objektif, yaitu fungsi yang bertugas untuk memaksimumkan atau meminimumkan (sesuai persoalan optimasi yang hendak dipecahkan). Contoh: jumlah koin yang bernilai minimum pada persoalan pertukaran koin.

Berikut ini merupakan salah satu skema umum strategi algoritma greedy di dalam bentuk *pseudo-code* yang dikutip dari catatan kuliah Dr. Rinaldi Munir:

```
function greedy(C : himpunan_kandidat) → himpunan_solusi
{ Mengembalikan solusi dari persoalan optimasi dengan algoritma greedy }
Deklarasi
  x : kandidat
  S : himpunan_solusi

Algoritma:
  S ← {}      { inisialisasi S dengan kosong }
  while (not SOLUSI(S)) and (C ≠ {} ) do
      x ← SELEKSI(C)      { pilih sebuah kandidat dari C}
      C ← C – {x}         { buang x dari C karena sudah dipilih }
      if LAYAK(S ∪ {x}) then      { x memenuhi kelayakan untuk dimasukkan ke dalam himpunan solusi
          S ← S ∪ {x}      { masukkan x ke dalam himpunan solusi }
      endif
  endwhile
  {SOLUSI(S) or C = {} }

  if SOLUSI(S) then    { solusi sudah lengkap }
    return S
  else
    write('tidak ada solusi')
  endif
```

Terdapat beberapa hal mengenai algoritma greedy yang perlu diingat sebagai berikut:

1. Optimum global yang dihasilkan sebagai solusi algoritma greedy **belum tentu** merupakan solusi terbaik. Hal ini

dikarenakan algoritma greedy yang tidak beroperasi secara menyeluruh (layaknya *exhaustive search*) dan karena terdapatnya beberapa pilihan fungsi seleksi yang berbeda dan harus dipilih yang terbaik untuk satu algoritma greedy. Contoh persoalan yang tidak tentu optimal apabila diselesaikan dengan algoritma greedy adalah persoalan pertukaran koin (*coin exchange problem*).

2. Pada beberapa persoalan tertentu, algoritma greedy akan selalu menghasilkan solusi optimal. Contohnya adalah penggunaan algoritma greedy pada persoalan minimasi waktu pada sistem. Meskipun demikian, hal tersebut harus dapat dibuktikan secara matematis meskipun algoritma greedy lebih menekankan sifat heuristik.
3. Meskipun belum tentu menghasilkan solusi optimal untuk setiap persoalan optimasi, namun kompleksitas algoritma greedy lebih kecil dibandingkan penyelesaian secara *brute force*. Hal ini membuat algoritma greedy efisien karena pada beberapa persoalan solusi optimum yang mutlak tidak selalu diutamakan dan algoritma greedy dapat memberikan solusi hampiran optimum dengan waktu yang lebih cepat.

C. *Persoalan Fractional Knapsack*

Persoalan Knapsack pada umumnya: Diberikan n buah objek dan sebuah knapsack dengan kapasitas bobot K. Setiap objek memiliki properti bobot (weight) w_i dan keuntungan (price) p_i. Tentukan cara memilih objek -objek yang dimasukkan ke dalam knapsack sedemikian sehingga diperoleh total keuntungan yang maksimal. Total bobot objek yang dimasukkan tidak boleh melebihi kapasitas knapsack.

Fractional Knapsack Problem adalah varian dari persoalan knapsack, namun solusi dapat dalam bentuk pecahan atau fraksi. Secara matematis, persoalan dapat dideskripsikan sebagai:

Maksimasi $F = \sum_{i=1}^{n} p_i x_i$

dengan kendala (*constraint*)

$$\sum_{i=1}^{n} w_i x_i \le K$$

yang dalam hal ini, $0 \le x_i \le 1, \; i = 1, 2, \ldots, n$

Gambar 2.1 – Deskripsi Persoalan Fractional Knapsack

Terdapat tiga strategi umum yang dapat digunakan untuk menyelesaikan persoalan ini dengan algoritma greedy:

