Penelitian Mandiri Sains Komputasi I

Update Progress

Mohammad Rizka Fadhli
Ikang
20921004@mahasiswa.itb.ac.id

28 October 2021

Contents

List of Figures

List of Tables

1 SEJARAH

1.1 Optimisasi

Optimisasi adalah **proses mencari nilai yang optimal** dari suatu masalah tertentu. Dalam matematika, optimisasi merujuk pada pencarian nilai minimal atau maksimal dari suatu *fungsi real*[1]. Notasi matematikanya dapat ditulis sebagai berikut:

Misalkan suatu fungsi f yang memetakan dari himpunan A ke bilangan *real*.

$$f : A \to \mathbb{R}$$

Cari suatu nilai $x_0 \in A$ sedemikian sehingga:

- $f(x_0) \leq f(x), \forall x \in A$ untuk proses **minimalisasi**.
- $f(x_0) \geq f(x), \forall x \in A$ untuk proses **maksimalisasi**.

Di dalam kalkulus, kita mengetahui salah satu pendekatan optimisasi di fungsi satu variabel bisa didapatkan dari turunan pertama yang bernilai **nol** (bisa berupa nilai maksimum atau minimum dari fungsi tersebut).

Nilai $x_0 \in [a, b]$ disebut minimum atau maksimum di f unimodal saat memenuhi:

$$\frac{d}{dx} f(x_0) = 0$$

Pierre De Fermat dan **Joseph-Louis Lagrange** adalah orang-orang yang pertama kali menemukan formula kalkulus untuk mencari nilai optimal. Sementara **Isaac Newton** dan **Johann C. F.Gauss** mengusulkan metode iteratif untuk mencari nilai optimal[2].

Salah satu bentuk optimisasi yakni *linear programming* dimulai oleh **Leonid Kantorovich** pada 1939. **Metode Simplex** merupakan salah satu metode penyelesaian optimisasi yang

[1] https://id.wikipedia.org/wiki/Optimisasi
[2] https://empowerops.com/en/blogs/2018/12/6/brief-history-of-optimization

terkenal, pertama kali diperkenalkan pada 1947 oleh **George Dantzig** sementara di tahun yang sama *Theory of Duality* diperkenalkan oleh **John von Neumann**.

1.2 Riset Operasi

Riset operasi adalah metode antar disiplin ilmu yang digunakan untuk menganalisa masalah nyata dan membuat keputusan untuk kegiatan operasional organisasi atau perusahaan[3].

Riset operasi dimulai pada era Perang Dunia II. Oleh karena peperangan, diperlukan suatu cara yang efektif untuk mengalokasikan *resources* yang ada sehingga pihak militer Inggris dan Amerika Serikat mengumpulkan ilmuwan-ilmuwan untuk mencari pendekatan yang saintifik dalam memecahkan masalah[4].

Pada tahun 1940, sekelompok *researchers* yang dipimpin oleh **PMS Blackett** dari ***the University of Manchester*** melakukan studi tentang **Sistem Radar Baru Anti Pesawat Terbang**. Kelompok *researchers* ini sering dijuluki sebagai **Kelompok Sirkus Blackett** (*Blackett's circus*). Julukan ini terjadi karena keberagaman latar belakang disiplin ilmu para *researchers* tersebut. Mereka terdiri dari disiplin ilmu fisiologi, matematika, astronomi, tentara, surveyor, dan fisika. Pada 1941, kelompok ini terlibat dalam penelitian radar deteksi kapal selam dan pesawat terbang. *Blackett* kemudian memimpin *Naval Operational Research* pada Angkatan Laut Kerajaan Inggris Raya. Prinsip-prinsip ilmiah yang digunakan untuk mengambil keputusan dalam suatu operasi dinamai sebagai **Riset Operasi**.

Saat Amerika Serikat mulai terlibat pada Perang Dunia II, prinsip riset operasi juga digunakan untuk berbagai operasi militer mereka. Kelompok riset operasi AS bertugas untuk menganalisis serangan udara dan laut tentara NAZI Jerman.

Selepas Perang Dunia II, penerapan riset operasi dinilai bisa diperluas ke dunia ekonomi, bisnis,*engineering*, dan sosial. Riset operasi banyak berkaitan dengan berbagai disiplin ilmu seperti matematika, statistika, *computer science*, dan lainnya. Tidak jarang beberapa pihak menganggap riset operasi itu *overlapping* dengan disiplin-disiplin ilmu tersebut.

[3] Pengantar Riset Operasi dan Optimisasi, KampusX: PO101
[4] Introduction to Operations Research, 7th Edition. Hillier / Lieberman hal. 1

Oleh karena tujuan utama dari aplikasi riset operasi adalah tercapainya **hasil yang optimal** dari semua kemungkinan perencanaan yang dibuat. Maka **pemodelan matematika dan optimisasi** bisa dikatakan sebagai disiplin utama dari riset operasi.

2 OPTIMISASI

2.1 Bahasan dalam Optimisasi

Bahasan dalam optimisasi dapat dikategorikan menjadi:

- Pemodelan masalah nyata menjadi masalah optimisasi.
- Pembahasan karakteristik dari masalah optimisasi dan keberadaan solusi dari masalah optimisasi tersebut.
- Pengembangan dan penggunaan algoritma serta analisis numerik untuk mencari solusi dari masalah tersebut.

