Pemanfaatan Algoritma Greedy untuk Penjadwalan Tugas Harian Pribadi dengan Prioritas dan Batasan Waktu

Indah Novita Tangdililing - 13523047
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: indahtangdililing@gmail.com, 13523047@std.stei.itb.ac.id

***Abstrak*—Manajemen tugas harian yang kompleks sering kali menurunkan produktivitas akibat proses penjadwalan manual yang tidak efisien. Makalah ini mengusulkan sebuah pendekatan komputasional menggunakan algoritma Greedy untuk otomatisasi penjadwalan pribadi. Model ini bekerja dengan merepresentasikan setiap tugas berdasarkan prioritas, batas waktu, dan durasi, lalu menggunakan formula heuristik hibrida untuk menghitung skor keputusan secara dinamis. Sebuah mekanisme penyaringan juga diimplementasikan untuk mengeliminasi tugas yang secara logis tidak mungkin diselesaikan tepat waktu.**

***Analisis melalui studi kasus menunjukkan bahwa algoritma ini sangat efektif dalam skenario harian yang umum, mampu menyeimbangkan berbagai faktor secara cerdas. Namun, turut diungkap keterbatasan utamanya, yaitu sifat "rabun jauh" yang dapat menjebak algoritma dalam solusi optimal lokal pada kondisi dengan batasan waktu yang sangat ketat.**

***Disimpulkan bahwa algoritma Greedy merupakan alat bantu yang praktis dan cepat, menawarkan pertukaran yang sepadan antara kecepatan komputasi dengan jaminan optimalitas lokal. Hal ini menjadikannya solusi untuk meningkatkan produktivitas harian, selama pengguna memahami batasannya.**

***Kata kunci*—Algoritma Greedy, Penjadwalan Tugas, Manajemen Waktu, Produktivitas Pribadi, Optimisasi Heuristik**

I. Pendahuluan

Di tengah era digital yang serba cepat, individu dihadapkan pada volume informasi dan tuntutan tugas yang belum pernah terjadi sebelumnya. Baik dalam konteks profesional, akademis, maupun personal, setiap hari kita dibanjiri oleh daftar pekerjaan yang harus diselesaikan: email yang harus dibalas, laporan yang harus ditulis, rapat yang harus dihadiri, dan kewajiban pribadi lainnya. Kompleksitas ini sering kali menciptakan sebuah paradoks pilihan, di mana banyaknya tugas justru melumpuhkan kemampuan kita untuk memutuskan harus memulai dari mana. Fenomena ini dikenal sebagai *analysis paralysis*, yang secara signifikan meningkatkan beban kognitif—yaitu jumlah usaha mental yang dibutuhkan untuk memproses informasi dan membuat keputusan.

Kondisi ini, jika tidak dikelola dengan baik, akan menimbulkan konsekuensi negatif yang signifikan. Penundaan atau prokrastinasi menjadi kebiasaan, tingkat stres meningkat, dan kualitas hasil kerja menurun akibat perhatian yang terfragmentasi. Lebih jauh lagi, ketidakmampuan mengelola waktu secara efektif dapat mengganggu keseimbangan antara pekerjaan dan kehidupan pribadi (*work-life balance*), yang merupakan fondasi dari kesehatan mental dan kesejahteraan secara umum.

Secara tradisional, manusia mengandalkan penjadwalan manual menggunakan agenda, catatan, atau sekadar ingatan. Namun, metode ini memiliki keterbatasan yang jelas. Prosesnya memakan waktu, sangat subjektif, dan sering kali dipengaruhi oleh bias kognitif atau kondisi emosional sesaat. Seseorang mungkin cenderung mengerjakan tugas yang mudah dan menyenangkan terlebih dahulu (bias urgensi semu), sambil mengabaikan tugas penting yang lebih sulit. Selain itu, jadwal manual bersifat kaku dan sulit beradaptasi ketika muncul tugas baru yang mendesak. Diperlukan sebuah pendekatan yang lebih sistematis, objektif, dan dinamis untuk mengatasi tantangan ini. Teknologi komputasi, khususnya algoritma, menawarkan potensi solusi yang menjanjikan.

