Optimisasi Perencanaan Diet Pemain Sepakbola Pria Tingkat Profesional Usia 24-30 Tahun dengan Alokasi Dana Minimum pada Periode Pramusim

Lie Reubensto - 2540124633

School of Computer Science, Universitas Bina Nusantara

Email: lie.reubensto@binus.ac.id

ABSTRAK

Sepakbola menjadi salah satu olahraga yang paling terkenal dan kompetitif di dunia. Banyak klub sepakbola mau mempersiapkan para pemainnya pada periode pramusim sebelum masuk ke musim liga yang cukup panjang. Kesehatan dan kebugaran menjadi kunci untuk performa maksimal dari pemain saat di lapangan. Untuk itu, klub harus bisa mengatur asupan nutrisi para pemainnya dan biaya yang dibutuhkan untuk mendapatkan bahan-bahan nutrisi tersebut. Tujuan dari penelitian ini adalah untuk mencari keberadaannya solusi optimal dengan alokasi dana seminimum mungkin bagi klub dan asupan nutrisi para pemain bisa terpenuhi. Metode penelitian yang dipakai adalah metode penelitian kuantitatif dengan bantuan metode *Integer Linear Programming* (ILP) dan pendekatan *Gomory* (*Cutting Plane Method*) sebagai metode perhitungan. Untuk membantu perhitungan, metode *Simplex* dan *Dual Simplex* juga digunakan. Hasil dari penelitian adalah solusi optimal bisa ditemukan dengan metode-metode tersebut dan dari data yang dipakai untuk membuat *constraint*, di mana beberapa bahan harus dibeli dalam jumlah yang bulat. Beberapa data seperti data harga yang digunakan merupakan data buatan. Program berbahasa Python digunakan untuk memvalidasi hasil. Dengan demikian, dengan metode-metode yang sudah disebut sebelumnya, solusi optimal dapat dicari, di mana biaya untuk pembelian bahan-bahan nutrisi bisa diminimalisir dan asupan nutrisi para pemain tetap bisa terpenuhi.

Kata kunci: Sepakbola, Alokasi Dana, Minimum, Optimisasi, Nutrisi, Pramusim, Integer Linear Programming, Gomory, Simplex, Big-M, Dual Simplex

BAB I
PENDAHULUAN

1.1 LATAR BELAKANG

Sepakbola merupakan salah satu olahraga yang paling terkenal dan kompetitif di dunia. Performa seorang pemain sepakbola di lapangan tidak hanya ditentukan oleh kemampuan teknis dan strategi, tetapi juga sangat dipengaruhi oleh kondisi fisik dan kesehatan pemain. Salah satu faktor kunci yang menentukan kondisi fisik dan kesehatan pemain adalah nutrisi yang mereka konsumsi. Dalam konteks ini, perencanaan diet yang tepat menjadi sangat penting, terutama selama periode pramusim di mana pemain menjalani latihan intensif untuk mempersiapkan diri menghadapi musim kompetisi (liga) yang panjang.

Pramusim adalah periode penting bagi pemain sepakbola profesional. Pada periode ini, pemain dituntut untuk mencapai kondisi fisik terbaik melalui berbagai program latihan berat. Nutrisi yang memadai selama periode ini sangat penting untuk mendukung pemulihan, meningkatkan performa, dan mencegah cedera. Namun, tantangan yang sering dihadapi adalah bagaimana merancang perencanaan diet yang efektif dan efisien, khususnya dengan alokasi dana yang terbatas. Pengeluaran untuk nutrisi sering kali menjadi beban signifikan bagi sebuah klub, terutama bagi klub dengan sumber daya finansial terbatas.

Penelitian ini bertujuan untuk mengembangkan model optimisasi perencanaan diet bagi pemain sepakbola pria profesional berusia 24-30 tahun dengan alokasi dana minimum, terutama selama periode pramusim. Usia 24-30 tahun dipilih karena pada rentang usia ini pemain umumnya berada pada puncak performa fisik (prima) mereka. Model ini diharapkan dapat memberikan solusi yang efektif dan praktis bagi klub dalam mengelola sumber daya mereka, sekaligus memastikan pemain mendapatkan asupan nutrisi yang sesuai dengan kebutuhan fisiologis dan tuntutan aktivitas fisik mereka.

Secara umum, penelitian ini akan mengintegrasikan aspek-aspek nutrisi olahraga dengan teknik optimisasi *Integer Linear Programming* (ILP). Fokus utama adalah memastikan bahwa diet yang dirancang memenuhi semua kebutuhan gizi yang diperlukan oleh pemain sepakbola dan meminimalkan biaya total. Metodologi yang digunakan akan mencakup pengumpulan data kebutuhan nutrisi pemain, harga bahan makanan, serta penerapan algoritma optimisasi untuk menemukan solusi diet yang optimal.

Hasil dari penelitian ini diharapkan tidak hanya memberikan kontribusi pada bidang nutrisi olahraga dan manajemen klub sepakbola, tetapi juga dapat diaplikasikan pada berbagai konteks lain di mana perencanaan diet dengan anggaran terbatas diperlukan. Pada akhirnya, penelitian ini bertujuan untuk mendukung tercapainya performa maksimal pemain sepakbola melalui pendekatan ilmiah dalam perencanaan diet yang berkelanjutan dan ekonomis.

1.2 TINJAUAN PUSTAKA

Perencanaan diet untuk para atlet telah menjadi fokus banyak penelitian dalam beberapa tahun terakhir. Nutrisi yang tepat sangat penting untuk mendukung performa para atlet, terutama pemain sepakbola profesional, selama periode latihan intensif seperti periode pramusim. Menurut Burke dan Deakin (2015), nutrisi yang baik dapat membantu dalam pemulihan otot, menjaga energi, dan meningkatkan daya tahan tubuh. Nutrisi yang optimal juga berperan dalam pencegahan cedera dan penyakit, yang dapat mempengaruhi kinerja pemain (Maughan, 2006).

Model optimisasi diet telah banyak digunakan dalam berbagai konteks untuk meminimalkan biaya dan memaksimalkan pemasukan. Di kasus ini, konteksnya adalah asupan nutrisi. Dalam konteks ini, *Integer Linear Programming* (ILP) menjadi metode yang sangat efektif. ILP memungkinkan penyertaan variabel *integer* (bilangan bulat) dalam model, yang sangat berguna dalam perencanaan diet di mana beberapa komponen (misalnya, unit makanan atau porsi) harus berupa bilangan bulat.

Dalam bidang sepakbola, beberapa penelitian telah mengeksplorasi kebutuhan nutrisi khusus untuk pemain. Pemain sepakbola memerlukan asupan karbohidrat yang memadai untuk

mempertahankan energi selama pertandingan dan latihan (Burke, Hawley, Wong & Jeukendrup, 2011). Selain itu, protein juga sangat penting untuk pemulihan otot dan pertumbuhan (Tipton & Wolfe, 2004). Penelitian lain menunjukkan bahwa asupan mikronutrien, seperti zat besi, kalsium, dan vitamin D, sangat penting untuk kesehatan tulang dan otot (Oliveira dkk., 2017).

Namun, tantangan utama dalam perencanaan diet untuk pemain sepakbola adalah mengelola biaya yang terkait dengan bahan makanan berkualitas tinggi. Dikutip dari dw.com (2018), klub bisa membutuhkan dana yang cukup besar untuk mempertahankan diet para pemainnya. Klub dengan anggaran terbatas akan merasa kesulitan menyediakan nutrisi yang memadai bagi pemain mereka. Oleh karena itu, optimisasi biaya dalam perencanaan diet menjadi sangat penting untuk memastikan semua pemain mendapatkan nutrisi yang mereka butuhkan tanpa melebihi anggaran yang tersedia.

BAB II
TEOREMA DASAR

2.1 TEORI

Untuk menyelesaikan masalah yang telah disebut sebelumnya, metode *Integer Linear Programming* (ILP) akan digunakan. Lebih tepatnya, ILP dengan pendekatan *Gomory* (*Cutting Plane Method*). Metode ILP dipakai untuk menemukan solusi optimal (maksimum/minimum tergantung fungsi tujuan) yang memenuhi syarat-syarat (kendala/*constraint*) yang ada dan memberikan hasil variabel penentu bilangan bulat untuk satu (*sole ILP*), beberapa (*mixed ILP*), atau bahkan semua (*pure ILP*) variabel penentu yang ada. Dengan menggunakan pendekatan *Gomory*, suatu skala *Gomory* dibuat untuk membantu menemukan jawaban yang bulat (*integer*). Skala ini akan berhenti dibentuk jika variabel-variabel penentu yang seharusnya *integer* sudah dalam bentuk *integer* semua dan tetap terus-menerus dibentuk apabila variabel-variabel tersebut masih dalam bentuk *integer*. Jika skala Gomory terbentuk terus-menerus tanpa henti, kemungkinan besar bahwa masalah tidak memiliki solusi dengan variabel penentu *integer*.

Untuk menerapkan metode tersebut, perlu juga bantuan metode *Simplex Big-M* dan *Dual Simplex* untuk proses perhitungan. Metode *Simplex Big-M* digunakan saat bagian awal untuk menemukan solusi optmal sejati (di mana nilai beberapa variabel penentunya bisa saja dalam bentuk desimal). Metode *Dual Simplex* dipakai saat perhitungan *Cutting Plane Method*, di mana skala *Gomory* akan terbentuk (menambah baris dan kolom dari tabel) untuk menentukan variabel mana saja yang perlu menjadi variabel *basis* (variabel yang berpengaruh dalam pembuatan solusi optimal) supaya nilai dari variabel-variabel penentu yang diinginkan berbentuk *integer*.

Untuk cara kerja metode *Simplex Big-M* dan *Dual Simplex* sendiri cukup sederhana. Secara umum, kedua metode harus menentukan baris kunci dan kolom kunci pada tabel berdasarkan aturan yang ada untuk masing-masing metode. Kemudian, variabel yang menjadi kolom kunci (pada tabel) akan menjadi *entering variable* (masuk ke variabel basis) dan variabel yang menjadi baris kunci (pada tabel) akan menjadi *leaving variable* (keluar dari variabel basis). Lalu, semua nilai di baris kunci diperbarui dengan nilai yang sedemikian sehingga memperbarui nilai angka kunci (angka yang berada di baris dan kolom kunci secara bersamaan) menjadi satu (1). Setelah itu, nilai-nilai di baris lain juga diperbarui berdasarkan nilai lama

mereka dikurangi hasil perkalian antara nilai kolom kunci (versi lama) baris di mana nilai tersebut berada dengan nilai baris kunci (versi baru) kolom di mana nilai tersebut berada.