2.2 Masalah Optimisasi

Masalah optimisasi adalah masalah matematika yang mewakili masalah nyata (*real*). Dari ekspresi matematika tersebut, ada beberapa hal yang perlu diketahui[5], yakni:

1. **Variabel** adalah suatu simbol yang memiliki banyak nilai dan nilainya ingin kita ketahui. Setiap nilai yang mungkin dari suatu variabel muncul akibat suatu kondisi tertentu di sistem.
2. **Parameter** di suatu model matematika adalah suatu konstanta yang menggambarkan suatu karakteristik dari sistem yang sedang diteliti. Parameter bersifat *fixed* atau *given*.
3. ***Constraints*** (atau kendala) adalah kondisi atau batasan yang harus dipenuhi. Kendala-kendala ini dapat dituliskan menjadi suatu persamaan atau pertaksamaan. Suatu masalah optimisasi dapat memiliki hanya satu kendala atau banyak kendala.

[5] Pengantar Riset Operasi dan Optimisasi, KampusX: PO101

4. ***Objective function*** adalah satu fungsi (pemetaan dari variabel-varibel keputusan ke suatu nilai di daerah *feasible*) yang nilainya akan kita minimumkan atau kita maksimumkan.

Ekspresi matematika dari model optimisasi adalah sebagai berikut:

Cari x yang meminimumkan $f(x)$ dengan kendala $g(x) = 0, h(x) \leq 0$ dan $x \in D$.

Dari ekspresi tersebut, kita bisa membagi-bagi masalah optimisasi tergantung dari:

1. Tipe variabel yang terlibat.
2. Jenis fungsi yang ada (baik *objective function* ataupun *constraints*).

2.3 Jenis-Jenis Masalah Optimisasi

Masalah optimisasi bisa dibagi dua menjadi dua kategori berdasarkan tipe *variables* yang terlibat[6], yakni:

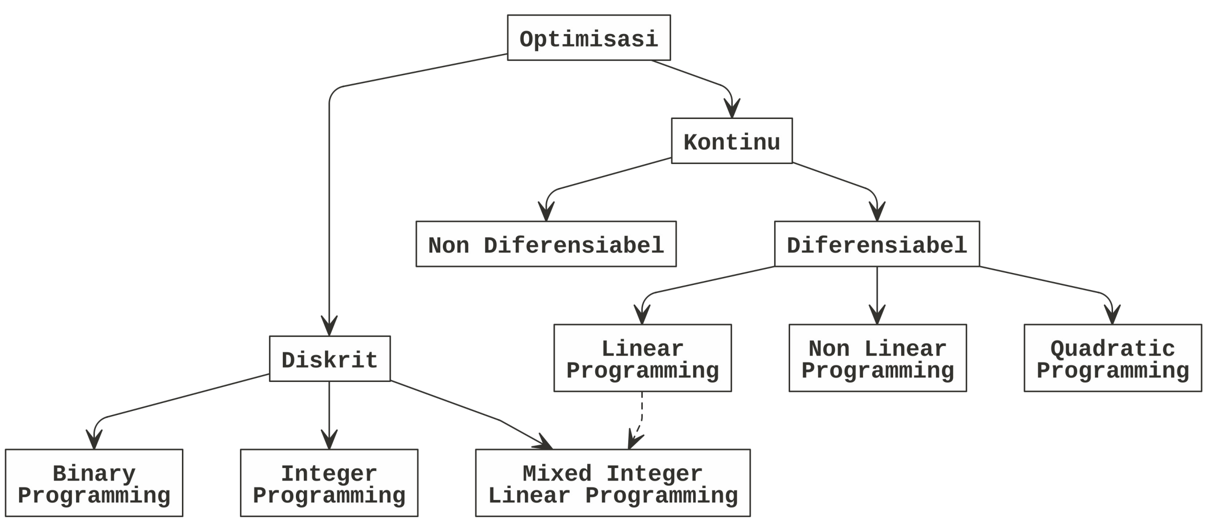

Figure 1: Optimisasi Berdasarkan Jenis Variabel

[6] Optimization problem. https://en.wikipedia.org/wiki/Optimization_problem

1. *Discrete Optimization*: merupakan masalah optimisasi di mana variabel yang terkait merupakan variabel diskrit, seperti *binary* atau *integer* (bilangan bulat). Namun pada masalah optimisasi berbentuk *mixed integer linear programming*, dimungkinkan suatu masalah optimisasi memiliki berbagai jeni variabel yang terlibat (integer dan kontinu sekaligus).
2. *Continuous Optimization*: merupakan masalah optimisasi di mana variabel yang terkait merupakan variabel kontinu (bilangan *real*). Pada masalah optimisasi jenis ini, fungsi-fungsi yang terlibat bisa diferensiabel atau tidak. Konsekuensinya adalah pada metode penyelesaiannya.