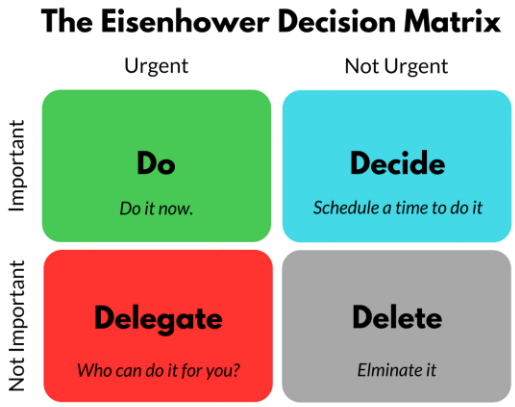
Makalah ini menguraikan desain dan analisis sebuah sistem penjadwalan otomatis yang berpusat pada algoritma Greedy. Tujuannya adalah untuk menyediakan sebuah kerangka kerja yang dapat membantu individu mengelola tugas harian mereka secara lebih efektif dengan mempertimbangkan dua faktor krusial: prioritas dan batas waktu. Tesis utamanya adalah: Algoritma Greedy, dengan kemampuannya membuat keputusan optimal lokal secara cepat berdasarkan kriteria heuristik, menawarkan solusi yang efisien, praktis, dan cukup baik (*good enough*) untuk mengatasi tantangan penjadwalan tugas harian pribadi yang bersifat dinamis dan kompleks.

II. Dasar Teori

A. Manajemen Waktu dan Produktivitas

Manajemen waktu adalah sebuah spektrum luas dari keterampilan, alat, dan teknik yang digunakan untuk mengelola waktu saat menyelesaikan tugas, proyek, dan tujuan tertentu. Fondasinya terletak pada kemampuan untuk membedakan antara berbagai jenis pekerjaan dan mengalokasikannya dengan benar.

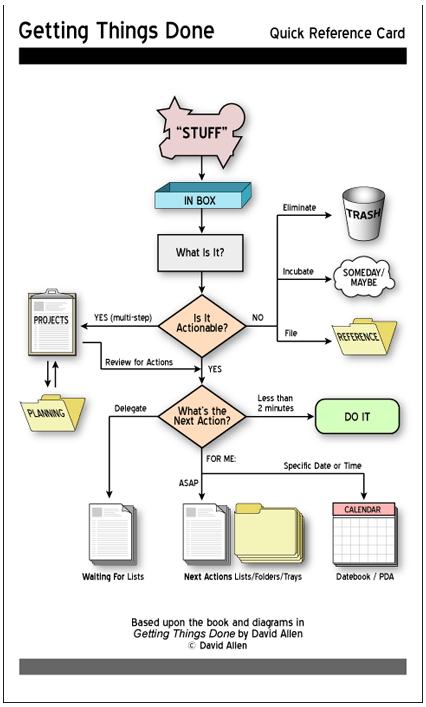
1. Matriks Eisenhower, *framework* yang mengkategorikan tugas berdasarkan dimensi kepentingan dan urgensi, menghasilkan empat kuadran.



Gambar 1. Matriks Eisenhower

- Kuadran I (Penting dan Mendesak): Krisis, masalah mendesak, proyek dengan tenggat waktu. (Untuk dikerjakan segera).
- Kuadran II (Penting & Tidak Mendesak): Perencanaan strategis, pencegahan, pembangunan relasi. (Untuk dijadwalkan).
- Kuadran III (Tidak Penting & Mendesak): Interupsi, beberapa email, rapat yang tidak perlu. (Untuk didelegasikan).
- Kuadran IV (Tidak Penting & Tidak Mendesak): Hal-hal sepele, aktivitas pembuang waktu. (Untuk dieliminasi).

2. Metode Get Things Done (GTD)



Gambar 2. Metode GTD

Dipopulerkan oleh David Allen, GTD adalah sistem manajemen alur kerja yang komprehensif. Sistem ini terdiri dari lima langkah: Tangkap (catat semua hal), Klarifikasi (proses apa artinya), Organisasi (letakkan di tempat yang tepat), Refleksi (tinjau secara berkala), dan Lakukan (kerjakan). Algoritma penjadwalan yang kita bahas berperan penting dalam langkah "Lakukan", yaitu membantu pengguna memutuskan "tindakan selanjutnya" dari daftar tugas yang sudah terorganisasi.

3. Teknik Pomodoro

Gambar 3. Pomodoro

Teknik ini adalah metode manajemen waktu yang berfokus pada eksekusi, menggunakan interval waktu (biasanya 25 menit) yang disebut "pomodoro", dipisahkan oleh istirahat singkat. Meskipun teknik ini tidak menentukan *apa* yang harus dikerjakan, ia dapat digunakan bersamaan dengan jadwal yang dihasilkan oleh algoritma Greedy untuk memastikan fokus yang mendalam pada setiap tugas yang telah dijadwalkan.