2.2 RUMUS

Misalkan ada sebuah kasus seperti ini:

$Fungsi\ Tujuan$:
$Minimum\ Z = c_1x_1 + c_2x_2$

$Kendala$:
$a_{11}x_1 + a_{12}x_2 \leq b_1$
$a_{21}x_1 + a_{22}x_2 \geq b_2$
$x_1, x_2 \geq 0\ dan\ x_1\ integer$

$Keterangan$:
$Z = hasil\ fungsi\ tujuan$
$x = variabel\ penentu$
$c = konstanta\ tujuan$
$a = konstanta\ kendala$
$b = konstanta\ hasil\ kendala$

Untuk lanjut, bentuk diatas pelu diubah terlebih dahulu menjadi bentuk simpleks:

$Fungsi\ Tujuan$:
$Z - c_1x_1 - c_2x_2 - 0s_1 - 0S_1 - 0A_1 = 0 \rightarrow Tanpa\ Big - M$
$Z - c_1x_1 - c_2x_2 - 0s_1 - 0S_1 - MA_1 = 0 \rightarrow Dengan\ Big - M$

$Kendala$:
$a_{11}x_1 + a_{12}x_2 + s_1 = b_1$
$a_{21}x_1 + a_{22}x_2 - S_1 + A_1 = b_2$
$x_1, x_2 \geq 0\ dan\ x_1\ integer$
$s_1, S_1, A_1 \geq 0$

$Keterangan$:
$s = variabel\ slack$
$S = variabel\ surplus$
$A = variabel\ artificial$

Untuk keberadaan variabel-variabel baru ini, ada aturannya untuk setiap tipe kendala. Jika kendalanya memiliki tanda $\leq$, hanya perlu memasukkan variabel *slack* (+). Jika kendalanya memiliki tanda $\geq$, perlu memasukkan variabel *surplus* (-) dan *artificial* (+). Jika kendalanya memiliki tanda =, hanya perlu measukkan variabel *artificial* (+).

Setiap variabel tersebut memiliki arti dan tujuannya tersendiri. Variabel *slack* digunakan ketika kendala memiliki tanda $\leq$ karena variabel tersebut digunakan untuk merepresentasi sumber daya yang tidak terpakai. Variabel *surplus* dan *artificial* digunakan bersamaan saat kendala memiliki tanda $\geq$ karena varibel-variabel tersebut digunakan untuk merepresentasi sumber daya yang melebihi batasan nilai hasil kendala. Variabel *artificial* dipakai sendiri ketika kendala memiliki tanda $=$ karena variabel tersebut digunakan sebagai variabel pembantu saja.

Setelah menentukan bentuk simpleksnya, kita akan menggunakan metode *Simplex Big-M* untuk mencari solusi optimal sejati:

Variabel Basis	x_1	x_2	s_1	S_1	A_1	Nilai Kanan (NK)
Z	$-c_1$	$-c_2$	0	0	-M	0
s_1	a_{11}	a_{12}	1	0	0	b_1
A_1	a_{21}	a_{22}	0	-1	1	b_2

Untuk metode *Simplex Big-M*, pertama kita harus menghilangkan terlebih dahulu nilai -M pada baris Z kolom variabel *artifical*. Ini dilakukan dengan memperbarui baris tersebut dengan bantuan baris lain. Lebih jelasnya, kita akan melakukan operasi baris untuk menghilangkan nilai -M tersebut dari kolom variabel *artificial*. Berikut adalah contohnya:

Variabel Basis	x_1	x_2	s_1	S_1	A_1	Nilai Kanan (NK)
Z	$-c_1 + Ma_{21}$	$-c_2 + Ma_{22}$	0	-M	0	Mb_2
s_1	a_{11}	a_{12}	1	0	0	b_1
A_1	a_{21}	a_{22}	0	-1	1	b_2

Operasi: $baris\ Z_{baru} = baris\ Z_{lama} + (M \cdot baris\ A_1)$

Selanjutnya, kita tentukan kolom kunci. Ini dilakukan dengan awalnya memperhatikan baris Z. Untuk kasus minimasi *Big-M*, pilihlah kolom (kecuali kolom NK) di mana nilai pada baris Z memiliki nilai (termasuk koefisien M) terbesar (misal pada baris Z, nilai kolom $x_1 = 7M - 4$ dan nilai kolom $x_2 = 6M - 9$, maka kita akan memiliki kolom x_1 sebagai kolom kunci). Untuk baris kunci, kita harus membagi nilai kanan (NK) dengan nilai kolom kunci di masing-masing baris. Hasilnya kita sebut sebagai indeks (kolom ini hanya sebagai kolom pembantu saja tiap iterasi dan tidak dipakai lagi jika baris kunci sudah ditemukan pada iterasi tersebut). Baris yang memiliki nilai indeks terkecil (harus positif dan > 0) merupakan baris kunci (misal baris s_1). Baris Z tidak bisa menjadi baris kunci. Jika dalam penentuan kolom dan baris kunci, ada nilai terkecil yang sama (dua/lebih kolom memiliki nilai pada baris Z yang sama atau dua/lebih baris punya nilai indeks yang sama), kita bebas mau menggunakan baris dan/atau kolom kunci yang mana dari kumpulan baris/kunci tersebut (namun pakai salah satu saja untuk masing-masing baris dan kolom kunci). Indeks pada baris Z tidak perlu dihitung karena variabel Z tidak mungkin menjadi *leaving variable*.

Variabel Basis	x_1	x_2	s_1	S_1	A_1	Nilai Kanan (NK)	Indeks
Z	$-c_1 + Ma_{21}$	$-c_2 + Ma_{22}$	0	-M	0	0	-
s_1	a_{11}	a_{12}	1	0	0	b_1	$\frac{b_1}{a_{11}}$
A_1	a_{21}	a_{22}	0	-1	1	b_2	$\frac{b_2}{a_{21}}$

Setelah menentukan baris dan kolom kunci, kita perbarui baris kunci supaya angka kunci (angka yang berada di baris dan kolom kunci secara bersamaan, disini angka kuncinya adalah a_{11}) menjadi 1. Selain itu, kita juga harus memasukkan variabel yang menjadi kolom kunci ke baris kunci di variabel basis (pada kasus ini, karena x_1 menjadi *entering variable* dan s_1 menjadi *leaving variable*, x_1 menggantikan s_1 di variabel basis). Sehingga tabel menjadi seperti ini:

Variabel Basis	x_1	x_2	s_1	S_1	A_1	Nilai Kanan (NK)
Z	$-c_1 + Ma_{21}$	$-c_2 + Ma_{22}$	0	-M	0	0
x_1	1	$\frac{a_{12}}{a_{11}}$	$\frac{1}{a_{11}}$	0	0	$\frac{b_1}{a_{11}}$
A_1	a_{21}	a_{22}	0	-1	1	b_2

Bisa dilihat bahwa setiap nilai baris kunci dibagi dengan nilai a_{11} supaya angka kunci bisa menjadi 1. Selanjutnya, untuk nilai pada baris lain diperbarui dengan rumus berikut:

$$Nilai\ Baru = Nilai\ lama - (NK_{lama} \times NB_{baru})$$

Di mana:

$$NK_{lama} = nilai\ kolom\ kunci\ (sebelum\ diperbarui)\ pada\ baris\ nilai\ lama\ berada$$

$$NB_{baru} = nilai\ baris\ kunci\ (setelah\ diperbarui)\ pada\ kolom\ nilai\ lama\ berada$$

Contoh:

Misal kita ingin memperbarui nilai $-c_2$ pada baris Z kolom x_2, maka nilai barunya adalh:

$$NB = -c_2 + Ma_{22} - \left([-c_1 + Ma_{21}] \times \frac{a_{12}}{a_{11}}\right)$$

Semua nilai pada semua baris dari kolom x_1 (kolom pertama) sampai kolom NK (kolom terakhir) harus diperbarui. Setelah itu, iterasi pertama untuk metode *Simplex* sudah selesai.

Setelah itu, kita harus cek apakah semua nilai (kecuali NK) di baris Z sudah lebih kecil dari atau sama dengan 0 (≤ 0) semua (termasuk koefisien M dan koefisien M hanya ditemukan di baris Z kolom variabel *artificial*). Jika belum, harus lanjut ke iterasi berikutnya (mencari baris dan kolom kunci lagi dan seterusnya). Jika sudah, hasil yang didapat merupakan hasil paling optimal dan kemungkinan besar nilai variabel penentunya berbentuk desimal.

Setelah mendapatkan hasil paling optimal, kita pindah ke *Cutting Plane Method*. Sekali lagi, metode ini digunakan untuk mencari solusi seoptimal-optimalnya dengan salah satu, beberapa, atau semua variabel penentu harus dalam bentuk *integer*. Dibilang seoptimal-optimalnya karena hasil dari metode ini tidak mungkin lebih optimal dibanding hasil metode *Simplex Big-M* (karena hasil dari *Simplex Big-M* merepresentasi hasil yang paling optimal).

Karena kasus kita adalah minimasi, hasil dari metode ini tidak mungkin memiliki hasil nilai Z yang lebih rendah dibanding hasil dari metode *Simplex Big-M*.