Selain itu, kita juga bisa membagi masalah optimisasi berdasarkan **kategori masalah** yang dihadapi sebagai berikut:

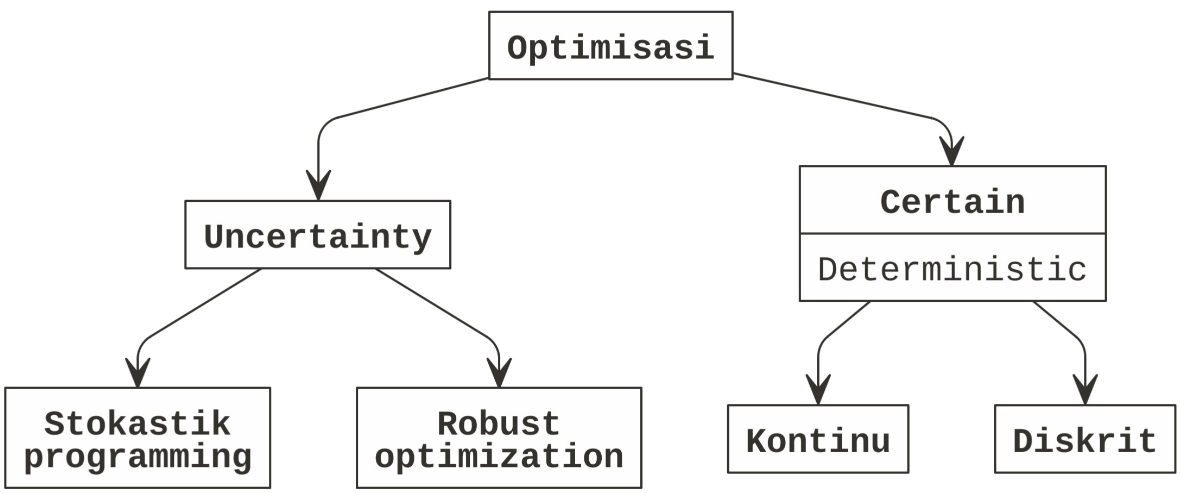

Figure 2: Optimisasi Berdasarkan Kategori Masalah

1. *Optimization under uncertainty*[7]; Pada beberapa kasus di dunia *real*, data dari masalah tidak dapat diketahui secara akurat karena berbagai alasan. Hal ini mungkin terjadi akibat:

    - Kesalahan dalam pengukuran, atau

[7] https://neos-guide.org/content/optimization-under-uncertainty

- Data melibatkan sesuatu di masa depan yang belum terjadi atau tidak pasti. Contoh: *demand* produk, harga barang, dan sebagainya.

2. *Deterministic optimization*;

- Model deterministik adalah model matematika di mana nilai dari semua parameter dan variabel yang terkandung di dalam model merupakan satu nilai pasti[8].
- Pendekatan deterministik memanfaatkan sifat analitik masalah untuk menghasilkan barisan titik yang konvergen ke solusi optimal[9].
- Semua algoritma perhitungan mengikuti pendekatan matematis yang ketat[10].

2.4 *Supplier Selection Problem*

Tema penelitian terkait *supplier selection problem* termasuk ke dalam masalah optimisasi deterministik yakni *mixed integer linear programming*, alasannya:

1. Parameter dan variabel yang terlibat merupakan suatu nilai pasti.
2. Variabel yang terlibat meliputi:
   - *Binary* karena melibatkan pengambilan keputusan *raw matt* dari *supplier* mana yang harus dipesan.
   - *Continuous* karena melibatkan angka kuantitas *raw matt* yang harus dipesan.
3. Fungsi *objective* dan *constraints* masih berupa *linear*.

[8] Pengantar Riset Operasi dan Optimisasi, KampusX: PO101
[9] https://www.hindawi.com/journals/mpe/2012/756023/
[10] https://link.springer.com/chapter/10.1007/978-3-642-31187-1_4

3 PENJELASAN SINGKAT JENIS OPTIMISASI

3.1 *Linear Programming*

Linear programming adalah bentuk metode optimisasi sederhana yang memanfaatkan relasi linear (semua fungsi dan *constraints* merupakan fungsi linear).

3.1.1 Contoh Masalah *Linear Programming*

Saya memiliki area parkir seluas 1.960 m^2. Luas rata-rata untuk mobil berukuran kecil adalah 4 m^2 dan mobil besar adalah 20 m^2. Daya tampung maksimum hanya 250 kendaraan, biaya parkir mobil kecil adalah Rp 7.000 per jam dan mobil besar adalah Rp 12.000 per jam. Jika dalam 1 jam area parkir saya terisi penuh dan tidak ada kendaraan yang pergi dan datang, maka berapa pendapatan maksimum yang bisa saya dapatkan dari tempat parkir itu?

Dari kasus di atas kita bisa tuliskan model matematikanya sebagai berikut:

Misal x_1 adalah mobil kecil dan x_2 adalah mobil besar.

$$max(7000x_1 + 12000x_2)$$

Dengan *constraints*:

$$4x_1 + 20x_2 \leq 1960$$

dan

$$x_1 + x_2 \leq 250$$

serta $x_1 \geq 0, x_2 \geq 0$.

3.2 *Integer Programming*

Integer programming adalah bentuk metode optimisasi di mana variabel yang terlibat merupakan bilangan bulat (*integer*). Jika fungsi-fungsi yang terkait merupakan *linear*, maka disebut dengan *integer linear programming.*

Sebagai contoh, variabel yang merupakan bilangan bulat adalah banyak orang.

3.2.1 Contoh *Integer Programming*

Jadwal Kebutuhan Tenaga Kesehatan Suatu rumah sakit membutuhkan tenaga kesehatan setiap harinya dengan spesifikasi berikut:

Table 1: Tabel Kebutuhan Nakes Harian

hari	Min Nakes Required	Max Nakes Required
Senin	24	29
Selasa	22	27
Rabu	23	28
Kamis	11	16
Jumat	16	21
Sabtu	20	25
Minggu	12	17

Di rumah sakit tersebut berlaku kondisi sebagai berikut:

1. Setiap nakes hanya diperbolehkan bekerja selama 5 hari berturut-turut dan harus libur selama 2 hari berturut-turut.
2. Tidak ada pemberlakuan *shift* bagi nakes.