1. Greedy by profit: Pada setiap langkah, pilih objek yang mempunyai keuntungan terbesar. Strategi ini mencoba memaksimumkan keuntungan dengan memilih objek yang paling menguntungkan terlebih dahulu.
2. Greedy by weight: Pada setiap langkah, pilih objek yang mempunyai berat (weight) yang paling kecil. Strategi ini fokus untuk memasukkan sebanyak mungkin objek dengan harapan mendapat keuntungan terbesar
3. Greedy by density. - Pada setiap langkah, knapsack diisi dengan objek yang mempunyai p_i/w_i terbesar. Strategi ini mencoba untuk menggabungkan keuntungan dari kedua strategi sebelumnya.

D. *Persoalan Jadwal Belajar Mandiri*

Persoalan yang dibahas adalah sebagai berikut: Diberikan berbagai penilaian, baik itu tugas/PR/Kuis/Ujian dari satu atau lebih mata kuliah yang diambil pada suatu semester, yang harus diselesaikan pada suatu waktu yang sama (tenggatnya sama) dan waktu ideal yang dibutuhkan untuk "menyelesaikan" masing-masing penilaian secara optimal (baik itu mengerjakan untuk tugas/PR ataupun belajar untuk kuis/ujian), maka buatlah suatu "jadwal" tertentu sehingga total kontribusi nilai yang diperoleh dari seluruh penilaian tersebut maksimum. Jadwal yang dimaksudkan adalah jumlah waktu yang harus diluangkan pada masing-masing penilaian dan perhitungan greedy didasarkan oleh dua faktor, yaitu jumlah SKS mata kuliah dari suatu penilaian dan bobot penilaian tersebut di dalam bobot total mata kuliah.

Sebagai contoh, misalkan pada tanggal 13 Mei 2021, terdapat tenggat pengumpulan tugas kecil strategi algoritma, di mana strategi algoritma merupakan mata kuliah dengan 3 SKS dan tugas kecil tersebut berbobot 5% dari seluruh penilaian pada mata kuliah strategi algoritma, serta idealnya dikerjakan dengan durasi 5 jam. Di saat bersamaan, terdapat pula tenggat pengumpulan tugas baca pemrograman berorientasi objek di mana pemrograman berorientasi objek merupakan mata kuliah dengan 3 SKS dan tugas tersebut memiliki bobot sebesar 1% dari seluruh penilaian, serta idealnya dikerjakan dengan durasi 2 jam. Selain itu, terdapat pula tenggat pengumpulan resume tugas resume Studium Generale di mana Studium Generale merupakan mata kuliah dengan 2 SKS dan tugas tersebut memiliki bobot sebesar 2% dari seluruh penilaian, serta idealnya dikerjakan dengan durasi 3 jam. Menghadapi tenggat tersebut, seorang mahasiswa X hanya memiliki 6 jam untuk mempersiapkan seluruhnya. Tentukan suatu jadwal tertentu dalam format di bawah sehingga X mendapatkan perolehan kontribusi nilai maksimum untuk semua penilaian tersebut.

Penilaian (i)	**Durasi (x_i)**
Tugas kecil strategi algoritma	a satuan waktu
Tugas baca pemrograman berorientasi objek	b satuan waktu
Resume Studium Generale	c satuan waktu

Tabel 2.1 – Tabel hasil penjadwalan

Pada persoalan ini, asumsikan bahwa waktu pengerjaan yang diluangkan pada suatu penilaian berkorelasi secara linear dengan hasil.

