# Pemecahan Masalah Optimasi dengan Sistem Persamaan Linier dalam Pemrograman Komputer : Kasus Penjadwalan Tugas dalam Alur Kerja

Sabilul Huda - 13523072[1,2]
*Program Studi Teknik Informatika*
*Sekolah Teknik Elektro dan Informatika*
*Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia*
[1]13523072@std.stei.itb.ac.id, [2]sabilulhuda060106@gmail.com

***Abstrak–Optimasi alur kerja program merupakan salah satu tantangan utama dalam pengembangan perangkat lunak, terutama untuk meningkatkan efisiensi waktu eksekusi program. Penelitian ini bertujuan untuk mengaplikasikan sistem persamaan linier (SPL) dalam mengoptimalkan alur kerja program berbasis graf dependensi. Setiap fungsi dalam program direpresentasikan sebagai simpul pada graf, dengan hubungan dependensi antar simpul menunjukkan urutan eksekusi fungsi. Melalui penyelesaian SPL yang merepresentasikan waktu mulai eksekusi setiap fungsi, diperoleh urutan optimal eksekusi program yang meminimalkan waktu total pelaksanaan. Hasil penerapan metode ini menunjukkan bahwa waktu mulai eksekusi fungsi utama (x1) dapat ditentukan bersama dengan fungsi lainnya, menghasilkan urutan eksekusi yang efisien. Penelitian ini juga memberikan wawasan tentang potensi penerapan SPL untuk mengelola alur kerja pada sistem dengan graf dependensi yang lebih kompleks. Studi ini diharapkan menjadi langkah awal bagi pengembangan metode optimasi berbasis SPL dalam pengelolaan alur kerja program berskala besar.***

***Kata Kunci–Sistem Persamaan Linier, Optimasi Program, Graf Dependensi, Alur Kerja Program, Efisiensi Waktu Eksekusi.***

## I. PENDAHULUAN

Optimasi merupakan salah satu bidang yang memiliki peranan penting dalam berbagai aspek kehidupan, terutama dalam bidang teknologi dan pemrograman komputer. Proses optimasi bertujuan untuk menemukan solusi terbaik dari berbagai alternatif yang tersedia berdasarkan kriteria tertentu. Dalam ranah pemrograman komputer, optimasi sering diterapkan pada masalah seperti alokasi sumber daya, perencanaan logistik, penjadwalan, dan analisis data. Kemampuan untuk menemukan solusi optimal dengan cara yang efisien sangat dibutuhkan, terutama dalam menghadapi masalah yang kompleks dan berskala besar.

Salah satu pendekatan matematis yang sering digunakan untuk menyelesaikan masalah optimasi adalah sistem persamaan linier. Sistem persamaan linier menyediakan kerangka kerja yang sederhana namun kuat untuk merepresentasikan dan memecahkan berbagai masalah optimasi. Dalam konteks ini, pemrograman komputer memainkan peranan penting dalam mengimplementasikan algoritma yang diperlukan untuk menyelesaikan sistem persamaan linier secara efisien. Dengan bantuan perangkat lunak dan teknik numerik, masalah optimasi yang melibatkan ratusan hingga ribuan variabel dapat diselesaikan dengan cepat.

Motivasi utama dari makalah ini adalah untuk menunjukkan bagaimana sistem persamaan linier dapat digunakan untuk menyelesaikan masalah optimasi dalam pemrograman komputer. Pendekatan ini tidak hanya relevan dalam teori, tetapi juga memiliki aplikasi praktis yang luas, mulai dari perencanaan jalur transportasi hingga pembelajaran mesin.

Makalah ini bertujuan untuk memberikan penjelasan yang sistematis mengenai peran sistem persamaan linier dalam pemrograman komputer untuk masalah optimasi. Selain itu, makalah ini akan menyajikan contoh aplikasi sederhana guna memperjelas konsep dan manfaat dari pendekatan ini.