Prinsip ini menggarisbawahi bahwa tidak semua tugas diciptakan setara. Keputusan penjadwalan yang cerdas harus mempertimbangkan kedua dimensi ini, sebuah prinsip yang akan menjadi inti dari model algoritma yang akan diimplementasikan.

B. Algoritma Penjadwalan

Dalam ilmu komputer dan riset operasi, masalah penjadwalan (*scheduling problem*) adalah bidang studi fundamental yang berfokus pada alokasi sumber daya (seperti mesin, prosesor, atau bahkan manusia) untuk menyelesaikan sekumpulan tugas dalam periode waktu tertentu. Tujuannya adalah untuk mengoptimalkan satu atau lebih metrik, seperti meminimalkan total waktu penyelesaian (*makespan*), mengurangi keterlambatan (*tardiness*), atau memaksimalkan jumlah tugas yang selesai (*throughput*). Masalah penjadwalan tugas pribadi dapat dilihat sebagai varian dari masalah penjadwalan mesin tunggal (*single machine scheduling*), di mana "mesin" adalah waktu produktif seorang individu.

Selain single machine scheduling, terdapat varian lain dari masalah penjadwalan seperti parallel machine scheduling dan flow shop scheduling yang umum digunakan dalam dunia industri. Walaupun konteks makalah ini berfokus pada individu sebagai "mesin tunggal", gagasan dari sistem paralel dapat menginspirasi model kolaboratif antarindividu, misalnya dalam konteks kerja tim.

Dalam dunia nyata, seseorang mungkin berbagi tanggung jawab dengan rekan kerja, asisten, atau bahkan sistem otomatis lainnya. Dalam hal ini, pendekatan Greedy dapat diperluas untuk menentukan tugas mana yang dikerjakan sendiri dan mana yang dapat didelegasikan, mirip dengan penjadwalan pada sistem multi-prosesor. Algoritma Greedy bisa dimodifikasi menjadi "greedy distributed" dengan mempertimbangkan ketersediaan waktu pihak lain, tingkat kompetensi, dan urgensi bersama.

Selain itu, aspek penjadwalan berbasis prioritas lunak (soft priority scheduling) juga relevan. Tidak semua prioritas dapat didefinisikan secara biner atau numerik. Misalnya, beberapa tugas memiliki nilai emosional atau sosial yang tinggi namun sulit dikuantifikasi. Oleh karena itu, sistem penjadwalan modern sering menggabungkan elemen kuantitatif (durasi, deadline) dan kualitatif (mood, motivasi, makna personal), yang membuka pintu untuk penerapan algoritma fuzzy atau logika kabur dalam skenario penjadwalan pribadi.

- Penjadwalan Preemptive vs. Non-preemptive: Penjadwalan *preemptive* memungkinkan sebuah tugas yang sedang berjalan untuk diinterupsi oleh tugas lain yang berprioritas lebih tinggi. Penjadwalan *non-preemptive* mengharuskan sebuah tugas untuk berjalan hingga selesai sebelum tugas berikutnya dapat dimulai. Untuk kesederhanaan dan realisme dalam konteks tugas harian (sulit untuk "menyimpan" progres sebuah laporan di tengah jalan), model kita menggunakan pendekatan non-preemptive.
- Algoritma Penjadwalan Klasik:
    - Shortest Job First (SJF): Algoritma ini selalu memilih tugas dengan durasi eksekusi terpendek. SJF terbukti optimal untuk meminimalkan waktu tunggu rata-rata, namun berisiko membuat tugas-tugas panjang "kelaparan" (tidak pernah dieksekusi).
    - Earliest Deadline First (EDF): Algoritma ini memilih tugas dengan batas waktu paling awal. EDF optimal untuk memenuhi tenggat waktu jika tugas bersifat preemptive, tetapi bisa mengabaikan tugas penting dengan tenggat yang lebih jauh.

Pendekatan hibrida yang akan kita gunakan pada dasarnya adalah upaya untuk mengambil kekuatan dari berbagai strategi ini (fokus pada prioritas, urgensi, dan terkadang durasi) dalam satu formula heuristik.