Untuk memulai metode *Cutting Plane*, kita harus menentukan terlebih dahulu variabel mana yang harus diproses. Untuk contoh kasus ini, hanya x_1 yang harus *integer*. Namun, jika ada lebih dari satu variabel penentu yang harus *integer*, kita harus lihat terlebih dahulu nilai kanan (NK) pada tabel hasil *Simplex Big-M* dari sebelumnya. Untuk semua variabel basis (kecuali baris Z), variabel yang memiliki sisa pecahan yang paling besar ketika bentuknya diubah menjadi pecahan campuran akan diproses terlebih dahulu. Misalnya ada tabel hasil *Simplex Big-M* seperti ini:

Variabel Basis	x_1	x_2	s_1	S_1	A_1	Nilai Kanan (NK)
Z	*	*	*	*	*	*
x_1	*	*	*	*	*	23/8 = 2 + 7/8
x_2	*	*	*	*	*	11/2 = 5 + 1/2

Ketika kedua variabel penentu x_1 dan x_2 harus *integer*, kita harus pilih x_1 terlebih dahulu untuk diproses karena $\frac{7}{8} > \frac{1}{2}$ (namun, di contoh kasus kita, hanya x_1 yang harus *integer*, sehingga jika hanya ada satu variabel penentu yang harus *integer*, kita tidak perlu mebanding-banding seperti ini, langsung saja memproses variabel tersebut). Jika nilai sisa pecahannya sama, bebas pilih variabel penentu yang mana yang ingin diproses terlebih dahulu. Setelah itu, kita akan membuat suatu skala Gomory (ibaratnya sebuah variabel dan kendala baru). Untuk membuat skala ini, kita harus mengambil persamaan dari variabel penentu yang ingin diproses terlebih dahulu dari sebelumnya dan mencari nilai sisa pecahan dari semua koefisien di persamaan tersebut. Misalnya ada tabel hasil *Simplex Big-M* seperti ini:

Variabel Basis	x_1	x_2	s_1	S_1	A_1	Nilai Kanan (NK)
Z	*	*	*	*	*	*
x_1	1	0	-11/6	0	0	23/8 = 2 + 7/8
x_2	*	*	*	*	*	11/2 = 5 + 1/2

Terlihat bahwa persamaan x_1 menjadi: $x_1 - \frac{11}{6}s_1 = \left(2 + \frac{7}{8}\right) \rightarrow x_1 + \left(-2 + \frac{1}{6}\right)s_1 = \left(2 + \frac{7}{8}\right)$

Koefisien antara suku harus dibuat positif (+) dan koefisien diubah bentuknya sedemikian sehingga sisa pecahannya positif dan kurang dari satu.

Setelah itu, kita bisa buat kendala *Gomory*-nya:

$$S_{gn} - \frac{1}{6}s_1 = -\frac{7}{8}$$

Di mana: $S_{gn} = skala\ Gomory\ ke - n \quad , \quad n = 1,2, \ldots$

Perhatikan bahwa semua koefisien (sisa pecahan) yang diambil sebelumnya bernilai positif, namun di kendala *Gomory* menjadi negatif (kecuali skala *Gomory* S_{gn}). Dalam suatu perhitungan, skala Gomory yang terbuat bisa banyak (tergantung dari kasus).

Setelah membuat kendala *Gomory*, kita bisa memasukkannya ke tabel:

Variabel Basis	x_1	x_2	s_1	S_1	A_1	S_{gn}	Nilai Kanan (NK)
Z	*	*	*	*	*	0	*
x_1	1	0	-11/6	0	0	0	23/8 = 2 + 7/8
x_2	*	*	*	*	*	0	11/2 = 5 + 1/2
S_{gn}	0	0	-1/6	0	0	1	-7/8

Selanjutnya, kita perlu melakukan metode *Dual Simplex*. Metode ini sangat mirip dengan metode *Simplex Big-M* yang sudah dibahas sebelumnya. Perbedaan yang ada hanyalah penentuan baris dan kolom kunci, serta tidak perlu menambahkan koefisien M. Kita harus menentukan baris kunci terlebih dahulu disini. Untuk kasus minimasi, ini dilakukan dengan melihat baris dengan nilai kanan (NK) negatif dan angka terbesar (singkatnya bilangan terkecil). Baris yang memiliki karakteristik seperti itu merupakan baris kunci (baris Z tidak bisa menjadi baris kunci). Untuk menentukan kolom kunci, kita harus menentukan nilai rasio terlebih dahulu. Nilai rasio bisa didapat dengan membagi nilai baris Z dengan nilai baris kunci per kolomnya.

Variabel Basis	x_1	x_2	s_1	S_1	A_1	S_{gn}	Nilai Kanan (NK)
Z	q	w	e	r	t	y	u
x_1	1	0	-11/6	0	0	0	23/8 = 2 + 7/8
x_2	*	*	*	*	*	0	11/2 = 5 + 1/2
S_{gn}	0	0	-1/6	0	0	1	-7/8
Rasio	-	-	-6e	-	-	y	-

Untuk kolom kunci, tentukan nilai rasio yang mana yang bila dimutlakkan (dijadikan positif), nilainya menjadi yang terkecil (namun harus > 0) dibanding lainnya. Kolom (kecuali NK) dengan nilai rasio tersebut menjadi kolom kunci. Jika ada dua (atau lebih) kolom dengan nilai rasio yang sama-sama terkecil (setelah dimutlakkan), bebas pilih kolom yang mana (di antara dua/lebih kolom tersebut) yang ingin dijadikan kolom kunci (misal disini $|-6e| < |y|$, maka s_1 menjadi kolom kunci).

Variabel Basis	x_1	x_2	s_1	S_1	A_1	S_{gn}	Nilai Kanan (NK)
Z	q	w	e	r	t	y	u
x_1	1	0	-11/6	0	0	0	23/8 = 2 + 7/8
x_2	*	*	*	*	*	0	11/2 = 5 + 1/2
S_{gn}	0	0	-1/6	0	0	1	-7/8
Rasio	-	-	-6e	-	-	y	-

Untuk selanjutnya sampai selesai memperbarui nilai pada semua baris, tahap-tahap *Dual Simplex* sama dengan tahap-tahap *Simplex Big-M*. Pada bagian akhir untuk kasus minimasi, kita harus cek apakah semua nilai di baris Z (kecuali NK) bernilai ≤ 0 dan semua nilai di kolom NK ≥ 0. Jika belum, kita perlu melakukan iterasi selanjutnya untuk *Dual Simplex*. Jika nilai kanan (NK) dari x_1 belum *integer*, buat skala *Gomory* lagi dan lakukan *Dual Simplex* lagi. Jika sudah NK dari x_1 sudah *integer*, namun belum memenuhi syarat kendala dan/atau tidak memenuhi syarat pengujian akhir metode *Dual Simplex*, perlu dilakukan iterasi selanjutnya untuk metode *Cutting Plane* ini. Iterasi berikutnya dilanjutkan dengan melakukan *Dual Simplex* terhadap hasil dari iterasi sebelumnya terlebih dahulu baru melakukan siklus pembuatan skala *Gomory* – lakukan *Dual Simplex*. Jika variabel penentu sudah berbentuk

integer dan sudah memenuhi semua syarat yang ada (baik dari sisi kendala kasus maupun dari sisi pengujian akhir metode *Dual Simplex*), metode *Cutting Plane* akhirnya selesai.

Pada kasus diatas, kita menganggap bahwa hanya satu variabel penentu saja yang harus *integer*. Jika ada lebih dari satu, kita harus menentukan variabel penentu mana yang mau diproses dan dijadikan dasar untuk pembuatan skala *Gomory*-nya. Ini dilakukan dengan cara membanding-bandingkan sisa pecahan mana yang lebih besar yang telah disebut sebelumnya. Jadi, kita tidak membandingkannya di awal saja, namun setiap selesai *Dual Simplex*, jika variabel penentu yang harus diubah ke *integer* cukup banyak, kita harus menentukan yang mana yang akan diproses terlebih dahulu. Kita tidak menunggu satu variabel penentu (yang misalnya terpilih untuk menjadi variabel yang diproses terlebih dahulu pada pembuatan skala *Gomory* paling pertama) untuk menjadi *integer* terlebih dahulu baru memproses variabel penentu yang lain. Jika suatu variabel penentu yang harus *integer* sudah menjadi *integer*, kita tidak perlu memprosesnya lagi. Namun ada kemungkinan kecil, jika NK dari suatu variabel penentu yang sudah berbentuk *integer* menjadi desimal lagi karena perhitungan-perhitungan yang dilakukan saat pemrosesan variabel penentu lainnya, variabel penentu tersebut akan masuk lagi ke daftar pemrosesan di siklus selanjutnya. Sebaliknya juga berlaku (sama-sama kemungkinannya kecil) jika suatu variabel penentu yang NK-nya (awalnya desimal) harus berbentuk *integer* menjadi *integer* karena pemrosesan variabel penentu lain (bukan karena pemrosesan variabel penentu tersebut sendiri), kita tak perlu memasukkan variabel penentu tersebut ke pemrosesan di siklus selanjutnya. Metode *Cutting Plane* akan selesai ketika semua variabel penentu yang harus berbentuk *integer* sudah menjadi *integer* dan hasil sudah memenuhi semua syarat yang ada (syarat-syarat dan pengujian yang sudah disebut sebelumnya).

BAB III
METODE & DATA

3.1 METODE PENELITIAN

Metode penelitian yang dipakai disini adalah metode penelitian kuantitatif, di mana data yang diambil dari *paper* referensi bersifat numerik (non-kategorik). Fokus utama dari penelitian ini adalah menentukan alokasi dana minimum untuk diet pemain sepak bola menggunakan metode *Integer Linear Programming* (ILP) dengan pendekatan *Gomory* dan bantuan metode *Simplex Big-M* dan *Dual Simplex*. Langkah-langkah yang dilakukan untuk melaksanakan penelitian meliputi:

1. Mencari Sumber Data

Data yang akan digunakan untuk membuat model matematika bergantung pada data asli yang berasal dari *paper* referensi mengenai rekomendasi asupan nutrisi untuk pemain sepakbola pria tingkat profesional. Untuk harga per unit vitamin atau suplemen-suplemen lainnya akan menggunakan harga asumsi (semu) karena harga asli dari bahan-bahan tersebut bisa berubah secara fluktuatif.

2. Pemilihan Bahan Nutrisi

Setelah mendapatkan data, kita harus menentukan kebutuhan nutrisi yang diperlukan oleh para pemain sepakbola untuk bahan-bahan tertentu, seperti zat besi, kalsium, vitamin D, dan seterusnya. Ini dilakukan karena kita hanya akan melakukan pemodelan untuk beberapa bahan saja di penelitian ini. Lebih tepatnya, kita hanya akan menggunakan lima bahan saja disini.

3. Formulasi Model Matematika

Membuat model matematika yang mencakup fungsi tujuan untuk meminimalkan alokasi dana dan kendala-kendala yang harus dipenuhi terkait kebutuhan nutrisi. Beberapa variabel penentu juga ditetapkan harus berbentuk *integer* karena karakteristik dari bahan tersebut (padat harus *integer*, cairan tidak harus *integer*).

4. Penerapan Metode *Simplex Big-M*

Menggunakan metode *Simplex Big-M* untuk menentukan solusi optimal awal/sejati dari model matematika.