Adapun ruang lingkup makalah ini mencakup penjelasan dasar mengenai sistem persamaan linier dan penerapannya dalam optimasi. Pembahasan akan difokuskan pada implementasi dasar yang mudah dipahami, tanpa mendalami aspek teknis yang terlalu kompleks.

Struktur makalah ini diawali dengan pendahuluan yang menjelaskan latar belakang dan tujuan. Bagian berikutnya membahas teori dasar sistem persamaan linier serta kaitannya dengan optimasi. Selanjutnya, akan disajikan contoh aplikasi untuk menggambarkan penerapan praktisnya. Akhirnya, kesimpulan akan meringkas poin-poin penting dari pembahasan makalah ini.

## II. STUDI LITERATUR

### *A. Sistem Persamaan Linier*

#### *A.1. Definisi Sistem Persamaan Linier*

Sistem persamaan linear adalah sekumpulan persamaan matematika di mana setiap persamaan adalah persamaan linear, yang berarti setiap variabel di dalamnya memiliki pangkat satu (tidak ada kuadrat, akar, atau pangkat lain). Sistem ini melibatkan satu atau lebih persamaan yang berbentuk:

$$a_1 x_1 + a_2 x_2 + \ \ldots \ + a_n x_n = b$$

dengan:

$a_1$, $a_2$, ..., $a_n$ adalah koefisien (bilangan tetap),

$x_1$, $x_2$, ..., $x_n$ adalah variabel yang dicari nilainya,

$b$ adalah konstanta. Contoh:

$$2x \ + \ 3y \ = \ 6$$
$$x \ - \ y \ = \ 2$$

Sistem ini memiliki dua persamaan linear dengan dua variabel, $x$ dan $y$. Tujuan utamanya adalah menentukan nilai $x$ dan $y$ yang memenuhi kedua persamaan tersebut secara bersamaan.

#### *A.2. Representasi SPL dalam Bentuk Matriks Koefisien Augmented*

Sistem persamaan linear (SPL) dapat direpresentasikan dalam bentuk matriks untuk menyederhanakan perhitungannya. Contoh:

$$2x + y + x \ = \ 5$$
$$4x + 3y + z \ = \ 6$$
$$- \ 2x + 2y + z \ = \ - \ 3$$

Matriks koefisien augmented dari SPL diatas merupakan matriks dengan jumlah baris = banyak persamaan dan jumlah kolom = banyak variabel + 1, yang berisi koefisien dari variabel pada persamaan yang bersesuaian, dan kolom paling kanan berisi konstanta pada SPL. Matriks koefisien augmented dari SPL diatas adalah:

$$[A|b] = \left[\begin{array}{ccc|c} 2 & 1 & 1 & 5 \\ 4 & 3 & 1 & 6 \\ -2 & 2 & 1 & -3 \end{array}\right]$$

#### *A.3. Matriks Eselon Baris*

Matriks eselon baris adalah bentuk matriks yang diperoleh setelah beberapa transformasi operasi baris elementer (OBE) sehingga matriks menjadi lebih sederhana untuk penyelesaian SPL. Karakteristik matriks eselon baris adalah: 1. semua baris nol (jika ada) berada di bagian bawah matriks, 2. elemen pertama (pivot) di setiap baris non-nol berada di sebelah kanan pivot baris di atasnya, 3. elemen di bawah pivot adalah nol. Contoh matriks eselon baris

$$\left[\begin{array}{ccc|c} 1 & 2 & -1 & 3 \\ 0 & 1 & 3 & -1 \\ 0 & 0 & 2 & 4 \end{array}\right]$$

#### *A.4. Matriks Eselon Baris Tereduksi*

Matriks eselon baris tereduksi adalah bentuk matriks yang lebih sederhana daripada eselon biasa. Karakteristik matriks eselon baris tereduksi: 1. Memenuhi semua sifat matriks eselon, 2. pivot (elemen utama) di setiap baris bernilai 1, 3. elemen di atas dan di bawah pivot adalah nol. Contoh matriks eselon baris tereduksi:

$$\left[\begin{array}{ccc|c} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 3 \end{array}\right]$$