Berapa banyak nakes yang harus dipekerjakan oleh rumah sakit tersebut? Bagaimana konfigurasi penjadwalannya?

Untuk memudahkan dalam mencari solusi permasalahan di atas, kita bisa membuat tabel ilustrasi berikut:

Table 2: Konfigurasi Penjadwalan Nakes

hari	Min Nakes Required	Max Nakes Required	x1	x2	x3	x4	x5	x6	x7
Senin	24	29	x			x	x	x	x
Selasa	22	27	x	x			x	x	x
Rabu	23	28	x	x	x			x	x
Kamis	11	16	x	x	x	x			x
Jumat	16	21	x	x	x	x	x		
Sabtu	20	25		x	x	x	x	x	
Minggu	12	17			x	x	x	x	x

Kolom $x_i, i = 1, 2, 3, 4, 5, 6, 7$ menandakan kelompok nakes yang perlu dipekerjaan pada hari-hari tertentu. Setiap nilai x_i tersebut merupakan **bilangan bulat positif** $x \geq 0, x \in \mathbb{Z}$.

Dari ilustrasi di atas, kita bisa membuat model optimisasinya sebagai berikut:

Objective Function

$$\min \sum_{i=1}^{7} x_i$$

Constraints

- Hari Senin: $24 \leq \sum x_i \leq 29, i \in \{1, 4, 5, 6, 7\}$.
- Hari Selasa: $22 \leq \sum x_i \leq 27, i \in \{1, 2, 5, 6, 7\}$.
- Hari Rabu: $23 \leq \sum x_i \leq 28, i \in \{1, 2, 3, 6, 7\}$.
- Hari Kamis: $11 \leq \sum x_i \leq 16, i \in \{1, 2, 3, 4, 7\}$.
- Hari Jumat: $16 \leq \sum x_i \leq 21, i \in \{1, 2, 3, 4, 5\}$.
- Hari Sabtu: $20 \leq \sum x_i \leq 25, i \in \{2, 3, 4, 5, 6\}$.
- Hari Minggu: $12 \leq \sum x_i \leq 17, i \in \{3, 4, 5, 6, 7\}$.

Kita juga perlu perhatikan bahwa $x_i \geq 0, i \in \{1, 2, 3, 4, 5, 6, 7\}$.

3.3 *Binary Programming*

Binary programming adalah bentuk metode optimisasi di mana variabel yang terlibat merupakan bilangan biner (0,1). Biasanya metode ini dipakai dalam masalah penjadwalan yang memerlukan prinsip *matching* antar kondisi yang ada.

3.3.1 Contoh *Binary Programming*

Jadwal Tatap Muka Terbatas Sekolah Beberapa minggu ke belakang, kasus harian Covid semakin menurun. Pemerintah mulai melonggarkan aturan PPKM yang mengakibatkan sekolah-sekolah mulai menggelar pengajaran tatap muka terbatas (PTMT) untuk siswanya secara *offline*.

Suatu sekolah memiliki kelas berisi 20 orang siswa. Mereka hendak menggelar PTMT dengana aturan sebagai berikut:

1. PTMT digelar dari Senin hingga Jumat (5 hari).
2. Dalam sehari, siswa yang boleh hadir dibatasi 4-8 orang saja.
3. Dalam seminggu, diharapkan siswa bisa hadir 2-3 kali.
4. Siswa yang hadir di selang sehari baru bisa hadir kembali.

Dari uraian di atas, kita bisa membuat model optimisasinya sebagai berikut:

Saya definisikan $x_{i,j} \in (0, 1)$ sebagai bilangan biner di mana $i \in \{1, 2, .., 20\}$ menandakan siswa dan $j \in \{1, 2, .., 5\}$ menandakan hari. Berlaku:

$$x_{i,j} = \begin{cases} 0, & \text{siswa i tidak masuk di hari j} \\ 1, & \text{siswa i masuk di hari j} \end{cases}$$

Objective Function

Tujuan utama kita adalah memaksimalkan siswa yang hadir.

$$\max \sum_{j=1}^{5} \sum_{i=1}^{20} x_{i,j}$$

Constraints

Dalam sehari, ada pembatasan jumlah siswa yang hadir.

$$4 \leq \sum_{i} x_{i,j} \leq 8, j \in \{1, 2, .., 5\}$$

Dalam seminggu, siswa hadir dalam frekuensi tertentu.

$$2 \leq \sum_{j} x_{i,j} \leq 3, i \in \{1, 2, .., 20\}$$

Ada jeda sehari agar siswa bisa masuk kembali.

$$x_{i,j} + x_{i,j+1} \leq 1$$

Jangan lupa bahwa $x_{i,j} \geq 0$.

3.4 *Mixed Integer Linear Programming*

Pada bagian sebelumnya, kita telah membahas masalah optimisasi dengan variabel berupa diskrit dan kontinu. Permasalahan *real* yang ada di kehidupan sehari-hari biasanya merupakan memiliki variabel yang *mixed* antara keduanya. Oleh karena itu, ada metode yang disebut dengan *mixed integer linear programming.* Pada masalah optimisasi tipe ini, *decision variables* yang terlibat bisa saja berupa *binary*, *integer*, dan *continuous* sekaligus.