C. Algoritma Greedy

Algoritma Greedy adalah paradigma yang sangat kuat karena kesederhanaannya. Agar sebuah masalah dapat diselesaikan secara efektif oleh algoritma Greedy, ia idealnya harus memiliki dua sifat utama:

1. Sifat Pilihan Greedy (Greedy Choice Property): Sifat ini menyatakan bahwa sebuah solusi optimal global dapat dicapai dengan membuat serangkaian pilihan yang optimal secara lokal. Dengan kata lain, pilihan terbaik yang dibuat pada saat ini tidak akan menghalangi pencapaian solusi akhir yang terbaik. Dalam konteks penjadwalan kita, kita membuat asumsi (yang tidak selalu benar, seperti yang akan kita lihat) bahwa memilih tugas dengan "skor tertinggi" saat ini adalah langkah terbaik menuju jadwal harian yang produktif.
2. Struktur Sub-Optimal (Optimal Substructure): Sifat ini berarti bahwa solusi optimal untuk suatu masalah mengandung solusi optimal untuk sub-masalahnya. Dalam kasus kita, jika sebuah jadwal J adalah jadwal optimal untuk n tugas, maka setelah memilih tugas pertama T_1, sisa jadwal tersebut haruslah merupakan jadwal optimal untuk n−1 tugas yang tersisa.

Salah satu contoh klasik yang sangat relevan adalah Masalah Pemilihan Aktivitas (Activity Selection Problem). Diberikan satu set aktivitas dengan waktu mulai dan waktu selesai, tujuannya adalah untuk memilih jumlah maksimum aktivitas yang tidak tumpang tindih yang dapat dilakukan oleh satu orang. Strategi Greedy yang terbukti optimal adalah: urutkan aktivitas berdasarkan waktu selesainya, lalu pilih aktivitas pertama. Kemudian, dari sisa aktivitas, pilih aktivitas berikutnya yang dimulai setelah aktivitas yang dipilih sebelumnya selesai. Proses ini diulangi terus-menerus. Analogi ini sangat dekat dengan tujuan kita untuk "memasukkan" sebanyak mungkin tugas penting ke dalam slot waktu yang terbatas.

Selain digunakan untuk penjadwalan, algoritma Greedy juga banyak diaplikasikan dalam berbagai bidang lainnya seperti kompresi data (misalnya Huffman coding), pemilihan jalur terpendek (seperti algoritma Dijkstra), dan penempatan iklan digital (greedy revenue maximization). Hal ini menunjukkan fleksibilitas paradigma Greedy dalam menyelesaikan masalah-masalah nyata yang membutuhkan

efisiensi tinggi dengan kompromi terhadap optimalitas. Dalam konteks personal productivity, paradigma ini sangat cocok karena waktu manusia adalah sumber daya terbatas dan keputusan harus dibuat dalam kondisi tekanan serta ketidakpastian.

III. Aplikasi Program Penjadwalan

A. *Algoritma*

1. Pemodelan Tugas dan Heuristik Keputusan

Untuk menerapkan algoritma, pertama-tama masalah perlu diformalkan. Setiap tugas T dimodelkan sebagai sebuah struktur data dengan atribut-atribut berikut.

- ID: Pengenal untuk setiap tugas.
- Deskripsi: Penjelasan singkat mengenai tugas.
- Durasi (d): Estimasi waktu yang dibutuhkan untuk menyelesaikan tugas dalam satuan menit.
- Prioritas (p): Tingkat kepentingan tugas, direpresentasikan secara numerik dengan skala 1 s.d. 5, di mana 1 berarti tidak penting dan 5 berarti sangat penting.
- Batas waktu (t): Waktu absolut kapan tugas harus selesai, dalam format tanggal, jam, menit.

Kunci dari algoritma Greedy terletak pada fungsi seleksi (heuristik) yang digunakan untuk membuat "pilihan optimal lokal". Daripada hanya memilih berdasarkan prioritas tertinggi atau tenggat terdekat, strategi hibrida yang menyeimbangkan keduanya terbukti paling efektif. Kita merancang sebuah "Skor Keputusan" untuk setiap tugas, dan pada setiap langkah, memilih tugas dengan skor tertinggi. Formula skor didefinisikan sebagai:

$$Skor(T_i) = (w_p \times p_i) + \frac{w_t}{t_i - t_{sekarang} + \in} + \frac{w_d}{d_i}$$

Di mana:

- p_i adalah nilai prioritas tugas.
- $t_i - t_{sekarang}$ adalah sisa waktu hingga batas, merepresentasikan urgensi tugas.
- d_i adalah durasi tugas.
- w_p, w_t, dan w_d adalah bobot yang merepresentasikan preferensi pengguna terhadap prioritas, urgensi waktu, dan urgensi durasi.
- $\in$ adalah konstanta kecil untuk mencegah pembagi nol.