#### *A.5. Operasi Baris Elementer(OBE)*

OBE adalah operasi pada baris matriks yang digunakan untuk mengubah matriks menjadi bentuk eselon atau eselon baris tereduksi. OBE tidak mengubah solusi SPL. Berikut adalah operasi yang boleh dilakukan selama melakukan OBE: 1. menukar baris, 2. mengalikan suatu baris dengan skalar tak-nol, 3. penjumlahan atau pengurangan baris. Ketiga operasi tersebut tidak mengubah solusi SPL yang dihasilkan. Contoh OBE, misalkan terdapat matriks augmented berikut:

$$[A|b] = \left[\begin{array}{ccc|c} 2 & 1 & 1 & 5 \\ 4 & 3 & 1 & 6 \\ -2 & 2 & 1 & -3 \end{array}\right]$$

Kali baris satu dengan $1/2$:

$$\left[\begin{array}{ccc|c} 1 & 0.5 & 0.5 & 2.5 \\ 4 & 3 & 1 & 6 \\ -2 & 2 & 1 & -3 \end{array}\right]$$

Kurangi $4R_1$ dari $R_2$:

$$\left[\begin{array}{ccc|c} 1 & 0.5 & 0.5 & 2.5 \\ 0 & 1 & -1 & -4 \\ -2 & 2 & 1 & -3 \end{array}\right]$$

OBE ini dilanjutkan hingga matriks mencapai bentuk eselon atau eselon baris tereduksi.

#### *A.6. Penyelesaian Sistem Persamaan Linier Menggunakan Metode Eliminasi Gauss*

Metode ini menggunakan operasi baris elementer untuk mengubah matriks koefisien menjadi bentuk segitiga atas. Langkah-langkahnya, pertama representasi matriks augmented, tuliskan SPL dalam bentuk matriks augmented [$A|b$] di mana A adalah matriks koefisien, dan b adalah vektor konstanta. Kedua adalah operasi baris elementer, lakukan operasi baris elementer untuk mendapatkan matriks eselon baris. Terakhir adalah

substitusi mundur, setelah terbentuk matriks eselon baris, mulailah dari baris terakhir untuk mencari nilai variabel dan substitusi ke atas. Contoh, tinjau SPL berikut:

$3x_1 + 5x_2 - 4x_3 = 7$

$-3x_1 - 2x_2 + 4x_3 = -1$

$6x_1 + x_2 - 8x_3 = -4$

Ubah SPL tersebut dalam bentuk matriks koefisien augmented berikut:

$$\begin{bmatrix} 3 & 5 & -4 & 7 \\ -3 & -2 & 4 & -1 \\ 6 & 1 & -8 & -4 \end{bmatrix}$$

Lakukan OBE hingga menjadi matriks eselon baris berikut:

$$\begin{bmatrix} 1 & 0 & -\frac{4}{3} & -1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Lakukan substitusi mundur sehingga didapat solusi $x_1 = -1 + 4/3\, x_3$, $x_2 = 2$, $x_3 = s$ dengan $s \in \Re$.

### *B. Kompleksitas Algoritma*

Kompleksitas algoritma merupakan ukuran kinerja sebuah algoritma berdasarkan sumber daya komputasi yang diperlukan, seperti waktu eksekusi dan penggunaan memori. Dua jenis kompleksitas yang paling umum digunakan adalah kompleksitas waktu dan kompleksitas ruang.