3.4.1 Contoh *MILP*

Pemilihan dan Penentuan Item Produksi Suatu pabrik makanan dan minuman berencana untuk membuat tiga produk baru yang bisa diproduksi di dua *plants* yang berbeda.

Table 3: Tabel Runtime Item Produk per Plant (harian - dalam jam)

Produk	Runtime Plant 1	Runtime Plant 2
Item 1	3	4
Item 2	4	6
Item 3	2	2

Plant 1 memiliki maksimum *working hours* sebesar 30 jam perhari.

Plant 2 memiliki maksimum *working hours* sebesar 40 jam perhari.

Table 4: Tabel Profit dan Potensi Sales Item Produk

Produk	Profit per ton	Sales potential per ton
Item 1	5	7
Item 2	7	5
Item 3	3	9

Masalah timbul saat mereka harus memilih **dua dari tiga** produk baru tersebut yang harus di produksi. Selain itu, mereka juga harus memilih **satu dari dua** *plants* yang memproduksi *items* tersebut.

Misalkan saya definisikan:

- $x_i \geq 0, i = 1, 2, 3$ sebagai `berapa ton` yang harus diproduksi dari item i.
- $y_i \in [0, 1], i = 1, 2, 3$ sebagai *binary*.
  - Jika bernilai 0, maka produk i tidak dipilih.
  - Jika bernilai 1, maka produk i dipilih.
- $z \in [0, 1]$ sebagai *binary*.

- Jika bernilai 0, maka *plant* pertama dipilih.
- Jika bernilai 1, maka *plant* kedua dipilih.

Saya akan mendefinisikan suatu variabel *dummy* $M = 99999$ berisi suatu nilai yang besar. Kelak variabel ini akan berguna untuk *reinforce model* (metode pemberian *penalty*) agar bisa memilih *items* dan *plants* secara bersamaan.

Objective function dari masalah ini adalah memaksimalkan *profit*.

$$\max \sum_{i=1}^{3} x_i \times \text{profit}_i$$

Constraints dari masalah ini adalah:

Tonase produksi tidak boleh melebihi angka *sales potential* per items.

$$x_i \leq \text{sales potential}_i, i = 1, 2, 3$$

Kita akan memilih dua produk sekaligus menghitung tonase. Jika produk tersebut **dipilih**, maka akan ada angka tonase produksinya. Sebaliknya, jika produk tersebut **tidak dipilih**, maka tidak ada angka tonase produksinya.

$$x_i - y_i \times M \leq 0, i = 1, 2, 3$$

$$\sum_{i=1}^{3} y_i \leq 2$$

Kita akan memilih *plant* dari waktu produksinya.

$$3x_1 + 4x_2 + 2x_3 - M \times z \leq 30$$

$$4x_1 + 6x_2 + 2x_3 + M \times z \leq 40 + M$$

4 ALGORITMA PENYELESAIAN OPTIMISASI

Pada bagian ini kita akan membahas macam-macam algoritma yang digunakan untuk menyelesaikan masalah optimisasi.

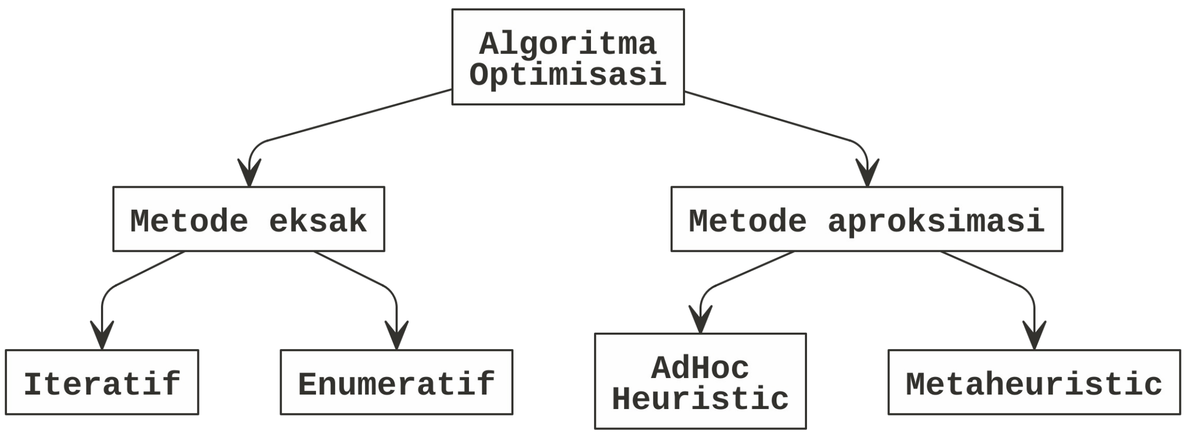

Figure 3: Algoritma Penyelesaian Optimisasi

Secara garis besar ada dua kelompok besar algoritma optimisasi, yakni:

1. *Exact method*,
2. *Approximate method*.

Perbedaan keduanya adalah pada **konsep atau pendekatan apa yang digunakan** untuk menyelesaikan masalah optimisasi. Kita akan bahas satu-persatu pada bagian selanjutnya.