5. Penerapan Pendekatan *Gomory* dan Metode *Dual Simplex*

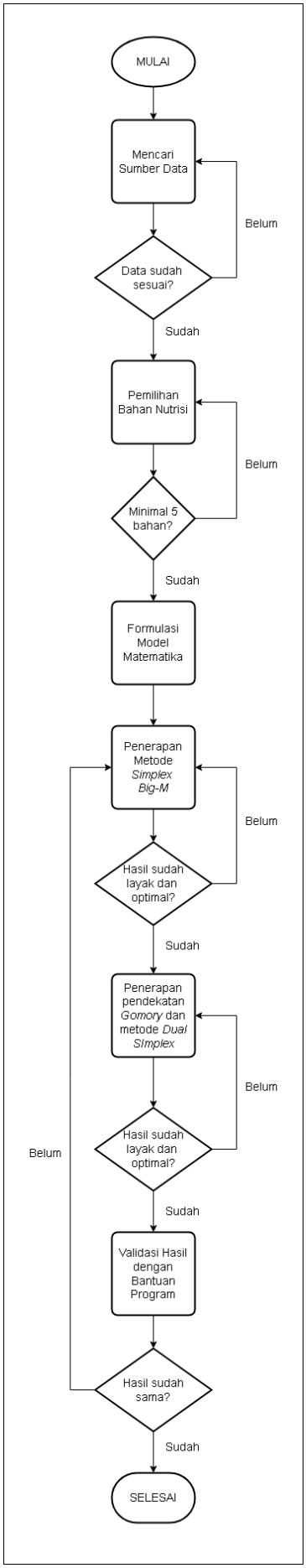
Menggunakan pendekatan *Gomory* dan metode *Dual Simplex* untuk menentukan solusi yang optimal dengan syarat bahwa variabel-variabel penentu tertentu memiliki nilai *integer* pada akhirnya (sesuai dengan yang telah ditetapkan di model matematika).

6. Validasi Hasil dengan Bantuan Program

Memvalidasi hasil optimal dari sebelumnya dengan membandingkannya terhadap hasil yang diberikan jika perhitungan menggunakan suatu program atau *software* tertentu.

3.2 FLOWCHART

Berikut merupakan tahapan-tahapan penelitian dalam bentuk diagram alur (*flowchart*):


MULAI
Mencari Sumber Data
Belum
Data sudah sesuai?
Sudah
Pemilihan Bahan Nutrisi
Belum
Minimal 5 bahan?
Sudah
Formulasi Model Matematika
Penerapan Metode Simplex Big-M
Belum
Hasil sudah layak dan optimal?
Sudah
Penerapan pendekatan Gomory dan metode Dual Simplex
Belum
Hasil sudah layak dan optimal?
Belum
Sudah
Validasi Hasil dengan Bantuan Program
Hasil sudah sama?
Sudah
SELESAI

Link Gambar Flowchart: Flowchart Penelitian AMM - Google Drive

3.3 DATA

Data yang digunakan dalam penelitian ini berasal dari *paper* referensi (artikel dari jurnal *International Journal of Environmental Research and Public Health*) tahun 2020 berjudul "*Assessment of the Dietary Intake of High-Rank Professional Male Football Players during a Preseason Training Week*" oleh para penulis Książek, Zagrodna, dan Słowińska-Lisowska. Data yang akan dipakai terdapat di halaman 6 (dari 11) pada *paper* tersebut. Berikut adalah tabel dari data yang dipakai (foto diambil langsung dari *paper*):

Table 5. Mineral and vitamin intake by football players.

Vitamin/Mineral	Recommendation (12)	Diet Excluding Supplements (Median/Mean ± SD)	Diet Including Supplements (Median/Mean ± SD)	*p*
Vitamin B_1 (mg/d)	1.5–3.0	1.8	2.3	0.0006
Vitamin B_2 (mg/g)	1.1/1000 kcal	2.2 ± 0.4	3.1 ± 1.1	0.0021
Niacin (mg/d)	14–20	28.9 ± 6.0	34.9 ± 8.6	0.0003
Vitamin B_6 (mg/d)	1.6–2.0	2.7	3.6	0.0001
Vitamin B_{12} (µg/d)	2.4–2.5	5.9	6.6	0.0001
Folate (µg/d)	400	354.4 ± 86.8	433.9 ± 121.4	0.0000
Vitamin A (µg/d)	700–900	913.8 ± 254.4	1609.6 ± 691.3	0.0000
Vitamin D (µg/d)	20–50 [17]	4.9	56.5	0.0000
Vitamin C (mg/d)	200	191.9 ± 71.1	255.7 ± 120.2	0.0005
Vitamin E (mg/d)	15	11.1 ± 4.6	16.0 ± 6.7	0.0000
Calcium (mg/d)	1300–1500	1179.9 ± 265.8	1291.5 ± 318.2	0.0000
Phosphorus (mg/d)	1250–1500	1881.6 ± 355.0	1969.8 ± 402.4	0.0000
Magnesium (mg/d)	400–450	469.5 ± 114.3	519.3 ± 136.3	0.0000
Potassium (mg/d)	2000	4510.8 ± 780.0	5019.8 ± 1035.1	0.0000
Sodium (mg/d)	1500	3889.6 ± 606.3	4362.6 ± 739.4	0.0000
Iron (mg/d)	15–18	14.9 ± 2.5	16.7 ± 3.3	0.0000
Zinc (mg/d)	11–15	13.3 ± 2.4	14.9 ± 3.4	0.0007
Iodine (µg/d)	150	158.2 ± 32.6	177.0 ± 42.2	0.0000

Dua kolom yang benar-benar dipakai disini sebenarnya hanya kolom nama bahan nutrisi (‘Vitamin/Mineral’) dan kolom rekomendasi asupannya per hari (‘Recommendation’). Untuk kasus ini, kita hanya memakai lima bahan berikut:

- Vitamin B1 (cairan, tidak harus *integer*)
- Vitamin A (cairan, tidak harus *integer*)
- Vitamin D (cairan, tidak harus *integer*)
- Iron (padat, harus *integer*)
- Zinc (padat, harus *integer*)

Data dari bahan-bahan ini digunakan untuk membangun model matematika. Lebih tepatnya, pada bagian kendala/syarat (*constraint*).

3.4 SKENARIO SIMULASI

Untuk skenario simulasi, ada beberapa hal yang harus dipahami sebelum melihat model matematika , yaitu:

- Harga dari setiap bahan yang dipakai disini merupakan harga semu (buatan).

- Asumsi klub sepakbola membeli semua kebutuhan bahan-bahan ini dari satu *Supplier*, dengan harga yang diberikan merupakan harga yang ditawarkan dari *Supplier* tersebut (bukan harga asli secara faktanya) sehingga tidak bisa dinegosiasi (anggap saja sudah terikat pada suatu kontrak).

- Asumsi hasil model matematika ini untuk satu pemain per hari sehingga klub sepakbola bisa dengan mudah mengalikan hasil alokasi dana optimal dan/atau banyak bahan-bahan nutrisi yang diperlukan dengan banyak pemain yang dimilikinya dan jumlah hari yang diperlukan (pramusim biasanya berlangsung selama satu sampai dua minggu).

- Asumsi kendala-kendala di model matematika (yang berbentuk pertidaksamaan multivariabel) merupakan suatu kewajiban yang harus diterima oleh klub karena telah berkontrak bersama *Supplier* untuk pembelian bahan-bahan untuk satu pemain per hari.

Berikut adalah daftar harga (buatan) bahan-bahan yang dipilih:

Nama Bahan	Harga (dalam satuan ribuan rupiah) Per Unit Bahan
Vitamin B1	0.5/mg
Vitamin A	0.3/µg
Vitamin D	0.2/µg
Iron	0.9/kapsul
Zinc	0.75/kapsul

Berikut adalah model matematika untuk simulasi ini:

Variabel Penentu:

x_1 = **Vitamin B1 per hari** (mg)
x_2 = **Vitamin A per hari** (µg)
x_3 = **Vitamin D per hari** (µg)
x_4 = **Iron per hari** (kapsul, 1 kapsul = 1.5 mg)
x_5 = **Zinc per hari** (kapsul, 1 kapsul = 1 mg)

Fungsi Tujuan:

$$Minimum\ Z = 0.5x_1 + 0.3x_2 + 0.2x_3 + 0.9x_4 + 0.75x_5$$

Syarat/Kendala:

$$x_1 + \frac{x_2}{1000} + \frac{x_3}{1000} \geq 2.8$$
$$x_4 + x_5 \geq 23$$
$$x_2 + x_3 \leq 1000$$

$x_1 + 1.5x_4 + x_5 \leq 32$
$x_1 + \frac{x_2}{1000} + \frac{x_3}{1000} + 1.5x_4 + x_5 \leq 34$

Vitamin B1: $1.5 \leq x_1 \leq 3$
Vitamin A: $700 \leq x_2 \leq 900$
Vitamin D: $20 \leq x_3 \leq 50$
Iron: $10 \leq x_4 \leq 12$
Zinc: $11 \leq x_5 \leq 15$
$x_4, x_5\ integer$

Penjelasan syarat/kendala:

$x_1 + \frac{x_2}{1000} + \frac{x_3}{1000} \geq 2.8$

Jumlah massa bahan-bahan vitamin (cairan) harus lebih besar dari atau sama dengan 2.8 miligram (karena Vitamin A $[x_2]$ dan D $[x_3]$ dalam unit mikrogram $[\mu g]$, nilai kedua variabel tersebut harus dibagi 1000 supaya bisa setara dengan unit miligram).

$x_4 + x_5 \geq 23$

Jumlah kapsul bahan-bahan mineral (padat) harus lebih banyak dari atau sama dengan 23 kapsul.

$x_2 + x_3 \leq 1000$

Jumlah massa Vitamin A dan D harus lebih kecil dari atau sama dengan 1000 mikrogram.

$x_1 + 1.5x_4 + x_5 \leq 32$

Jumlah Vitamin B1, Iron, dan Zinc harus lebih kecil dari atau sama dengan 32 miligram (x_4 dikali dengan 1.5 karena 1 kapsul Iron bermassa 1.5 miligram dan x_5 dikali dengan 1 karena 1 kapsul Zinc bermassa 1 miligram).

$x_1 + \frac{x_2}{1000} + \frac{x_3}{1000} + 1.5x_4 + x_5 \leq 34$

Jumlah massa untuk kelima bahan harus lebih kecil dari atau sama dengan 34 miligram.