Kompleksitas waktu mengukur seberapa cepat sebuah algoritma berjalan seiring bertambahnya ukuran input (). Hal ini sering dinyatakan dalam notasi Big-O, yang memberikan batas atas dari waktu eksekusi algoritma dalam skenario terburuk. Beberapa kategori kompleksitas waktu yang sering ditemukan adalah: 1.Konstan, waktu tidak bergantung pada ukuran input, 2. Linear, waktu meningkat secara langsung proporsional dengan ukuran input, 3. Kuadratik, waktu meningkat secara kuadratik terhadap ukuran input, 4. Logaritmik, waktu meningkat logaritmik seiring ukuran input bertambah.

Kompleksitas ruang mengukur jumlah memori tambahan yang dibutuhkan oleh algoritma, selain ruang untuk menyimpan input. Sama seperti kompleksitas waktu, kompleksitas ruang juga dinyatakan dalam notasi Big-O.

### *C. Teori Graf*

Graf adalah struktur data yang digunakan untuk merepresentasikan hubungan antar objek. Sebuah graf terdiri dari dua komponen utama yaitu simpul (Vertex, Node) yang merepresentasikan objek atau entitas, dan sisi (Edge) yang merepresentasikan hubungan atau koneksi antara simpul.

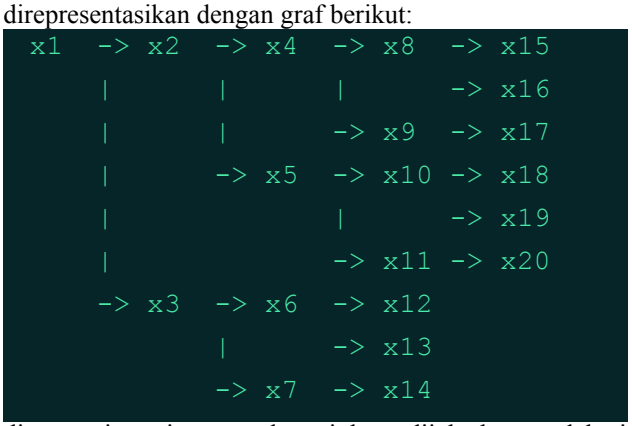
Graf dapat diklasifikasikan menjadi beberapa jenis berdasarkan karakteristiknya: 1. graf tidak berarah (undirected graph) yaitu hubungan antar simpul tidak memiliki arah tertentu, 2. graf berarah (directed graph) yaitu hubungan antar simpul memiliki arah tertentu, direpresentasikan dengan panah, 3. graf berbobot (weighted graph) yang setiap sisi memiliki bobot atau nilai tertentu, 4. graf siklik dan asiklik dimana graf siklik memiliki setidaknya satu jalur yang membentuk lingkaran dan graf asiklik (DAG - Directed Acyclic Graph) adalah graf berarah tanpa siklus, sering digunakan dalam pemodelan alur kerja.

Dalam konteks alur kerja program, graf digunakan untuk merepresentasikan dependensi antar tugas atau operasi. Setiap simpul mewakili sebuah tugas, dan setiap sisi menunjukkan hubungan dependensi, seperti urutan eksekusi atau transfer data. Topological Sorting, digunakan untuk menentukan urutan eksekusi tugas dalam DAG. Algoritma ini memiliki kompleksitas , di mana adalah jumlah simpul dan adalah jumlah sisi. Penjadwalan Tugas, graf digunakan untuk menyusun jadwal optimal berdasarkan waktu eksekusi tugas, mengurangi konflik dependensi, dan memaksimalkan penggunaan sumber daya. Optimasi Alur Kerja, dengan menggunakan teori graf, alur kerja dapat dioptimalkan untuk mengurangi waktu eksekusi total (makespan) atau meminimalkan penggunaan sumber daya.