Dalam beberapa kasus, kita bisa mendapatkan *exact method* bisa untuk menyelesaikan masalah optimisasi dengan efisien. Namun di kasus lain yang lebih kompleks tidak demikian. Kelemahan utama metode *exact* adalah pada waktu komputasinya yang relatif lebih lama.

4.1 *Exact Method*

Ciri khas dari *exact method* adalah metode ini menjamin penyelesaian yang optimal karena menggunakan penyelesaian analitis metode matematika.

4.2 *Approximate Method*

Ciri khas dari *approximate method* adalah metode ini tidak menjamin penyelesaian yang optimal karena bersifat *aproksimasi* atau pendekatan atau hampiran[11]. Oleh karena itu kita perlu melakukan definisi di awal **seberapa dekat** nilai **hampiran** tersebut bisa kita terima.

Metode ini bisa dibagi menjadi dua berdasarkan keterkaitannya dengan suatu masalah, yakni:

1. *Heuristic*, metode ini bersifat *problem dependent.* Artinya metode tersebut hanya bisa dipakai untuk jenis permasalahan tertentu.
    - Contoh: metode *nearest neighborhood* hanya bisa dipakai untuk menyelesaikan masalah dalam lingkup *travelling salesperson problem* (**TSP**).
2. *Meta heuristic*, metode ini bersifat *problem independent.* Artinya metode tersebut tidak tergantung dari jenis permasalahan tertentu. Contoh:
    - *Genetic algorithm.*
    - *Simulated annealing.*
    - *Spiral optimization.*
    - *Artifical bee colony algorithm.*

Namun demikian kedua metode ini bisa saling melengkapi dalam prakteknya.

5 METODE *SIMPLEX*

Metode *simplex* adalah salah satu metode yang paling umum digunakan dalam menyelesaikan permasalahan *linear programming.* Metode ini dikembangkan oleh seorang profesor matematika bernama George Dantzig[12] pada 1947 pasca perang dunia II. Sedangkan nama *simplex* diusulkan oleh Theodore Motzkin[13].

[11] https://www.math.unipd.it/~luigi/courses/metmodoc1819/m02.meta.en.partial01.pdf
[12] https://en.wikipedia.org/wiki/George_Dantzig
[13] https://en.wikipedia.org/wiki/Theodore_Motzkin

Metode *simplex* menggunakan prosedur aljabar[14]. Namun *underlying concept* dari metode ini adalah *geometric*.

5.1 Metode Simplex dengan Ilustrasi Geometris

Jika kita bisa memahami konsep geometrinya, kita bisa mengetahui bagaimana cara kerjanya dan kenapa metode ini sangat efisien.

Saya akan ambil satu contoh masalah optimisasi sederhana untuk memberikan ilustrasi bagaimana cara kerja metode ini.

Contoh Masalah Optimisasi Cari $x_1, x2$ yang max $(Z = 3x_1 + 5x_2)$ dengan *constraints*:

$$x1 \leq 4$$
$$2x_2 \leq 12$$
$$3x_1 + 2x_2 \leq 18$$
$$\text{serta } x_1 \geq 0, x_2 \geq 0$$

Masalah di atas jika dibuat grafiknya:

[14] Introduction to Operations Research, 7th Edition. Hillier / Lieberman hal. 109

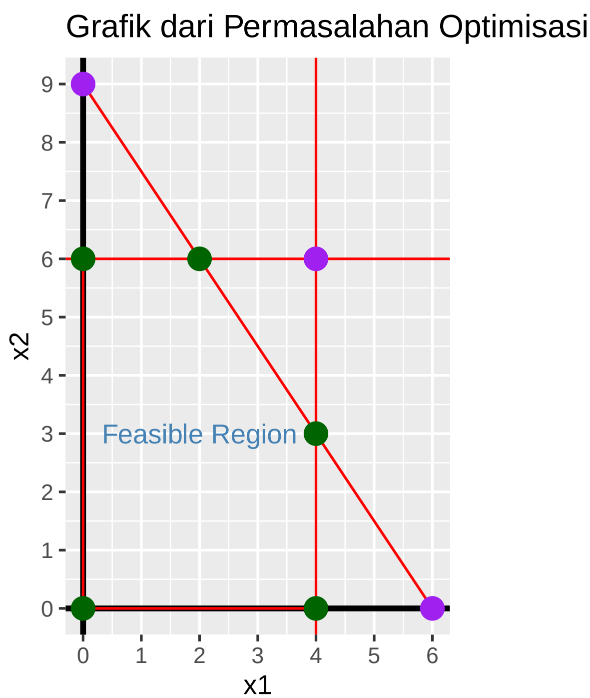

Figure 4: Grafik Permasalahan Optimisasi

Titik-titik hijau merupakan **beberapa titik** solusi yang *feasible* karena berada pada area penerimaan seluruh *constraints* yang ada. Titik hijau ini menjadi spesial karena berada pada perpotongan 2 garis *constraints*. Selanjutnya titik hijau ini akan didefinisikan sebagai **CPF** (*corner point feasible*).

> *For a linear programming problem with n decision variables, each of its corner-point solutions lies at the intersection of n constraint boundaries.*[15]

Sedangkan titik ungu merupakan titik solusi non *feasible* karena solusi yang ada tidak berlaku untuk semua *constraints*.