III. Pengembangan Persoalan Fractional Knapsack Untuk Memecahkan Persoalan Jadwal Belajar Mandiri

Di mana:
t : penjumlahan seluruh hasil penilaian setelah pembobotan
h_i : hasil untuk suatu penilaian setelah pembobotan
n : jumlah penilaian yang terlibat (tenggatnya sama)

Solusi persoalan ini merupakan pengembangan dari solusi persoalan Fractional Knapsack Problem, di mana persoalan dapat diselesaikan dengan 3 strategi berbeda, yaitu *greedy by price*, *greedy by weight*, *greedy by density*. *Price*, *weight*, *density* pada persoalan jadwal belajar mandiri tentu memiliki arti yang berbeda, namun memiliki konsep yang sama secara fundamental, sehingga persoalan jadwal belajar mandiri juga dapat diselesaikan dengan metode persoalan fractional knapsack. **(Catatan: *price* dalam persoalan ini merupakan istilah yang memiliki makna sama dengan *profit* yang menandakan "harga", "keuntungan", atau "nilai" dari suatu penilaian. Penilaian sendiri merupakan istilah tugas/kuis/ujian/PR yang dinilai).** Untuk menyesuaikan dengan konsep "price", "weight", dan "density" pada persoalan knapsack, maka diperlukan beberapa *adjustment* yang akan dijelaskan pada uraian-uraian berikut.

Berbeda dengan persoalan Fractional Knapsack Problem, perhitungan "*price*" di dalam persoalan jadwal belajar mandiri diartikan sebagai hasil pembobotan suatu penilaian maksimum. Dengan demikian, penilaian dengan "*price*" tertinggi adalah penilaian dengan perkalian bobot dalam mata kuliah dan SKS yang tertinggi, atau:

$$p = b * S$$

Di mana:
p : "price" suatu penilaian
b : bobot penilaian di dalam mata kuliah
S : SKS mata kuliah

Sementara itu, "*weight*" di dalam persoalan ini dapat diartikan sebagai waktu ideal dari untuk mempersiapkan suatu penilaian atau w. Dengan demikian, "*density*" dari suatu penilaian dapat dihitung berupa $\rho_i = p_i / w_i$.

Mengingat pada persoalan waktu pengerjaan yang diluangkan pada suatu penilaian berkorelasi secara linear dengan hasil, maka hasil h_i dari suatu penilaian setelah pembobotan merupakan fungsi dari x_i di mana x_i merupakan waktu yang diluangkan untuk penilaian tersebut sebagai berikut:

$$h = \frac{x}{w} * p$$

Di mana:
h : hasil untuk suatu penilaian setelah pembobotan
x : waktu yang diluangkan untuk suatu penilaian
w : waktu ideal untuk mempersiapkan suatu penilaian
p : "*price*" atau kontribusi maksimum suatu penilaian

Tujuan akhir dari persoalan ini adalah memaksimumkan nilai penjumlahan seluruh hasil penilaian setelah pembobotan atau maksimasi(t) dengan t:

$$t = \sum_{i=1}^{n} h_i$$

Perlu diingat bahwa layaknya persoalan Fractional Knapsack, maka berlaku pula:

$$\sum_{i=1}^{n} x_i \leq K$$

Di mana dalam persoalan ini, K merupakan waktu total yang dimiliki oleh mahasiswa dan $x_i \leq w_i$. Berdasarkan *nature* dari persoalan di mana akan dibuat suatu rekomendasi penjadwalan untuk mahasiswa yang "tidak sempat" mempersiapkan semua penilaian, maka persoalan akan difokuskan pada:

$$\sum_{i=1}^{n} w_i > K$$

IV. Implementasi Dan Analisis

A. *Implementasi*

Dalam implementasi pemecahan persoalan jadwal belajar mandiri, maka penulis membuat suatu aplikasi berbasis web (Web App) dengan menggunakan kakas React (Javascript) dan Bootstrap. Aplikasi yang dibuat tersebut dapat memecahkan persoalan dengan ketiga strategi yang telah disebutkan, *greedy by price*, *greedy by weight*, dan *greedy by density*.