Vitamin B1: $1.5 \leq x_1 \leq 3$
Vitamin A: $700 \leq x_2 \leq 900$
Vitamin D: $20 \leq x_3 \leq 50$
Iron: $10 \leq x_4 \leq 12$
Zinc: $11 \leq x_5 \leq 15$

$x_4, x_5\ integer$

Karena x_4 (Iron) dan x_5 (Zinc) memiliki unit kapsul, maka mereka harus *integer*. Dan juga karena disini didefinisikan bahwa satu kapsul memiliki massa sebesar 1.5 mg dan 1 mg untuk masing-masing kapsul Iron dan Zinc, *constraint* untuk variabel x_4 dan x_5 juga harus disesuaikan. Di tabel sebelumnya, disebut bahwa rekomendasi asupan Iron adalah 15-18 mg/d (mg/hari), sehingga *constraint* untuk x_4 harus disesuaikan menjadi $\frac{15}{1.5} \leq x_4 \leq \frac{18}{1.5} \rightarrow 10 \leq x_4 \leq 12$. Untuk x_5, dari 11-15 mg/d tetap $11 \leq x_5 \leq 15$. *Constraint* variabel-variabel lainnya mengikuti secara langsung dari tabel data.

BAB IV
HASIL ANALISIS

4.1 TAHAPAN SOLUSI

Berikut adalah langkah-langkah solusi dari model matematika yang telah dibuat sebelumnya:

Variabel Penentu:

$x_1 =$ **Vitamin B1 per hari** (mg)
$x_2 =$ **Vitamin A per hari** (µg)
$x_3 =$ **Vitamin D per hari** (µg)
$x_4 =$ **Iron per hari** (kapsul, 1 kapsul = 1.5 mg)
$x_5 =$ **Zinc per hari** (kapsul, 1 kapsul = 1 mg)

Fungsi Tujuan (Meminimumkan Biaya Pembelian Bahan-Bahan):

$$Minimum\ Z = 0.5x_1 + 0.3x_2 + 0.2x_3 + 0.9x_4 + 0.75x_5$$

Syarat/Kendala:

$$x_1 + \frac{x_2}{1000} + \frac{x_3}{1000} \geq 2.8$$
$$x_4 + x_5 \geq 23$$
$$x_2 + x_3 \leq 1000$$
$$x_1 + 1.5x_4 + x_5 \leq 32$$
$$x_1 + \frac{x_2}{1000} + \frac{x_3}{1000} + 1.5x_4 + x_5 \leq 34$$

Vitamin B1: $1.5 \leq x_1 \leq 3$
Vitamin A: $700 \leq x_2 \leq 900$
Vitamin D: $20 \leq x_3 \leq 50$
Iron: $10 \leq x_4 \leq 12$
Zinc: $11 \leq x_5 \leq 15$
$x_4, x_5\ integer$

Untuk memudahkan perhitungan, *constraint* akan kita ubah menjadi:

Vitamin B1: $0.5 \leq x_1' \leq 2$, di mana $x_1' = x_1 - 1$
Vitamin A: $0 \leq x_2' \leq 200$, di mana $x_2' = x_2 - 700$
Vitamin D: $0 \leq x_3' \leq 30$, di mana $x_3' = x_3 - 20$
Iron: $0 \leq x_4' \leq 2$, di mana $x_4' = x_4 - 10$
Zinc: $0 \leq x_5' \leq 4$, di mana $x_5' = x_5 - 11$

$x_4', x_5'\ integer$

Sehingga fungsi tujuan dan syarat lain menjadi:

Fungsi Tujuan:
$Min\ Z = 0.5x_1' + 0.3x_2' + 0.2x_3' + 0.9x_4' + 0.75x_5' + 231.75$

Syarat:

$$x_1' + \frac{x_2'}{1000} + \frac{x_3'}{1000} \geq 1.08 \rightarrow 1000x_1' + x_2' + x_3' \geq 1080$$

$$x_4' + x_5' \geq 2$$

$$x_2' + x_3' \leq 280$$

$$x_1' + 1.5x_4' + x_5' \leq 5$$

$$x_1' + \frac{x_2'}{1000} + \frac{x_3'}{1000} + 1.5x_4' + x_5' \leq 6.28 \rightarrow 1000x_1' + x_2' + x_3' + 1500x_4' + 1000x_5' \leq 6280$$

Selanjutnya, kita akan mengubah fungsi tujuan dan syarat menjadi bentuk simpleks:

$$Z - 0.5x_1' - 0.3x_2' - 0.2x_3' - 0.9x_4' - 0.75x_5' - 0(S_1 + S_2 + s_3 + s_4 + s_5) - M(A_1 + A_2) = 231.75$$

$$1000x_1' + x_2' + x_3' - S_1 + A_1 = 1080$$

$$x_4' + x_5' - S_2 + A_2 = 2$$

$$x_2' + x_3' + s_3 = 280$$

$$x_1' + 1.5x_4' + x_5' + s_4 = 5$$

$$1000x_1' + x_2' + x_3' + 1500x_4' + 1000x_5' + s_5 = 6280$$

$$0.5 \leq x_1' \leq 2$$

$0 \leq x_2' \leq 200$

$0 \leq x_3' \leq 30$

$0 \leq x_4' \leq 2$

$0 \leq x_5' \leq 4$

$S_1, A_1, S_2, A_2, s_3, s_4, s_5 \geq 0$

$x_4', x_5'\ integer$

Selanjutnya, kita akan membentuk tabel simpleks:

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	NK
Z	-0.5	-0.3	-0.2	-0.9	-0.75	0	-M	0	-M	0	0	0	231.75
A_1	1000	1	1	0	0	-1	1	0	0	0	0	0	1080
A_2	0	0	0	1	1	0	0	-1	1	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	280
s_4	1	0	0	1.5	1	0	0	0	0	0	1	0	5
s_5	1000	1	1	1500	1000	0	0	0	0	0	0	1	6280

Setelah itu, kita akan memperbarui nilai baris Z supaya koefisien M menghilang dari kolom A_1 dan A_2 pada baris tersebut.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	NK
Z	1000M-0.5	M-0.3	M-0.2	M-0.9	M-0.75	-M	0	-M	0	0	0	0	1082M+231.75
A_1	1000	1	1	0	0	-1	1	0	0	0	0	0	1080
A_2	0	0	0	1	1	0	0	-1	1	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	280
s_4	1	0	0	1.5	1	0	0	0	0	0	1	0	5
s_5	1000	1	1	1500	1000	0	0	0	0	0	0	1	6280

Operasi: $baris\ Z_{baru} = baris\ Z_{lama} + (M \cdot baris\ A_1) + (M \cdot baris\ A_2)$

Iterasi 1:

Pertama, kita tentukan kolom kunci. Terlihat bahwa kolom x_1' yang memiliki nilai koefisien M terbesar di baris Z sehingga kolom tersebut menjadi kolom kunci.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	NK
Z	1000M-0.5	M-0.3	M-0.2	M-0.9	M-0.75	-M	0	-M	0	0	0	0	1082M+231.75
A_1	1000	1	1	0	0	-1	1	0	0	0	0	0	1080
A_2	0	0	0	1	1	0	0	-1	1	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	280
s_4	1	0	0	1.5	1	0	0	0	0	0	1	0	5
s_5	1000	1	1	1500	1000	0	0	0	0	0	0	1	6280

Kedua, kita harus tentukan baris kunci. Untuk itu, kita perlu bantuan dari kolom indeks, di mana hasil indeks (positif) terkecil menjadikan baris tersebut baris kunci.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	NK	Indeks
Z	1000M-0.5	M-0.3	M-0.2	M-0.9	M-0.75	-M	0	-M	0	0	0	0	1082M+231.75	-
A_1	1000	1	1	0	0	-1	1	0	0	0	0	0	1080	1.08
A_2	0	0	0	1	1	0	0	-1	1	0	0	0	2	-
s_3	0	1	1	0	0	0	0	0	0	1	0	0	280	-
s_4	1	0	0	1.5	1	0	0	0	0	0	1	0	5	5
s_5	1000	1	1	1500	1000	0	0	0	0	0	0	1	6280	6.28

Bisa dilihat bahwa baris A_1 memiliki nilai indeks terkecil sehingga menjadi baris kunci.

Ketiga, kita perlu memperbarui baris kunci dengan membagi semua nilainya dengan angka kunci. A_1 menjadi *leaving variable* dan x_1' menjadi *entering variable*.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	NK
Z	1000 M-0.5	M-0.3	M-0.2	M-0.9	M-0.75	-M	0	-M	0	0	0	0	1082M+231.75
x_1'	1	0.001	0.001	0	0	-0.001	0.001	0	0	0	0	0	1.08
A_2	0	0	0	1	1	0	0	-1	1	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	280
s_4	1	0	0	1.5	1	0	0	0	0	0	1	0	5
s_5	1000	1	1	1500	1000	0	0	0	0	0	0	1	6280

Keempat, kita perlu memperbarui nilai baris-baris lainnya.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	NK
Z	0	-0.2995	-0.1995	M-0.9	M-0.75	-0.0005	-M+0.0005	-M	0	0	0	0	2M+232.29
x_1'	1	0.001	0.001	0	0	-0.001	0.001	0	0	0	0	0	1.08
A_2	0	0	0	1	1	0	0	-1	1	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	280
s_4	0	-0.001	-0.001	1.5	1	0.001	-0.001	0	0	0	1	0	3.92
s_5	0	0	0	1500	1000	1	-1	0	0	0	0	1	5200

Ingat Rumus:

$$Nilai\ Baru = Nilai\ lama - (NK_{lama} \times NB_{baru})$$

Di mana:

$$NK_{lama} = nilai\ kolom\ kunci\ (sebelum\ diperbarui)\ pada\ baris\ nilai\ lama\ berada$$

$$NB_{baru} = nilai\ baris\ kunci\ (setelah\ diperbarui)\ pada\ kolom\ nilai\ lama\ berada$$

Iterasi 2:

Karena masih ada nilai (koefisien M disini) yang lebih besar dari 0 di baris Z, iterasi lagi.

Pertama, kita tentukan kolom kunci. Terlihat bahwa kolom x_5' yang memiliki nilai koefisien M terbesar di baris Z sehingga kolom tersebut menjadi kolom kunci (x_4' juga sebenarnya memiliki koefisien M yang sama besar, namun konstanta setelahnya lebih kecil, $-0.9 < -0.75$).