Table 5: Titik yang termasuk ke dalam CPF

Titik.ke	CPF
1	(0, 0)

[15] Introduction to Operations Research, 7th Edition. Hillier / Lieberman hal. 110

Titik.ke	CPF
2	(0, 6)
3	(2, 6)
4	(4, 3)
5	(4, 0)

Properties of CPF Solutions Untuk setiap permasalahan *linear programming* yang memiliki *feasible soultions* dan *feasible region* yang terbatas:

> **Property 1**: (a) If there is exactly one optimal solution, then it must be a CPF solution. (b) If there are multiple optimal solutions (and a bounded feasible region), then at least two must be adjacent CPF solutions.
>
> **Property 2**: There are only a finite number of CPF solutions.
>
> **Property 3**: If a CPF solution has no adjacent CPF solutions that are better (as measured by Z), then there are no better CPF solutions anywhere. Therefore, such a CPF solution is guaranteed to be an optimal solution (by Property 1), assuming only that the problem possesses at least one optimal solution (guaranteed if the problem possesses feasible solutions and a bounded feasible region).

Untuk mulai melakukan metode simplex kita perhatikan kembali grafik di atas. Kita bisa temukan beberapa pasang **CPF** berbagi *constraint* yang sama satu sama lain.

Sebagai contoh:

1. CPF_1 dan CPF_2 berbagi *constraint* yang sama, yakni saat $x_1 \geq 0$.

2. CPF_2 dan CPF_3 berbagi *constraint* yang sama, yakni saat $x_2 \leq 6$.

Definisi umum:

> *For any linear programming problem with n decision variables, two CPF solutions are* ***adjacent*** *to each other if they share $n - 1$ constraint boundaries. The two adjacent CPF solutions are connected by a line segment that lies on these same shared constraint boundaries. Such a line segment is referred to as an* ***edge*** *of the feasible region.*

Feasible region di atas memiliki 5 *edges* di mana setiap 2 *edges* memotong / memunculkan **CPF**. Setiap **CPF** memiliki 2 **CPF** lainnya yang *adjacent.*

Table 6: Adjacent CPF

Titik.ke	CPF	Adjacent.CPF
1	(0, 0)	(0, 6) dan (4, 0)
2	(0, 6)	(2, 6) dan (0, 0)
3	(2, 6)	(4, 3) dan (0, 6)
4	(4, 3)	(4, 0) dan (2, 6)
5	(4, 0)	(0, 0) dan (4, 3)

CPF pada kolom pertama *adjacent* terhadap dua **CPF** di kolom setelahnya tapi kedua **CPF** tersebut tidak saling *adjacent* satu sama lain.

> ***Optimality test:*** *Consider any linear programming problem that possesses at least one optimal solution. If a CPF solution has no adjacent* ***CPF*** *solutions that are better (as measured by Z), then it must be an optimal solution.*

Berdasarkan *optimality test* tersebut, kita bisa mencari solusi optimal dari **CPF** dengan cara mengambil **initial CPF** untuk dites secara rekursif.

- **STEP 1** Pilih *initial* **CPF**, misal $(0,0)$. Kita akan hitung nilai $Z(0,0) = 0$. Bandingkan dengan *adjacent* **CPF**-nya, yakni $Z(0,6) = 30$ dan $Z(4,0) = 12$.
- **STEP 2** Oleh karena $Z(0,6)$ memiliki nilai tertinggi, maka kita akan pilih titik ini di iterasi pertama. Kita akan bandingkan terhadap *adjacent* **CPF**-nya, yakni: $Z(2,6) = 36$. Perhatikan bahwa *adjacent* **CPF** $(0,0)$ sudah kita evaluasi pada langkah sebelumnya.
- **STEP 3** Oleh karena $Z(2,6)$ memiliki nilai tertinggi, maka kita akan pilih titik ini di iterasi kedua. Kita akan bandingkan terhadap *adjacent* **CPF**-nya, yakni: $Z(4,3) = 27$. Kita dapatkan bahwa titik $(2,6)$ menghasilkan Z tertinggi.

Kesimpulan: $(2,6)$ merupakan titik yang bisa memaksimumkan Z.

5.1.1 ***Flowchart*** Metode Simplex dari Contoh Masalah

Secara garis besar, *flowchart* dari metode simplex untuk masalah di atas adalah:

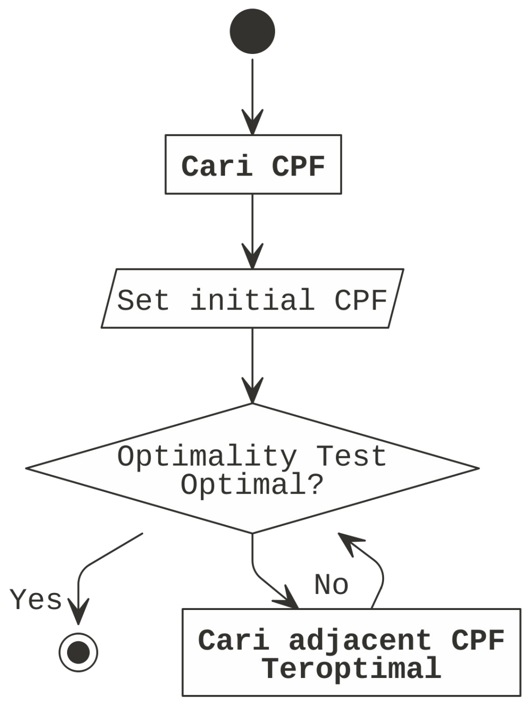

Figure 5: Algoritma Metode Simplex

Algoritma di atas akan sangat mudah dilakukan saat kita berhadapan dengan masalah optimisasi dengan 2 *decision variables* (atau 3 *decision variables*). Pada contoh di atas ada x_1, x_2.