## III. METODE

### *A. Menentukan SPL Berdasarkan Graf Dependensi Fungsi dan Durasi Fungsi*

Misalkan terdapat suatu program yang memiliki 20 fungsi, yaitu $x1$, $x2$, …, $x20$, dengan x1 sebagai program utama. Terdapat aturan dependensi fungsi yang direpresentasikan dengan graf berikut:




dimana xi -> xj menyatakan xi dapat dijalankan setelah xj dijalankan. Diberikan pula durasi tiap fungsi sebagai berikut:

- $x1 : 9s$
- $x2 : 9s$
- $x3 : 1\,s$
- $x4 : 17s$
- $x5 : 14s$
- $x6 : 5s$

- $x7 : 8s$
- $x8 : 11s$
- $x9 : 13s$
- $x10 : 6s$
- $x11 : 13s$
- $x12 : 5s$
- $x13 : 12s$
- $x14 : 19s$
- $x15 : 8s$
- $x16 : 4s$
- $x17 : 14s$
- $x18 : 19s$
- $x19 : 12s$
- $x20 : 20s$

Misalkan $t_i$ adalah waktu mulai fungsi $x_i$ dijalankan, maka berdasarkan graf dependensi dan durasi fungsi, diperoleh SPL berikut:

$t1 = t2 + 9$
$t1 = t3 + 1$
$t2 = t4 + 17$
$t2 = t5 + 14$
$t3 = t6 + 5$
$t3 = t7 + 8$
$t4 = t8 + 11$
$t4 = t9 + 13$
$t5 = t10 + 6$
$t5 = t11 + 13$
$t6 = t12 + 5$
$t6 = t13 + 12$
$t7 = t14 + 19$
$t8 = t15 + 8$
$t8 = t16 + 4$
$t9 = t17 + 14$
$t10 = t18 + 19$
$t10 = t19 + 12$
$t11 = t20 + 20$
$t12 = t13 = \ldots = t20 = 0$

Dengan subtitusi $t12 = t13 = \ldots = t20 = 0$ dan memilih nilai terbesar dari $x_i$ jika memiliki dua nilai atau lebih, didapat SPL final sebagai berikut:

$t1 = t2 + 9$
$t1 = t3 + 1$
$t2 = t4 + 17$
$t2 = t5 + 14$
$t3 = t6 + 5$
$t3 = t7 + 8$
$t4 = t8 + 11$
$t4 = t9 + 13$
$t5 = t10 + 6$
$t5 = t11 + 13$
$t6 = 12$
$t7 = 19$
$t8 = 8$
$t9 = 14$
$t10 = 19$
$t11 = 20$

Berikut adalah SPL dalam bentuk matriks koefisien augmented:

$$\left[\begin{array}{cccccccccc|c} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9 \\ 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 17 \\ 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 14 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 5 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 8 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 11 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 13 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 6 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 13 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 12 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 19 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 14 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 19 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \mid 20 \end{array}\right]$$

### *B. Program untuk Menyelesaikan SPL menggunakan Metode Gauss*

Berikut adalah program dalam bahasa python yang digunakan untuk mencari solusi dari SPL tersebut:

```python
def gauss_elimination(matrix):
    n = len(matrix)  # jumlah baris
    m = len(matrix[0])  # jumlah kolom

    # Forward elimination
    for i in range(n):
        # Mencari pivot
        max_row = i
        for k in range(i + 1, n):
            if abs(matrix[k][i]) > abs(matrix[max_row][i]):
                max_row = k

        # Menukar baris
        matrix[i], matrix[max_row] = matrix[max_row], matrix[i]

        # Pastikan pivot bukan nol
        if matrix[i][i] == 0:
            print("Sistem tidak memiliki solusi unik.")
            return None

        # Normalisasi baris pivot
        pivot = matrix[i][i]
        for j in range(i, m):
            matrix[i][j] /= pivot

        # Eliminasi baris di bawah
        for k in range(i + 1, n):
            factor = matrix[k][i]
            for j in range(i, m):
                matrix[k][j] -= factor * matrix[i][j]

    # Back substitution
    solution = [0] * (m - 1)
    for i in range(n - 1, -1, -1):
        solution[i] = matrix[i][-1]
        for j in range(i + 1, m - 1):
            solution[i] -= matrix[i][j] * solution[j]

    return solution
```