Untuk memastikan fleksibilitas penggunaan, sistem ini juga dapat dikembangkan lebih lanjut dengan fitur seperti integrasi kalender, pengingat adaptif, serta pembelajaran preferensi pengguna dari waktu ke waktu. Misalnya, jika pengguna cenderung menunda tugas dengan durasi panjang, sistem dapat meningkatkan bobot penalti durasi dalam skor. Hal ini membawa pendekatan Greedy ke arah yang lebih kontekstual dan personal, menjadikannya bukan hanya alat bantu statis tetapi adaptif terhadap kebiasaan pengguna.

2. *Langkah Algoritma*

Berikut adalah langkah-langkah implementasi algoritma penjadwalan Greedy dalam bentuk pseudocode:

```
FUNCTION BuatJadwal(daftarTugas, waktuMulai, waktuSelesaiHarian):
    jadwalFinal = []
    tugasMustahil = [] // Daftar untuk tugas yang tidak bisa dijadwalkan
    waktuSekarang = waktuMulai
    daftarTugasTersedia = salin(daftarTugas)

    WHILE daftarTugasTersedia NOT EMPTY:
        tugasTerbaik = NULL
        skorTertinggi = -INFINITY
        tugasUntukDihapus = [] // Catat tugas yang mustahil

        FOR tugas IN daftarTugasTersedia:
            bisaSelesaiSebelumTenggat = (waktuSekarang + tugas.Durasi
            <= tugas.BatasWaktu)
            muatDiHariKerja = (waktuSekarang + tugas.Durasi
            <= waktuSelesaiHarian)

            IF bisaSelesaiSebelumTenggat AND muatDiHariKerja:
                skorTugas = HitungSkor(tugas, waktuSekarang)
                IF skorTugas > skorTertinggi:
                    skorTertinggi = skorTugas
                    tugasTerbaik = tugas
            ELSE:
                // Jika tugas sudah tidak mungkin,
                // tandai untuk dihapus & dipindahkan
                ADD tugas TO tugasUntukDihapus
                ADD tugas TO tugasMustahil

        // Hapus semua tugas yang mustahil dari pertimbangan
        FOR tugas IN tugasUntukDihapus:
            DELETE tugas FROM daftarTugasTersedia

        // Jika ditemukan tugas terbaik yang layak
        IF tugasTerbaik != NULL:
            ADD tugasTerbaik TO jadwalFinal
            waktuSekarang = waktuSekarang + tugasTerbaik.Durasi
            DELETE tugasTerbaik FROM daftarTugasTersedia

        // Jika tidak ada lagi tugas yang bisa dipilih, hentikan
        ELSE:
            BREAK

    RETURN jadwalFinal, tugasMustahil
```

Gambar 4. Pseudocode Penjadwalan

B. *Studi Kasus dan Analisis Hasil*

Untuk mengilustrasikan penerapan praktis dari algoritma yang telah dirancang, berikut disajikan sebuah studi kasus simulasi. Misalkan satu hari kerja dimulai pukul 09:00 dengan waktu kerja berakhir pukul 12:00. Digunakan strategi hibrida dengan bobot $w_p = 0.5$, $w_t = 0.3$, dan $w_d = 0.2$.

ID	Deskripsi	Durasi (m)	Prioritas (1-5)	Batas waktu
A	Revisi minor PPT	20	4	10:00
B	Analisis data besar	120	4	11:30

C	Bayar tagihan penting	10	5	09:30
D	Riset makalah	90	3	12:00
E	Telepon vendor (sudah terlambat)	30	5	09:00

Tugas E yang apabila dikerjakan melewati batas waktu akan langsung ditandai oleh algoritma sebagai “mustahil”. Skor tugas E tidak akan dihitung dengan rumus. Skor tugas A, B, C, dan D berturut-turut adalah 2.015, 2.004, 2.530, dan 1.504. Pada proses seleksi, tugas D juga tidak mendapatkan jadwal pengerjaan dan harus dimasukkan ke list mustahil dan akan di-reschedule untuk jadwal berikutnya.