Untuk setiap pilihan strategi, maka dilakukan *preprocessing* terlebih dahulu dengan mengurutkan data penilaian dari *price* terbesar (untuk *greedy by price*), *weight* terkecil (untuk *greedy by weight*), dan *density* terbesar (untuk *greedy by density*), seperti tampak pada potongan kode berikut:

```
// Sorting berdasarkan strategi
if(strategi === "price") {
    dataPenilaian.sort(function(a, b){return b.price - a.price});
}
else if(strategi === "weight") {
    dataPenilaian.sort(function(a, b){return a.weight - b.weight});
}
else {
    dataPenilaian.sort(function(a, b){return (b.price/b.weight) - (a.price/a.weight)});
}
```

Gambar 4.1 – Potongan kode preprocessing

Setelah dilakukan *preprocessing*, maka dilakukan tahap algoritma *greedy* dengan strategi yang dipilih. Untuk strategi *greedy by price*, maka waktu yang dimiliki oleh mahasiswa akan "secara rakus" digunakan sepenuhnya dari penilaian dengan *price* tertinggi. Apabila waktu yang diluangkan sudah mencapai waktu ideal dari penilaian tersebut, maka waktu akan diluangkan

ke penilaian dengan *price* tertinggi berikutnya. Hal yang sama dilakukan pula pada kedua strategi lainnya dengan indikator *weight* dan *density*. Meskipun demikian, pada strategi *greedy by weight*, waktu diluangkan mulai dari *weight* terendah terlebih dahulu ke *weight* tertinggi. Dalam kode program, algoritma ini diimplementasikan sebagai berikut:

```
const handleGenerateSchedule = () => {
    let dataPenilaian = data;
    let tableData = [];
    let maxWeightAvailable = waktuMahasiswa;
    let profit = 0;
    let maxProfit = 0;

    for(let i=0;i<dataPenilaian.length;i++) {
        if(maxWeightAvailable >= Number(dataPenilaian[i].weight)) {
            profit = profit + dataPenilaian[i].price;
            tableData.push({penilaian: dataPenilaian[i].penilaian, durasi: dataPenilaian[i].weight});
            maxWeightAvailable = maxWeightAvailable - dataPenilaian[i].weight;
        }
        else {
            profit = profit + (maxWeightAvailable/dataPenilaian[i].weight) * dataPenilaian[i].price;
            tableData.push({penilaian: dataPenilaian[i].penilaian, durasi: maxWeightAvailable});
            maxWeightAvailable = 0;
        }
        maxProfit = maxProfit + dataPenilaian[i].price;
    }

    setTableData(tableData);
    setFinalScore((profit/maxProfit) * 100);
    setShowJadwal(true);
}
```

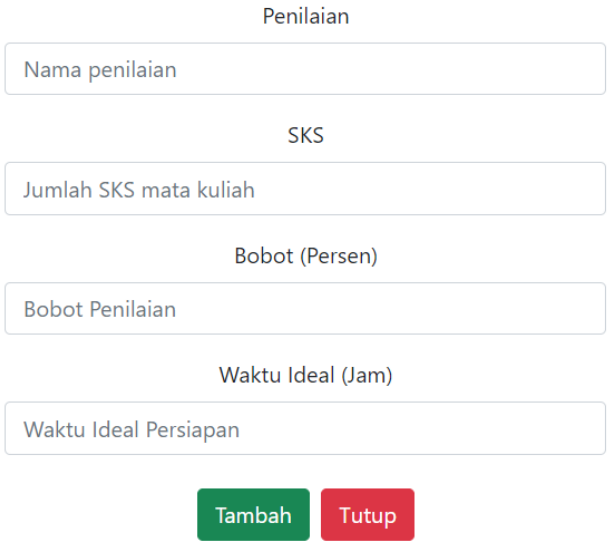
Gambar 4.2 – Potongan kode algoritma greedy

Hasil dari penjadwalan kemudian akan disajikan dalam bentuk Tabel 2.1 ke dalam layar. Selain itu, nilai F yang didapatkan dari strategi yang dipilih juga akan ditampilkan ke dalam layar dalam bentuk persentase.