Bagaimana jika masalah yang dihadapi memiliki banyak *decision variables*?

Tentunya kita tidak bisa melakukan analisa secara visual seperti di atas. Namun kita bisa menggunakan bantuan aljabar dan operasi baris elementer untuk menemukan solusi yang optimal.

5.2 Metode Simplex dengan Operasi Matriks

Suatu masalah optimisasi bisa kita tulis dalam bentuk matriks sehingga bisa diselesaikan dengan melakukan *operasi baris elementer*.

5.2.1 Contoh Penyelesaian Masalah Optimisasi dengan *Simplex*

Cari x, y sehingga $\max (P = 5x + 4y)$ dengan *constraints*:

$$3x + 5y \leq 78$$
$$4x + y \leq 36$$
$$\text{serta } x \geq 0, y \geq 0$$

Untuk menyelesaikannya, kita perlu menambahkan u, w sebagai variabel pembantu. Fungsi objectif P juga harus diubah (dipindah sisi namun P tetap positif).

$$3x + 5y + u \leq 78$$
$$4x + y + w \leq 36$$
$$-5x - 4y + P = 0$$

Setelah itu kita buat matriks (dalam hal ini saya akan buatkan tabelnya) sebagai berikut:

Table 7: Initial Condition Bentuk Matriks Simplex

x	y	u	w	P	b
3	5	1	0	0	78
4	1	0	1	0	36
-5	-4	0	0	1	0

STEP 1 Kita akan pilih kolom yang memiliki nilai **negatif terbesar** pada baris terakhir, yakni kolom x. Selanjutnya kita akan pilih baris mana yang akan menjadi pivot dengan cara menghitung rasio $\frac{b}{x}$ untuk semua baris dan memilih baris dengan **rasio terendah**.

Table 8: Pemilihan Baris Pivot

x	y	u	w	P	b	rasio
3	5	1	0	0	78	26
4	1	0	1	0	36	9
-5	-4	0	0	1	0	0

STEP 2 Kita akan buat baris 2 kolom x menjadi bernilai 1, caranya dengan melakukan OBE seperti: $Row_2 = \frac{Row_2}{4}$.

Table 9: OBE Iterasi 1

x	y	u	w	P	b
3	5.00	1	0.00	0	78
1	0.25	0	0.25	0	9
-5	-4.00	0	0.00	1	0

STEP 3 Sekarang tujuan kita selanjutnya adalah membuat kolom x baris 1 dan 3 menjadi bernilai **nol**. Caranya adalah:

$$Row_1 = Row_1 - 3Row_2$$

$$Row_3 = Row_3 + 5Row_2$$

Table 10: OBE Iterasi 2

x	y	u	w	P	b
0	4.25	1	-0.75	0	51
1	0.25	0	0.25	0	9
0	-2.75	0	1.25	1	45

STEP 4 Kita akan lakukan hal yang sama pada *step 1*, yakni memilih kolom dengan negatif terbesar. Yakni kolom y. Lalu kita akan hitung rasio setiap baris dan akan memilih rasio paling rendah.

Table 11: Pemilihan Baris Pivot Kembali

x	y	u	w	P	b	rasio
0	4.25	1	-0.75	0	51	12.00000
1	0.25	0	0.25	0	9	36.00000
0	-2.75	0	1.25	1	45	-16.36364

STEP 5 Maka kita akan pilih baris `1` menjadi pivot. Kolom y pada baris `1` harus bernilai `1` sehingga kita harus membuat $Row_1 = \frac{4Row_1}{17}$.

Table 12: OBE Iterasi 3

x	y	u	w	P	b
0	1.00	0.2352941	-0.1764706	0	12
1	0.25	0.0000000	0.2500000	0	9
0	-2.75	0.0000000	1.2500000	1	45

STEP 6 Kita akan buat klom y di baris 2 dan 3 menjadi **nol** dengan cara:

$$Row_2 = Row_2 - \frac{Row_1}{4}$$

$$Row_3 = Row_3 + \frac{11Row_1}{4}$$

Table 13: OBE Iterasi 4

x	y	u	w	P	b
0	1	0.2352941	-0.1764706	0	12
1	0	-0.0588235	0.2941176	0	6
0	0	0.6470588	0.7647059	1	78

Dari tabel terakhir di atas, kita bisa menuliskan $x = 6, y = 12$ dan nilai $\max{(P)} = 78$. Bagaimana dengan nilau u dan w? Karena tidak ada nlai 1 ditemukan pada kolom variabel tersebut, kita bisa simpulkan bahwa $u = 0, w = 0$.

5.3 *Post-Optimality Analysis*

Post-optimality analysis adalah analisa yang dilakukan pasca kita telah menemukan *optimal solution* dari hasil perhitungan. Contohnya kita bisa melakukan **reoptimisasi**.

5.4 *Sensitivity Analysis*

Salah satu proses dalam membuat model optimisasi adalah *parameter estimation*. Ada kalanya perubahan data mengakibatnya berubahnya suatu parameter. *Sensitivity analysis* bertujuan untuk mengidentifikasi parameter yang sensitif (parameter yang harus dihitung dengan baik untuk menghindari kesalahan saat mencari solusi optimal).

6 References