Urutan	Waktu	ID	Deskripsi
1	09:00 – 09:10	C	Bayar tagihan penting
2	09:10 – 09:30	A	Analisis data besar
3	09:30 – 11:30	B	Revisi minor PPT

Analisis hasil menunjukkan bahwa algoritma tidak hanya memilih prioritas tertinggi (Tugas C dan E), tetapi juga mempertimbangkan urgensi dan durasi. Tugas C dipilih pertama karena skor keseluruhannya (prioritas tinggi, sangat mendesak, sangat singkat) menjadikannya pilihan yang jelas. Algoritma mampu secara otomatis mengidentifikasi dan menyisihkan tugas yang sudah tidak mungkin diselesaikan (Tugas E) dan tugas yang tidak muat dalam slot waktu yang tersedia (Tugas D). Skor tugas dihitung ulang setiap iterasi, memastikan keputusan selalu dibuat berdasarkan kondisi *saat ini*, bukan hanya kondisi awal.

Studi kasus berikutnya menggunakan data tugas berikut. Misalkan satu hari kerja dimulai pukul 13:00 dengan waktu kerja berakhir pukul 17:00. Akan digunakan strategi bobot w_p = 0.5, w_t = 0.3, dan w_d = 0.2

ID	Deskripsi	Durasi (m)	Prioritas (1-5)	Batas waktu
X	Buat fitur baru	150	5	17:00
Y	Fix bug A	60	4	15:30
Z	Fix bug B	30	4	14:30

Menggunakan heuristik yang telah dirancang sebelumnya, skor tugas X adalah 2.503, tugas Y dan Z dengan skor masing-masing 2.005 dan 2.010. Jadwal yang dihasilkan oleh algoritma Greedy sesuai rancangan hanya akan mengambil tugas X untuk dikerjakan. Padahal solusi optimal global dapat mengerjakan ketiga tugas dengan urutan Z, Y, lalu X.

Kasus kedua merupakan contoh nyata keterbatasan algoritma Greedy, karena mengejar optimal lokal, terkadang algoritma tidak akan sampai pada optimal secara global. Algoritma ini hanya melihat "Apa langkah terbaik *saat ini*?". Mengerjakan tugas prioritas 5 (Tugas A) adalah pilihan yang paling menguntungkan secara instan. Algoritma tidak memiliki kemampuan untuk mempertimbangkan, "Jika saya melakukan ini sekarang, apa dampaknya pada kemampuan saya untuk menyelesaikan tugas-tugas lain nanti?".

Pada studi kasus ketiga, diasumsikan hari kerja berlangsung dari pukul 08:00 hingga 12:00, dengan total waktu kerja efektif 240 menit. Namun, di tengah hari akan ada gangguan eksternal berupa rapat mendadak selama 30 menit yang tidak dapat dipindahkan pada pukul 09:30–10:00. Hal ini mensimulasikan realitas gangguan tak terduga dalam dunia kerja.

ID	Deskripsi	Durasi (menit)	Prioritas (1-5)	Batas Waktu
P	Balas email penting	20	3	10:30
Q	Menulis laporan	100	4	12:00
R	Cek dana mingguan	30	2	11:00
S	Review proposal	60	5	11:30

Untuk mempertimbangkan gangguan ini, algoritma Greedy disesuaikan agar memperlakukan waktu rapat sebagai “slot waktu tak tersedia” dan tidak menjadwalkan tugas apa pun di dalamnya. Berdasarkan skor keputusan dan alokasi waktu yang tersedia (terbagi dua: 08:00–09:30 dan 10:00–12:00), sistem menghasilkan urutan tugas sebagai berikut:

Waktu	ID	Deskripsi
08:00-08:20	P	Membalas email penting
08:20-09:20	S	Review proposal
09:30-10:00	-	Rapat internal (gangguan)
10:00-11:40	Q	Menulis laporan

Tugas R tidak dimasukkan karena tidak muat dalam slot waktu tersisa, meskipun prioritasnya tidak terlalu rendah. Hal ini menunjukkan fleksibilitas algoritma Greedy dalam menyesuaikan penjadwalan dengan kendala non-tugas. Algoritma tetap efektif meskipun menghadapi jadwal dinamis, selama input gangguan dapat diketahui terlebih dahulu dan dimasukkan ke dalam sistem.

Hal menarik lainnya adalah bagaimana skor tugas perlu diperbarui ulang setelah slot waktu "rapat" selesai. Sistem secara otomatis menghitung ulang skor berdasarkan waktu terkini, bukan kondisi awal, yang memperkuat karakteristik adaptif algoritma Greedy.