B. Eksperimen dan Analisis

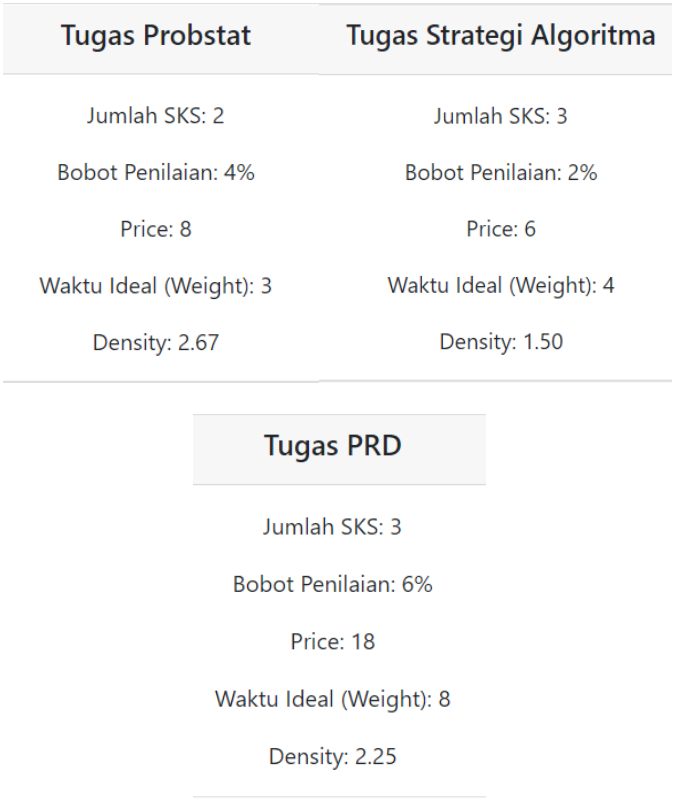
1. Eksperimen I dan interpretasi input

Sebagai salah satu eksperimen/*test case*, penulis memasukkan beberapa data penilaian melalui *User Interface* seperti pada Gambar 4.3.



Gambar 4.3 – *User Interface* untuk input data

Data-data yang dimaksudkan dapat dilihat sebagai berikut:

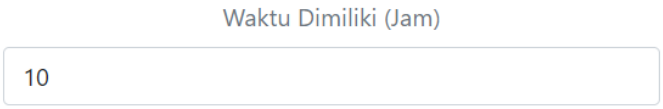


Gambar 4.4 – Data-data penilaian eksperimen I

Setelah memasukkan seluruh data, kemudian selanjutnya penulis akan melakukan eksperimen dengan ketiga strategi yang ada secara satu persatu. Untuk itu, strategi dapat dipilih dengan menekan *radio button* berikut:

Gambar 4.5 – *Radio button* strategi greedy

Penulis kemudian memasukkan waktu yang dimiliki oleh mahasiswa (K) dalam jam sebagai berikut:

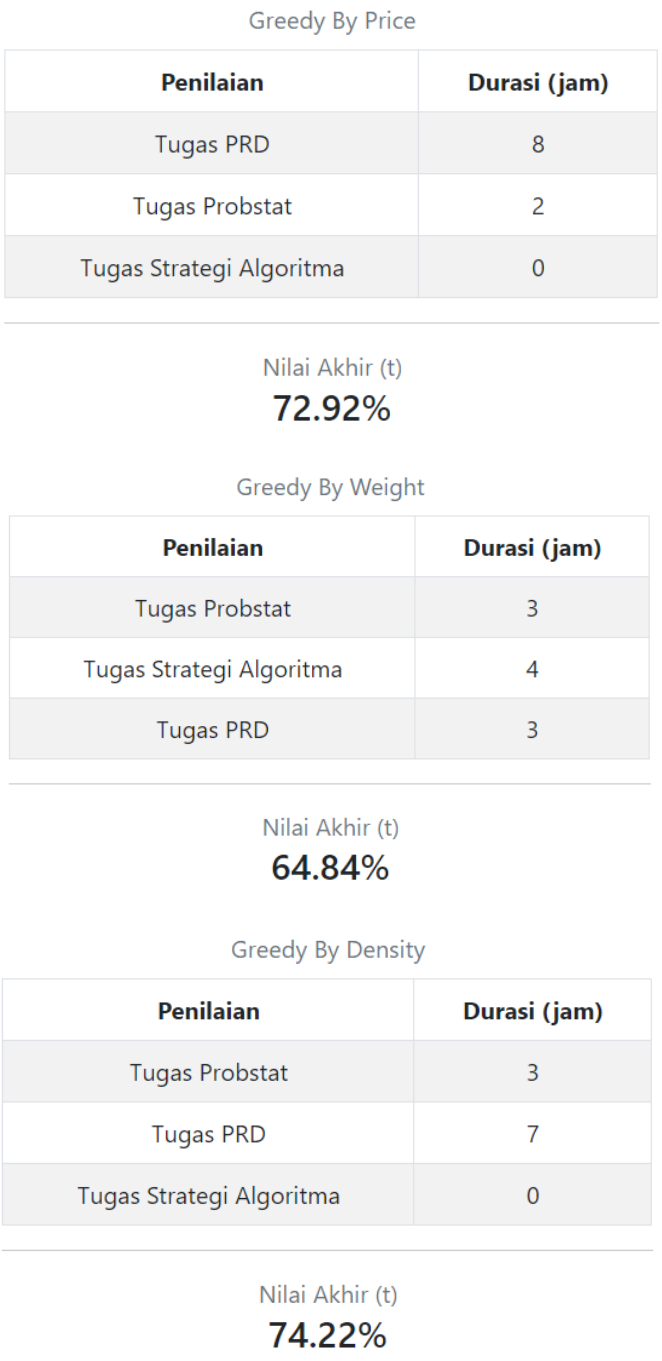


Gambar 4.6 – Nilai K untuk eksperimen I

Seluruh data yang telah dimasukkan di atas menggambarkan seorang mahasiswa yang memiliki tiga penilaian berbeda, yaitu Tugas Probstat, Tugas Strategi Algoritma, dan Tugas PRD, yang masing-masing memiliki SKS mata kuliah yang berbeda dan bobot penilaian di dalam

mata kuliah yang berbeda, serta secara ideal dapat diselesaikan dalam waktu yang berbeda, yang harus dikumpulkan pada tenggat waktu yang sama. Untuk mempersiapkan semua penilaian tersebut, mahasiswa tersebut hanya memiliki waktu 10 jam.

Setelah itu, penulis menekan "Buat Jadwal" dengan pengulangan untuk masing-masing strategi sehingga didapatkan hasil sebagai berikut:



Gambar 4.7 – Hasil eksperimen I

Melihat dari hasil yang ada pada eksperimen I, maka terdapat perbedaan nilai akhir yang dapat diperoleh pada masing-masing strategi. Dalam hal ini, nilai (t) paling tinggi diperoleh melalui strategi *greedy by density* dengan nilai 74.22%.

2. Eksperimen II

Melalui tahapan yang serupa dengan eksperimen I, dilakukan input data dan hasil sebagai berikut:

Gambar 4.8 – Data-data penilaian eksperimen II

Greedy By Weight

Penilaian	Durasi (jam)
Kuis Matematika Diskrit	4
Tugas Logika Komputasional	5
Kuis Aljabar Linier	1

Nilai Akhir (t)

68.18%

Greedy By Density

Penilaian	Durasi (jam)
Kuis Matematika Diskrit	4
Kuis Aljabar Linier	6
Tugas Logika Komputasional	0

Nilai Akhir (t)

70.91%

Gambar 4.9 – Hasil eksperimen II

Melihat dari hasil yang ada pada eksperimen II, maka terdapat perbedaan nilai akhir yang dapat diperoleh pada masing-masing strategi. Meskipun demikian, nilai akhir dengan strategi *greedy by price* dan *greedy by density* bernilai sama dan juga merupakan nilai tertinggi dengan nilai 70.91%.