**Gambar 1** Program Penyelesaian SPL Menggunakan Metode Gauss

## IV. HASIL DAN PEMBAHASAN

Setelah matriks koefisien augmented diproses oleh

program yang telah dibuat, didapat hasil sebagai berikut:

$t1 = 45s$
$t2 = 36s$
$t3 = 17s$
$t4 = 19s$
$t5 = 25s$
$t6 = 12s$
$t7 = 19s$
$t8 = 8s$
$t9 = 14s$
$t10 = 19s$
$t11 = 20s$
$t12 = t13 = \ldots = t20 = 0s$

Berdasarkan hasil tersebut, didapat bahwa waktu optimum program ini adalah 45 s.

## V. KESIMPULAN

Dalam optimasi alur kerja program, pendekatan menggunakan Sistem Persamaan Linier (SPL) telah terbukti efektif untuk memodelkan dan menganalisis ketergantungan antar komponen program. Dengan memanfaatkan graf dependensi, setiap fungsi dalam program dapat dioptimalkan urutannya untuk meminimalkan waktu eksekusi total. Metode ini memungkinkan identifikasi jalur kritis dan perhitungan waktu penyelesaian secara efisien. Hasil analisis menunjukkan bahwa SPL mampu meningkatkan performa program dengan restrukturisasi urutan eksekusi, memberikan efisiensi signifikan dalam waktu dan sumber daya. Pendekatan ini juga memiliki potensi besar untuk diterapkan dalam skenario lebih kompleks, seperti sistem paralel dan terdistribusi.

## VI. SARAN

Saran untuk peneliti selanjutnya adalah agar optimasi alur kerja program menggunakan sistem persamaan linier tidak hanya mempertimbangkan waktu mulai eksekusi, tetapi juga dapat mengintegrasikan faktor-faktor lain seperti penggunaan memori, tingkat paralelisme, dan efisiensi alokasi sumber daya komputasi. Selain itu, penelitian lebih lanjut dapat mengembangkan metode untuk menangani graf dependensi yang lebih kompleks atau dinamis, sehingga pendekatan ini dapat diterapkan pada sistem berskala besar dengan perubahan dependensi waktu nyata.

## DAFTAR PUSTAKA


[1] H. Anton dan C. Rorres, Elementary Linear Algebra: Applications Version, 11th ed. Hoboken, NJ: Wiley, 2013.

[2] G. Strang, Introduction to Linear Algebra, 5th ed. Wellesley, MA: Wellesley-Cambridge Press, 2016.

[3] S. Cormen, T. H. Leiserson, C. E. Rivest, dan C. Stein, Introduction to Algorithms, 3rd ed. Cambridge, MA: MIT Press, 2009.

[4] D. Knuth, The Art of Computer Programming: Volume 1 - Fundamental Algorithms, 3rd ed. Boston, MA: Addison-Wesley, 1997.

[5] D. B. West, Introduction to Graph Theory, 2nd ed. Upper Saddle River, NJ: Prentice Hall, 2001.

[6] T. H. Cormen, "Topological sorting in directed acyclic graphs," Introduction to Algorithms, Cambridge, MA: MIT Press, 2009, ch. 22, pp. 601–605.

[7] F. S. Hillier dan G. J. Lieberman, Introduction to Operations Research, 9th ed. New York: McGraw-Hill, 2010.

[8] W.-K. Chen, Linear Networks and Systems. Belmont, CA: Wadsworth, 1993, pp. 123–135.

[9] B. Smith, "An approach to graphs of linear forms (Unpublished work style)," unpublished.

[10] E. H. Miller, "A note on reflector arrays (Periodical style—Accepted for publication)," IEEE Trans. Antennas Propagat., to be published.


## 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, 27 Desember 2024

ttd

Sabilul Huda
13523072