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	NK
Z	0	-0.2995	-0.1995	M-0.9	M-0.75	-0.0005	-M+0.0005	-M	0	0	0	0	2M+232.29
x_1'	1	0.001	0.001	0	0	-0.001	0.001	0	0	0	0	0	1.08
A_2	0	0	0	1	1	0	0	-1	1	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	280
s_4	0	-0.001	-0.001	1.5	1	0.001	-0.001	0	0	0	1	0	3.92

s_5	0	0	0	1500	1000	1	-1	0	0	0	0	1	5200

Kedua, kita harus tentukan baris kunci. Untuk itu, kita perlu bantuan dari kolom indeks, di mana hasil indeks (positif) terkecil menjadikan baris tersebut baris kunci.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	NK	Indeks
Z	0	-0.2995	-0.1995	M-0.9	M-0.75	-0.00005	-M+0.00005	-M	0	0	0	0	2M+232.29	-
x_1'	1	0.001	0.001	0	0	-0.001	0.001	0	0	0	0	0	1.08	-
A_2	0	0	0	1	1	0	0	-1	1	0	0	0	2	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	280	-
s_4	0	-0.001	-0.001	1.5	1	0.001	-0.001	0	0	0	1	0	3.92	3.92
s_5	0	0	0	1500	1000	1	-1	0	0	0	0	1	5200	5.2

Bisa dilihat bahwa baris A_2 memiliki nilai indeks terkecil sehingga menjadi baris kunci.

Ketiga, kita perlu memperbarui baris kunci dengan membagi semua nilainya dengan angka kunci. A_2 menjadi *leaving variable* dan x_5' menjadi *entering variable*.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	NK
Z	0	-0.2995	-0.1995	M-0.9	M-0.75	-0.0005	-M+0.0005	-M	0	0	0	0	2M+232.29
x_1'	1	0.001	0.001	0	0	-0.001	0.001	0	0	0	0	0	1.08
x_5'	0	0	0	1	1	0	0	-1	1	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	280
s_4	0	-0.001	-0.001	1.5	1	0.001	-0.001	0	0	0	1	0	3.92
s_5	0	0	0	1500	1000	1	-1	0	0	0	0	1	5200

Ternyata angka kunci sudah satu sehingga tidak perlu diperbarui.

Keempat, kita perlu memperbarui nilai baris-baris lainnya.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	NK

Z	0	-0.2995	-0.1995	-0.15	0	-0.0005	-M+0.0005	-0.75	-M+0.75	0	0	0	233.79
x_1'	1	0.001	0.001	0	0	-0.001	0.001	0	0	0	0	0	1.08
x_5'	0	0	0	1	1	0	0	-1	1	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	280
s_4	0	-0.001	-0.001	0.5	0	0.001	-0.001	1	-1	0	1	0	1.92
s_5	0	0	0	500	0	1	-1	1000	-1000	0	0	1	3200

Karena di baris Z nilainya sudah ≤ 0 (termasuk koefisien M), metode *Simplex Big-M* sudah selesai dengan memberikan hasil:

$$x_1' = 1.08, x_2' = x_3' = x_4' = 0, x_5' = 2$$

$$x_1 = 2.08, x_2 = 700, x_3 = 20, x_4 = 10, x_5 = 13, dan\ Z = 233.79$$

Ingat:

$Minimum\ Z = 0.5x_1 + 0.3x_2 + 0.2x_3 + 0.9x_4 + 0.75x_5$

Vitamin B1: $0.5 \leq x_1' \leq 2$, di mana $x_1' = x_1 - 1$
Vitamin A: $0 \leq x_2' \leq 200$, di mana $x_2' = x_2 - 700$
Vitamin D: $0 \leq x_3' \leq 30$, di mana $x_3' = x_3 - 20$
Iron: $0 \leq x_4' \leq 2$, di mana $x_4' = x_4 - 10$
Zinc: $0 \leq x_5' \leq 4$, di mana $x_5' = x_5 - 11$
$x_4', x_5'\ integer$

Vitamin B1: $1.5 \leq x_1 \leq 3$
Vitamin A: $700 \leq x_2 \leq 900$
Vitamin D: $20 \leq x_3 \leq 50$
Iron: $10 \leq x_4 \leq 12$
Zinc: $11 \leq x_5 \leq 15$
$x_4, x_5\ integer$

Dari metode *Simplex Big-M*, diberikan hasil optimal yang di mana dua variabel penentu kita yang seharusnya *integer* sudah menjadi *integer* (x_4 dan x_5) sehingga perhitungan sebenarnya sudah selesai. **Alokasi dana minimum untuk satu pemain per harinya sebesar Rp. 233.790,- dengan detail: Vitamin B1 2.08 mg, Vitamin A 700 μg, Vitamin D 20 μg, Iron 10 kapsul (1.5 mg), dan Zinc 13 kapsul (1 mg)**. Alokasi dana tersebut bisa disesuaikan dengan banyaknya pemain dan banyaknya hari dengan cara dikalikan nantinya.

Namun, jika semisalnya x_1 yang berbentuk desimal di atas tiba-tiba harus berubah menjadi *integer* (misal ada kasus bahwa *Supplier* hanya bisa menjual Vitamin B1 kapsul/padat di mana 1 kapsul = 1 mg). Jika seperti itu, kita akan lanjut ke tahap metode *Cutting Plane*.

Namun, ini merupakan skenario mendadak sehingga kita hanya akan melihat bagaimana proses umum metode tersebut saja. Perhitungan yang ditunjukkan tidak akan lengkap dan berperan sebagai fondasi awal jawaban atas skenario mendadak sebelumnya.

Iterasi 1:

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	NK
Z	0	-0.2995	-0.1995	-0.15	0	-0.0005	-M+0.0005	-0.75	-M+0.75	0	0	0	233.79
x_1'	1	0.001	0.001	0	0	-0.001	0.001	0	0	0	0	0	1.08
x_5'	0	0	0	1	1	0	0	-1	1	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	280
s_4	0	-0.001	-0.001	0.5	0	0.001	-0.001	1	-1	0	1	0	1.92
s_5	0	0	0	500	0	1	-1	1000	-1000	0	0	1	3200

Kita akan membuat kolom baru bernama ‘pemecahan’ untuk melihat sisa pecahan dari setiap variabel penentu yang seharusnya *integer*, namun belum *integer* dari perhitungan sebelumnya.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	NK	Pemecahan
Z	0	-0.2995	-0.1995	-0.15	0	-0.0005	-M+0.0005	-0.75	-M+0.75	0	0	0	233.79	-
x_1'	1	0.001	0.001	0	0	-0.001	0.001	0	0	0	0	0	1.08	$1+\frac{2}{25}$
x_5'	0	0	0	1	1	0	0	-1	1	0	0	0	2	-
s_3	0	1	1	0	0	0	0	0	0	1	0	0	280	-
s_4	0	-0.001	-0.001	0.5	0	0.001	-0.001	1	-1	0	1	0	1.92	-
s_5	0	0	0	500	0	1	-1	1000	-1000	0	0	1	3200	-

Selanjutnya, kita akan membuat skala *Gomory* berdasarkan baris x_1'.

$$x_1' + 0.001x_2' + 0.001x_3' - 0.001S_1 + 0.001A_1 = 0.58$$

$$x_1' + \left(0 + \frac{1}{1000}\right)x_2' + \left(0 + \frac{1}{1000}\right)x_3' + \left(-1 + \frac{999}{1000}\right)S_1 + \left(0 + \frac{1}{1000}\right)A_1 = \left(1 + \frac{2}{25}\right)$$

Skala (kendala) *Gomory*:

$$S_{g1} - \frac{1}{1000}x_2' - \frac{1}{1000}x_3' - \frac{999}{1000}S_1 - \frac{1}{1000}A_1 = -\frac{2}{25}$$

Setelah itu, kita masukkan kendala ini ke tabel optimal:

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	S_{g1}	NK
Z	0	-0.2995	-0.1995	-0.15	0	-0.00005	-M+0.00005	-0.75	-M+0.75	0	0	0	0	233.79
x_1'	1	0.001	0.001	0	0	-0.001	0.001	0	0	0	0	0	0	0.58
x_5'	0	0	0	1	1	0	0	-1	1	0	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	0	280
s_4	0	-0.001	-0.001	0.5	0	0.001	-0.001	1	-1	0	1	0	0	1.92
s_5	0	0	0	500	0	1	-1	1000	-1000	0	0	1	0	3200
S_{g1}	0	-0.001	-0.001	0	0	-0.999	-0.001	0	0	0	0	0	1	-0.08

Mulai dari sini, kita akan menerapkan metode *Dual Simplex* untuk mencari nilai optimal yang *integer* untuk x_1. Karena $x_1' = x_1 - 1.5$, kita ingin menerapkan metode ini supaya hasil x_1' memiliki desimal .5 supaya x_1 memiliki nilai bulat.

Pertama, kita harus tentukan baris kunci. Terlihat bahwa baris S_{g1} memiliki NK bertanda negatif (dan satu-satunya nilai dengan tanda tersebut) sehingga baris tersebut menjadi baris kunci.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	S_{g1}	NK
Z	0	-0.2995	-0.1995	-0.15	0	-0.00005	-M+0.00005	-0.75	-M+0.75	0	0	0	0	233.79
x_1'	1	0.001	0.001	0	0	-0.001	0.001	0	0	0	0	0	0	0.58

x_5'	0	0	0	1	1	0	0	-1	1	0	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	0	280
s_4	0	-0.001	-0.001	0.5	0	0.001	-0.001	1	-1	0	1	0	0	1.92
s_5	0	0	0	500	0	1	-1	1000	-1000	0	0	1	0	3200
S_{g1}	0	-0.001	-0.001	0	0	-0.999	-0.001	0	0	0	0	0	1	-0.08

Kedua, kita harus tentukan kolom kunci. Disini akan dibuat baris baru untuk menampung hasil pembagian antara nilai-nilai baris Z dengan baris kunci yang dinamakan baris Rasio. Kolom yang memiliki nilai rasio terkecil (setelah dimutlakkan) akan menjadi kolom kunci.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	S_{g1}	NK
Z	0	-0.2995	-0.1995	-0.15	0	-0.00005	-M+0.00005	-0.75	-M+0.75	0	0	0	0	233.79
x_1'	1	0.001	0.001	0	0	-0.001	0.001	0	0	0	0	0	0	0.58
x_5'	0	0	0	1	1	0	0	-1	1	0	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	0	280
s_4	0	-0.001	-0.001	0.5	0	0.001	-0.001	1	-1	0	1	0	0	1.92
s_5	0	0	0	500	0	1	-1	1000	-1000	0	0	1	0	3200
S_{g1}	0	-0.001	-0.001	0	0	-0.999	-0.001	0	0	0	0	0	1	-0.08
Rasio	-	299.5	199.5	-	-	$\frac{1}{1998}$	1000M-0.5	-	-	-	-	-	0	-

Terlihat disini bahwa kolom S_1 memiliki nilai rasio terkecil (nilai rasio yang dipilih harus > 0 setelah dimutlakkan) sehingga kolom tersebut menjadi kolom kunci.