3. Eksperimen III

Melalui tahapan yang serupa dengan eksperimen I dan II, dilakukan input data dan hasil sebagai berikut:

Kuis Kalkulus	Kuis Fisika Dasar
Jumlah SKS: 4	Jumlah SKS: 4
Bobot Penilaian: 6%	Bobot Penilaian: 7%
Price: 24	Price: 28
Waktu Ideal (Weight): 6	Waktu Ideal (Weight): 8
Density: 4.00	Density: 3.50

Tugas Pengenalan Komputasi
Jumlah SKS: 3
Bobot Penilaian: 4%
Price: 12
Waktu Ideal (Weight): 9
Density: 1.33

Waktu Dimiliki (Jam)

10

Gambar 4.10 – Data-data penilaian eksperimen III

Greedy By Price

Penilaian	Durasi (jam)
Kuis Fisika Dasar	8
Kuis Kalkulus	2
Tugas Pengenalan Komputasi	0

Nilai Akhir (t)

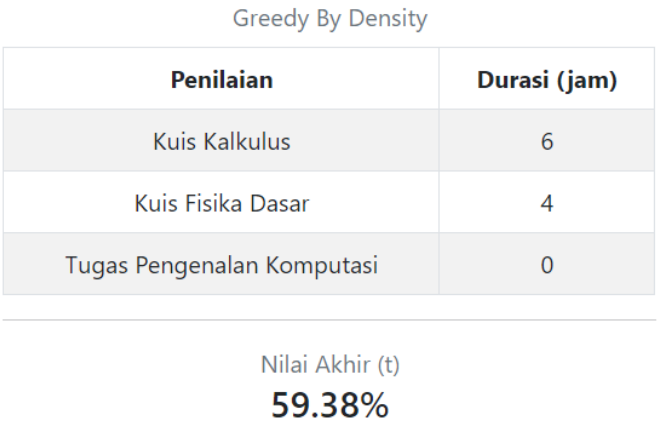
56.25%

Greedy By Weight

Penilaian	Durasi (jam)
Kuis Kalkulus	6
Kuis Fisika Dasar	4
Tugas Pengenalan Komputasi	0

Nilai Akhir (t)

59.38%



Gambar 4.11 – Hasil eksperimen III

Melihat dari hasil yang ada pada eksperimen II, maka terdapat perbedaan nilai akhir yang dapat diperoleh pada masing-masing strategi. Meskipun demikian, nilai akhir dengan strategi *greedy by weight* dan *greedy by density* bernilai sama dan juga merupakan nilai tertinggi dengan nilai 59.38%.

V. KESIMPULAN

Persentase nilai akhir (t) yang diperoleh dalam masing-masing strategi menandakan persentase nilai yang berhasil didapatkan dari seluruh penilaian tersebut. Karena waktu yang dimiliki mahasiswa (K), lebih kecil dibandingkan total waktu yang dibutuhkan untuk mempersiapkan (menyelesaikan) seluruh penilaian secara ideal (Sigma w_i), maka nilai 100% mustahil untuk dicapai. Tabel tersebut dapat dibaca dari kiri ke kanan sebagai penilaian dan durasi (jam) yang harus diluangkan untuk mempersiapkan (menyelesaikan) penilaian tersebut. Menariknya, terdapat beberapa rekomendasi di mana pada suatu penilaian, durasi (jam)-nya bernilai 0. Hal ini menandakan bahwa algoritma greedy menyarankan agar penilaian tersebut tidak perlu dikerjakan karena tidak *worthwhile.*

Pada setiap hasil eksperimen, terlihat suatu pola yaitu strategi *greedy by density* yang selalu menghasilkan nilai yang maksimal. Pada eksperimen II dan III, nilai strategi *greedy by density* memiliki besar yang sama dengan baik strategi *greedy by price* pada eksperimen II, maupun strategi *greedy by weight* pada eksperimen III. Meskipun demikian, terbukti bahwa baik strategi *greedy by price*, maupun strategi *greedy by weight* tidak dapat menjamin nilai yang optimal (seperti pada hasil eksperimen I).