C. Analisis Kelebihan dan Keterbatasan

1. Kelebihan:
   - Kecepatan dan Kesederhanaan: Algoritma ini sangat cepat dan mudah diimplementasikan, cocok untuk aplikasi interaktif di mana pengguna mengharapkan respons instan.
   - Solusi Cukup Baik (*Good Enough*): Untuk penjadwalan pribadi, solusi yang "cukup baik" dan cepat didapat seringkali lebih berharga daripada

solusi "sempurna" yang butuh waktu lama untuk dihitung.

- Adaptif: Jika ada tugas baru atau perubahan prioritas, seluruh jadwal baru dapat dibuat ulang dalam hitungan detik.

2. Keterbatasan:

- Tidak Menjamin Solusi Optimal Global: Sifatnya yang "rabun jauh" (*myopic*) dapat menyebabkan hasil sub-optimal. Misalnya, memilih tugas prioritas tinggi yang berdurasi sangat panjang di pagi hari bisa menyebabkan beberapa tugas prioritas sedang dengan tenggat sore hari menjadi terlewat.
- Ketergantungan pada Heuristik: Kualitas jadwal sangat bergantung pada desain formula skor dan bobot yang dipilih. Bobot yang salah dapat menghasilkan jadwal yang terasa tidak intuitif.

Untuk mengurangi dampak keterbatasan ini, pendekatan Greedy dapat dikombinasikan dengan metode metaheuristik seperti simulated annealing atau algoritma genetika, yang memungkinkan eksplorasi ruang solusi yang lebih luas. Alternatif lainnya adalah menggunakan algoritma berbasis lookahead terbatas yang mempertimbangkan konsekuensi jangka pendek dari pilihan saat ini, meskipun dengan tambahan beban komputasi. Hal ini membuka peluang untuk riset lanjutan yang mengeksplorasi keseimbangan antara kompleksitas perhitungan dan kualitas solusi dalam konteks penjadwalan pribadi.

D. Perbandingan dengan Pendekatan Lain

Walaupun algoritma Greedy menawarkan kecepatan dan kesederhanaan, penting untuk membandingkan pendekatan ini dengan strategi algoritma lain yang lebih kompleks, seperti Dynamic Programming dan Backtracking, terutama dalam konteks penjadwalan yang lebih kritis atau skala besar.

1. Dynamic Programming (DP)

DP bekerja dengan menyimpan hasil submasalah yang telah diselesaikan, dan menggunakannya kembali saat dibutuhkan. Pendekatan ini cocok ketika semua kombinasi tugas perlu dievaluasi untuk mencapai optimalitas global. Namun, DP memiliki keterbatasan pada kompleksitas waktu dan ruang yang tinggi, membuatnya kurang cocok untuk aplikasi interaktif harian.

3. Backtracking

Backtracking melakukan eksplorasi menyeluruh (brute-force sistematis) terhadap semua kemungkinan jadwal dan melakukan backtrack jika solusi saat ini tidak memungkinkan. Walaupun menjamin solusi optimal, pendekatan ini sangat lambat dan tidak praktis untuk konteks dinamis dengan banyak tugas dan batasan waktu ketat.

Dengan demikian, pemilihan algoritma sangat tergantung pada konteks penggunaan. Dalam kasus penjadwalan pribadi harian yang bersifat ringan hingga menengah, algoritma Greedy menawarkan *trade-off* yang efisien antara kualitas dan kecepatan.

IV. KESIMPULAN

Makalah ini telah membuktikan bahwa algoritma Greedy dapat dimanfaatkan secara efektif sebagai pendekatan praktis untuk penjadwalan tugas harian pribadi yang mempertimbangkan prioritas, durasi, dan batas waktu. Dengan menggabungkan prinsip keputusan lokal optimal dan formula heuristik hibrida, sistem ini mampu menghasilkan jadwal yang efisien dalam waktu komputasi yang sangat singkat.

Keunggulan utama algoritma Greedy terletak pada kesederhanaannya, kecepatan respon, dan kemampuannya beradaptasi terhadap perubahan input secara dinamis. Pendekatan ini sangat cocok untuk konteks personal di mana kebutuhan utama adalah *solusi cukup baik* yang cepat, bukan keoptimalan mutlak.