Ketiga, kita harus memperbarui nilai baris kunci sehingga angka kuncinya menjadi satu. S_1 menjadi *entering variable* dan S_{g1} menjadi *leaving variable*.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	S_{g1}	NK
Z	0	-0.2995	-0.1995	-0.15	0	-0.00005	-M+0.00005	-0.75	-M+0.75	0	0	0	0	233.79

x_1'	1	0.001	0.001	0	0	-0.001	0.001	0	0	0	0	0	0	0.58
x_5'	0	0	0	1	1	0	0	-1	1	0	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	0	280
s_4	0	-0.001	-0.001	0.5	0	0.001	-0.001	1	-1	0	1	0	0	1.92
s_5	0	0	0	500	0	1	-1	1000	-1000	0	0	1	0	3200
S_1	0	$\frac{1}{999}$	$\frac{1}{999}$	0	0	1	$\frac{1}{999}$	0	0	0	0	0	$-\frac{1000}{999}$	$\frac{80}{999}$

Keempat, kita perlu memperbarui nilai baris-baris lainnya.

VB	x_1'	x_2'	x_3'	x_4'	x_5'	S_1	A_1	S_2	A_2	s_3	s_4	s_5	S_{g1}	NK
Z	0	-0.2994994995	-0.1994994995	-0.15	0	0	$-M+\frac{1}{1998}$	-0.75	-M+0.75	0	0	0	$-\frac{1}{1998}$	233.79004
x_1'	1	$\frac{1}{999}$	$\frac{1}{999}$	0	0	0	$\frac{1}{999}$	0	0	0	0	0	$-\frac{1}{999}$	0.5801
x_5'	0	0	0	1	1	0	0	-1	1	0	0	0	0	2
s_3	0	1	1	0	0	0	0	0	0	1	0	0	0	280
s_4	0	$-\frac{1}{999}$	$-\frac{1}{999}$	0.5	0	0	$-\frac{1}{999}$	1	-1	0	1	0	$\frac{1}{999}$	1.92
s_5	0	$-\frac{1}{999}$	$-\frac{1}{999}$	500	0	0	$-\frac{1000}{999}$	1000	-1000	0	0	1	$\frac{1000}{999}$	3199.92
S_1	0	$\frac{1}{999}$	$\frac{1}{999}$	0	0	1	$\frac{1}{999}$	0	0	0	0	0	$-\frac{1000}{999}$	$\frac{80}{999}$

Tabel ini merupakan hasil sementara dari perhitungan metode *Cutting Plane* dan bukan merupakan hasil akhir. Perlu dibuat lagi skala *Gomory* dan diterapkannya lagi metode *Dual Simplex* sampai x_1' akhirnya berbentuk *integer*. Jika x_1 sudah *integer*, namun masih ada syarat/kendala yang terpenuhi, maka akan bergerak ke iterasi 2 metode *Cutting Plane*. Hanya pada saat x_1' sudah berbentuk *integer* dan tidak ada syarat yang dilanggar baru metode *Cutting Plane* dapat diberhentikan. Jawaban dari skenario mendadak ini akan ditemukan dengan bantuan program di sub-bab berikutnya.

4.2 VALIDASI HASIL DENGAN PROGRAM

Untuk membantu memvalidasi perhitungan di atas beserta mncari hasil optimal untuk skenario mendadak yang telah disebut sebelumnya, kita akan menggunakan bantuan *Google Colab* dengan bahasa pemrograman Python-nya untuk melakukannya.

Ada tiga bagian yang akan dibahas disini, yaitu hasil *Simplex* awal, hasil ILP (dengan variabel x_4 dan x_5 harus bernilai *integer*), dan hasil skenario mendadak (dengan variabel x_1, x_4, dan x_5 harus bernilai *integer*). Untuk mempermudah pencarian jawaban optimal, kita akan menggunakan bantuan Library Python yang bernama 'PuLP' sehingga kita hanya perlu memasukkan variabel-variabel penentu, fungsi tujuan, dan *constraints*.

```
[106] 1 # Install Library PuLP untuk Melakukan Perhitungan Simplex dan Integer Linear Programming (ILP)
7s    2 !pip install pulp

      Requirement already satisfied: pulp in /usr/local/lib/python3.10/dist-packages (2.8.0)

[107] 1 # Import Library untuk Perhitungan Simplex dan ILP
0s    2 import pulp
```

Untuk kasus *Simplex* awal, hal pertama yang harus dilakukan adalah mendefinisikan masalahnya. Untuk skenario kita, masalahnya adalah untuk meminimalisasikan biaya.

```
[108] 1 # Definisikan Kasus Simplex Minimalisasi (Meminimkan Biaya Pembelian Bahan-Bahan Nutrisi)
0s    2 simplex = pulp.LpProblem("Simplex_Awal", pulp.LpMinimize)
```

Selanjutnya, kita harus memasukkan informasi tentang variabel-variabel penentu, seperti nama, batasan (atas dan bawah), dan tipe (*integer* atau *continuous*). Bisa dilihat bahwa semua variabel penentu diatur untuk bisa memberikan jawaban berdesimal (*continuous*).

```
[109] 1 # Karakteristik Variabel Penentu
0s    2 x1 = pulp.LpVariable('x1_VitaminB1', lowBound=1.5, upBound=3, cat='Continuous')
      3 x2 = pulp.LpVariable('x2_VitaminA', lowBound=700, upBound=900, cat='Continuous')
      4 x3 = pulp.LpVariable('x3_VitaminD', lowBound=20, upBound=50, cat='Continuous')
      5 x4 = pulp.LpVariable('x4_Iron', lowBound=10, upBound=12, cat='Continuous')
      6 x5 = pulp.LpVariable('x5_Zinc', lowBound=11, upBound=15, cat='Continuous')
```

Setelah itu, kita harus memasukkan fungsi tujuannya.

```
[110] 1 # Fungsi Tujuan
0s    2 simplex += (0.5 * x1) + (0.3 * x2) + (0.2 * x3) + (0.9 * x4) + (0.75 * x5), "Objective"
```

Kemudian, kita harus memasukkan semua *constraint*-nya (*constraint* disini dimodifikasi sedikit supaya tidak ada pecahan dengan penyebut 1000).

```
[111] 1 # Syarat/Kendala (Constraint)
0s    2 simplex += (1000 * x1) + x2 + x3 >= 2800, "Constraint 1"
      3 simplex += x4 + x5 >= 23, "Constraint 2"
      4 simplex += x2 + x3 <= 1000, "Constraint 3"
      5 simplex += x1 + (1.5 * x4) + x5 <= 32, "Constraint 4"
      6 simplex += (1000 * x1) + x2 + x3 + (1500 * x4) + (1000 * x5) <= 34000, "Constraint 5"
```

Lalu, setelah semua kebutuhan masuk, kita akhirnya bisa mencari hasil optimalnya.

```
[112]  1 # Mencari Hasil Optimal
       2 simplex.solve()

       1
```

Untuk maksud dari angka 1 hasil fungsi solve() adalah ditemukannya solusi optimal. Berikut adalah kemungkinan lain *output*-nya solve():

```
[113]  1 # Arti dari Output solve()
       2 pulp.LpStatus

       {0: 'Not Solved',
        1: 'Optimal',
        -1: 'Infeasible',
        -2: 'Unbounded',
        -3: 'Undefined'}
```

Setelah solve() selesai eksekusi, kita bisa langsung melihat hasilnya dengan menge-*print* seperti berikut:

```
[114]  1 # Print Status Hasil (Apakah Optimal atau Tidak)
       2 print("Status:", pulp.LpStatus[simplex.status])

       Status: Optimal

[115]  1 # Print Nilai Optimal Setiap Variabel Penentu
       2 print("Optimal values:")
       3 for variable in simplex.variables():
       4     print(f"{variable.name} = {variable.varValue}")

       Optimal values:
       x1_VitaminB1 = 2.08
       x2_VitaminA = 700.0
       x3_VitaminD = 20.0
       x4_Iron = 10.0
       x5_Zinc = 13.0

[116]  1 # Print Hasil Optimal Fungsi Tujuan
       2 print("Optimal objective function value:", pulp.value(simplex.objective))

       Optimal objective function value: 233.79
```

Bisa dilihat bahwa status dari hasil perhitungan merupakan ‘Optimal’ yang berarti ini adalah hasil yang terbaik-baiknya untuk biaya seminim mungkin dengan tetap memenuhi syarat-syarat yang ada. Hasil nilai setiap variabel penentu dan juga fungsi tujuan (biaya) pun sama dengan perhitungan manual yang dilakukan sebelumnya. Seperti yang sudah disebut

sebelumnya, semua variabel penentu disini diatur untuk bisa memiliki desimal, namun x_4 dan x_5 (dua variabel penentu yang seharusnya *integer*) berbentuk *integer* disini.