Keunggulan strategi *greedy by* density pada setiap eksperimen ini sejalan dengan teori yang tertuang dalam buku "Introduction to the Design & Analysis of Algorithms" karya Anany Levitin yang menyebutkan bahwa strategi *greedy by density* akan selalu menjamin hasil yang optimal pada persoalan Fractional Knapsack. Karena persoalan jadwal belajar mandiri merupakan pengembangan dari persoalan Fractional Knapsack, maka teori tersebut juga berlaku pada persoalan ini. Dengan demikian, hasil optimal melalui strategi *greedy by density* tidak hanya dicapai secara kebetulan saja pada ketiga eksperimen ini, namun sudah terbukti secara matematis pada seluruh persoalan. Ketiga eksperimen tersebut mengafirmasi teori yang ada.

Oleh karena itu, maka dapat disimpulkan bahwa algoritma greedy dengan strategi *greedy by density* dapat digunakan untuk membuat rekomendasi jadwal yang optimal (terjamin) atau menyelesaikan permasalahan jadwal belajar mandiri.

VI. LINK VIDEO YOUTUBE

https://youtu.be/E0SUIPFThjk

VII. LINK DEPLOY WEBSITE

https://academic-scheduler.netlify.app/

VIII. UCAPAN TERIMA KASIH

Penulis ingin mengucapkan terima kasih sebesar-besarnya kepada beberapa pihak yang telah berperan besar dalam penyelesaian makalah ini :

1. Tuhan Yang Maha Esa karena berkat dan rahmat-Nya, makalah ini dapat terselesaikan dengan baik tanpa kurang satu bagian pun.
2. Dr. Ir. Rinaldi, M.T. sebagai dosen K4 dan seluruh tim dosen strategi algoritma yang telah berperan juga dalam kegiatan perkuliahan secara daring.
3. Teman-teman seangkatan karena telah memberikan dukungan selama perkuliahan sehingga penulis dapat menulis makalah ini dengan baik.

DAFTAR PUSTAKA

[1] LTPB ITB, diakses melalui https://ltpb.itb.ac.id/akademik/satuan-kredit-semester-sks/#:~:text=Satu%20SKS%20setara%20dengan%20upaya,jam%20kegiatan%20mahasiswa%20secara%20mandiri pada 10 Mei 2021 pukul 12.13.

[2] Rinaldi Munir, Diktat Kuliah IF2211 : Matematika Diskrit, Bandung : Program Studi Teknik Informatika Sekolah teknik Elektro dan informatika Institut Teknologi Bandung "Algoritma Greedy (Bagian 1), diakses melalui https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag1.pdf pada 10 Mei 2021 pukul 14.31.

[3] Rinaldi Munir, Diktat Kuliah IF2211 : Matematika Diskrit, Bandung : Program Studi Teknik Informatika Sekolah teknik Elektro dan informatika Institut Teknologi Bandung "Algoritma Greedy (Bagian 2), diakses melalui https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag2.pdf pada 10 Mei 2021 pukul 15.12.

[4] Rinaldi Munir, Diktat Kuliah IF2211 : Matematika Diskrit, Bandung : Program Studi Teknik Informatika Sekolah teknik Elektro dan informatika Institut Teknologi Bandung "Algoritma Greedy (Bagian 3), diakses melalui https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag3.pdf pada 10 Mei 2021 pukul 16.31.

[5] Anany Levitin, Introduction to the Design & Analysis of Algorithms, Addison-Wesley, 2003.

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, 11 Mei 2021

Gregorius Dimas Baskara