Namun demikian, keterbatasan inheren dari strategi Greedy—terutama kecenderungannya terhadap solusi optimal lokal—menjadi catatan penting. Dalam kasus tertentu, algoritma dapat memilih tugas yang menguntungkan secara instan namun mengorbankan penyelesaian tugas-tugas lainnya. Oleh karena itu, keberhasilan sistem ini sangat bergantung pada desain formula skor yang tepat dan pemahaman pengguna terhadap sifat algoritma.

Ke depan, sistem ini dapat dikembangkan menjadi aplikasi produktivitas yang lebih komprehensif, dengan integrasi ke dalam kalender digital, sistem notifikasi, dan pelacakan kebiasaan pengguna. Kekuatan sesungguhnya dari pendekatan ini bukanlah menggantikan kecerdasan manusia, melainkan mendukung pengambilan keputusan yang lebih terstruktur, efisien, dan reflektif dalam kehidupan sehari-hari.

V. IMPLIKASI SOSIAL DAN ETIKA

Penggunaan algoritma dalam pengambilan keputusan harian secara otomatis membawa sejumlah implikasi sosial dan etika yang layak dipertimbangkan. Otomatisasi penjadwalan bisa mengubah cara manusia mengelola waktu, namun juga berisiko menurunkan fleksibilitas dan mengabaikan nilai-nilai kontekstual yang tidak bisa sepenuhnya diwakili oleh angka prioritas atau durasi.

Misalnya, sistem penjadwal otomatis bisa saja terus menyarankan kegiatan "produktivitas tinggi", tanpa memberi ruang pada aktivitas reflektif, waktu luang, atau interaksi sosial yang sulit dikategorikan sebagai "tugas penting". Jika tidak diawasi dengan bijak, hal ini dapat menciptakan tekanan psikologis yang berlawanan dengan tujuan awal sistem itu sendiri.

Oleh karena itu, penting agar sistem yang dirancang tidak hanya mengutamakan efisiensi, tetapi juga mengintegrasikan *design for well-being* dengan memberikan peringatan waktu istirahat, mendorong evaluasi pribadi, atau bahkan memberi jeda yang disengaja dari aktivitas. Selain itu, isu seperti privasi data pengguna dan transparansi dalam pengambilan keputusan juga perlu diperhatikan jika sistem diintegrasikan dengan platform daring.

LINK VIDEO YOUTUBE

https://youtu.be/5kop17K8mLc?si=BIkqIxH5fis0TUHp

UCAPAN TERIMA KASIH

Penulis mengucap syukur kepada Tuhan Yang Maha Esa karena atas berkat dan rahmat-Nya, penulis dapat menyelesaikan tugas makalah Strategi Algoritma yang berjudul "Pemanfaatan Algoritma Greedy untuk Penjadwalan Tugas Harian Pribadi dengan Prioritas dan Batasan Waktu". Ucapan terima kasih juga penulis sampaikan kepada dosen pengampu mata kuliah IF2211 Strategi Algoritma, Bapak Ir. Rinaldi Munir, Ibu Dr. Nur Ulfa Maulidevi, dan Bapak Monterico Adrian, S.T., M.T. selaku dosen yang telah mengajar mata kuliah ini dan membuat penulis dapat menyelesaikan tugas makalah ini dengan baik. Penulis juga mengucapkan terima kasih kepada keluarga dan seluruh pihak yang senantiasa mendukung saya dalam pembuatan makalah ini.

REFERENSI

[1] Allen, David. (2001). *Getting Things Done: The Art of Stress-Free Productivity*. Penguin Books.

[2] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3rd ed.). MIT Press.

[3] Eisenhower, D. D. (1961). *Farewell Address to the Nation*.

[4] Pinedo, M. L. (2016). *Scheduling: Theory, Algorithms, and Systems* (5th ed.). Springer.

[5] Lawler, E. L., Lenstra, J. K., & Rinnooy Kan, A. H. G. (1982). Recent developments in deterministic sequencing and scheduling: A survey. In *Deterministic and Stochastic Scheduling* (pp. 35-73). Springer, Dordrecht.

[6] Newport, Cal. (2016). *Deep Work: Rules for Focused Success in a Distracted World*. Grand Central Publishing.

[7] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/04-Algoritma-Greedy-(2025)-Bag1.pdf

[8] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/05-Algoritma-Greedy-(2025)-Bag2.pdf

[9] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/06-Algoritma-Greedy-(2025)-Bag3.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, 1 Juni 2025

Indah Novita Tangdililing
13523047