Misal di bagian pengaturan variabel penentu, kita atur x_4 dan x_5 harus bernilai *integer*. Bisa dilihat di bagian 'cat='.

```
[117]  # Definisikan Kasus ILP Minimalisasi (Meminimkan Biaya Pembelian Bahan-Bahan Nutrisi)
0s     ilp = pulp.LpProblem("ILP", pulp.LpMinimize)

[118]  # Karakteristik Variabel Penentu
0s     y1 = pulp.LpVariable('y1_VitaminB1', lowBound=1.5, upBound=3, cat='Continuous')
       y2 = pulp.LpVariable('y2_VitaminA', lowBound=700, upBound=900, cat='Continuous')
       y3 = pulp.LpVariable('y3_VitaminD', lowBound=20, upBound=50, cat='Continuous')
       y4 = pulp.LpVariable('y4_Iron', lowBound=10, upBound=12, cat='Integer')
       y5 = pulp.LpVariable('y5_Zinc', lowBound=11, upBound=15, cat='Integer')
```

Untuk pengaturan lainnya tetap sama dengan percobaan sebelumnya.

```
[119]  # Fungsi Tujuan
0s     ilp += (0.5 * y1) + (0.3 * y2) + (0.2 * y3) + (0.9 * y4) + (0.75 * y5), "Objective"

[120]  # Syarat/Kendala (Constraint)
0s     ilp += (1000 * y1) + y2 + y3 >= 2800, "Constraint 1"
       ilp += y4 + y5 >= 23, "Constraint 2"
       ilp += y2 + y3 <= 1000, "Constraint 3"
       ilp += y1 + (1.5 * y4) + y5 <= 32, "Constraint 4"
       ilp += (1000 * y1) + y2 + y3 + (1500 * y4) + (1000 * y5) <= 34000, "Constraint 5"

[121]  # Mencari Hasil Optimal
0s     ilp.solve()

       1
```

Hal yang berbeda disini hanyalah nama variabel-variabel penentunya (dan nama kasus/masalah-nya yang didefinisikan di bagian pulp.LpProblem()).

```
[122]  1 # Print Status Hasil (Apakah Optimal atau Tidak)
0s     2 print("Status:", pulp.LpStatus[ilp.status])

       Status: Optimal

[123]  1 # Print Nilai Optimal Setiap Variabel Penentu
0s     2 print("Optimal values:")
       3 for variable in ilp.variables():
       4     print(f"{variable.name} = {variable.varValue}")

       Optimal values:
       y1_VitaminB1 = 2.08
       y2_VitaminA = 700.0
       y3_VitaminD = 20.0
       y4_Iron = 10.0
       y5_Zinc = 13.0

[124]  1 # Print Hasil Optimal Fungsi Tujuan
0s     2 print("Optimal objective function value:", pulp.value(ilp.objective))

       Optimal objective function value: 233.79
```

Sekali lagi, hasilnya sama seperti perhitungan manual dan perhitungan yang dilakukan sebelumnya di program saat semua variabel penentu diatur bisa memiliki desimal.

Untuk bagian ketiga, misal ada skenario mendadak di mana x_1 harus berbentuk *integer*. Kita bisa menggunakan bantuan program untuk menemukan solusi optimal dengan skenario seperti itu. Di bab sebelumnya, sudah ditunjukkan cara untuk menghitungnya secara manual, namun hanya sampai poin tertentu. Bisa juga dilihat bahwa perubahan nilai x_1'-nya sangat sedikit per siklus skala *Gomory* – Metode *Dual Simplex* dan kita harus bekerja dengan angka yang sangat kecil $(\frac{1}{999})$ sehingga membutuhkan waktu yang cukup banyak hanya untuk menghitung dan menyesuaikan satu variabel. Oleh karena itu, kita akan menggunakan bantuan program untuk bisa menemukan hasil optimalnya dengan jauh lebih cepat.

Untuk prosedurnya sendiri sama seperti dua bagian sebelumnya.

```
[125]  # Definisikan Kasus ILP Minimalisasi (Meminimkan Biaya Pembelian Bahan-Bahan Nutrisi)
       ilp_other = pulp.LpProblem("ILP_SkenarioMendadak", pulp.LpMinimize)

[126]  # Karakteristik Variabel Penentu
       z1 = pulp.LpVariable('z1_VitaminB1', lowBound=1.5, upBound=3, cat='Integer')
       z2 = pulp.LpVariable('z2_VitaminA', lowBound=700, upBound=900, cat='Continuous')
       z3 = pulp.LpVariable('z3_VitaminD', lowBound=20, upBound=50, cat='Continuous')
       z4 = pulp.LpVariable('z4_Iron', lowBound=10, upBound=12, cat='Integer')
       z5 = pulp.LpVariable('z5_Zinc', lowBound=11, upBound=15, cat='Integer')

[127]  # Fungsi Tujuan
       ilp_other += (0.5 * z1) + (0.3 * z2) + (0.2 * z3) + (0.9 * z4) + (0.75 * z5), "Objective"

[128]  # Syarat/Kendala (Constraint)
       ilp_other += (1000 * z1) + z2 + z3 >= 2800, "Constraint 1"
       ilp_other += z4 + z5 >= 23, "Constraint 2"
       ilp_other += z2 + z3 <= 1000, "Constraint 3"
       ilp_other += z1 + (1.5 * z4) + z5 <= 32, "Constraint 4"
       ilp_other += (1000 * z1) + z2 + z3 + (1500 * z4) + (1000 * z5) <= 34000, "Constraint 5"
```

```
[129]  # Mencari Hasil Optimal
       ilp_other.solve()

       1
```

```
[130]  # Print Status Hasil (Apakah Optimal atau Tidak)
       print("Status:", pulp.LpStatus[ilp_other.status])

       Status: Optimal

[131]  # Print Nilai Optimal Setiap Variabel Penentu
       print("Optimal values:")
       for variable in ilp_other.variables():
           print(f"{variable.name} = {variable.varValue}")

       Optimal values:
       z1_VitaminB1 = 3.0
       z2_VitaminA = 700.0
       z3_VitaminD = 20.0
       z4_Iron = 10.0
       z5_Zinc = 13.0

[132]  # Print Hasil Optimal Fungsi Tujuan
       print("Optimal objective function value:", pulp.value(ilp_other.objective))

       Optimal objective function value: 234.25
```

Sekali lagi, perbedaannya disini dengan program-program sebelumnya adalah penamaan variabel-variabel penentu, penamaan kasus (‘ILP_SkenarioMendadak’), dan x_1 juga diatur harus memiliki nilai berbentuk *integer* (tidak hanya x_4 dan x_5). Bisa dilihat disini bahwa ketika x_1 harus *integer,* **Alokasi dana minimum untuk satu pemain per harinya**

sebesar Rp. 234.250,- dengan detail: Vitamin B1 3 mg, Vitamin A 700 μg, Vitamin D 20 μg, Iron 10 kapsul (1.5 mg), dan Zinc 13 kapsul (1 mg).

Berdasarkan hasil validasi dan skenario mendadak sebelumnya, bisa dibilang bahwa perhitungan manual di sub-bab sebelumnya sudah benar dan meskipun x_1 harus *integer*, solusi yang seoptimal-optimalnya (sebisanya) masih ada.

Link Google Colab Kode: Projek AMM 2540124633 - Google Colab

BAB V
KESIMPULAN

Berdasarkan percobaan (skenario) sebelumnya, kita mampu menemukan solusi optimal dari model matematika dengan menggunakan metode *Integer Linear Programming* (ILP) dengan pendekatan *Gomory* (*Cutting Plane Method*). Untuk menunjang metode tersebut, metode *Simplex* dan *Dual Simplex* digunakan saat proses perhitungan hasil optimal. Dengan segala *constraint* yang ada, kita mampu menemukan solusi optimal dengan detail seperti berikut:

Alokasi dana minimum untuk satu pemain per harinya sebesar Rp. 233.790,- dengan detail:

- Vitamin B1 2.08 mg

- Vitamin A 700 µg

- Vitamin D 20 µg

- Iron 10 kapsul (1.5 mg)

- Zinc 13 kapsul (1 mg)

Daftar Harga:

Nama Bahan	Harga (dalam satuan ribuan rupiah) Per Unit Bahan
Vitamin B1	0.5/mg
Vitamin A	0.3/µg
Vitamin D	0.2/µg
Iron	0.9/kapsul
Zinc	0.75/kapsul

Kita juga mampu menemukan solusi dari skenario mendadak (di mana x_1 harus *integer*) dengan bantuan program dengan detail seperti berikut:

Alokasi dana minimum untuk satu pemain per harinya sebesar Rp. 234.250,- dengan detail:

- Vitamin B1 3 mg

- Vitamin A 700 μg

- Vitamin D 20 μg

- Iron 10 kapsul (1.5 mg)

- Zinc 13 kapsul (1 mg)

Dengan daftar harga yang sama seperti sebelumnya.

Sepakbola menjadi salah satu olahraga yang paling terkenal dan kompetitif di dunia. Dengan semakin banyaknya pemain yang ada dan tersedia, suatu klub sepakbola juga mau bisa mengelola kesehatan dan kebugaran para pemainnya dengan lebih efektif dan efisien, terlebih dalam hal asupan nutrisi. Dengan adanya penelitian ini, diharapkan proses pengelolaan (manajemen) tersebut bisa terbantu menjadi lebih mudah dan praktis. Terlebih lagi pada saat periode pramusim, di mana para pemain bersiap-siap untuk memasuki musim liga yang panjang.

Saran yang dapat diberikan untuk para peneliti yang meneliti topik sejenis kedepannya adalah untuk menggunakan data yang mencakup semua pemain sepakbola pria tingkat profesional dengan rentang umur yang lebih luas, bahkan sampai ke rentang umur remaja atau lebih tua dari umur 30 tahun. Penelitian ini mengasumsi bahwa pemain-pemain sepakbola yang dikelola klub berada di rentang umur 24-30 tahun, di mana pada rentang umur inilah para pemain disebut dalam kondisi 'prima'-nya. Namun, dengan observasi kita sendiri, kita bisa lihat bahwa sekarang ini ada banyak pemain yang sudah terjun ke dunia sepakbola profesional saat masih berumur 16 atau 17 tahun. Ada juga pemain sepakbola yang masih melanjutkan karir profesionalnya saat sudah cukup berumur (seperti 35-39 tahun). Demikian saran yang bisa diberikan penulis untuk para peneliti kedepannya.

REFERENSI

Książek, A., Zagrodna, A., & Słowińska-Lisowska, M. (2020). Assessment of the Dietary Intake of High-Rank Professional Male Football Players during a Preseason Training Week. *International Journal of Environmental Research and Public Health, 17*(22), 8567. https://doi.org/10.3390/ijerph17228567

Burke, L., & Deakin, V. (2015). *Clinical Sports Nutrition*. McGraw-Hill Education.

Maughan, R. (2006). *Nutrition and Football: The FIFA/FMARC Consensus on Sports Nutrition*. Routledge.

Burke, L. M., Hawley, J. A., Wong, S. H. S., & Jeukendrup, A. E. (2011). Carbohydrates for Training and Competition. *Journal of Sports Sciences*, *29*(sup1), S17–S27. https://doi.org/10.1080/02640414.2011.585473

Tipton, K. D., & Wolfe, R. R. (2004). Protein and Amino Acids for Athletes. *Journal of Sports Sciences*, *22*(1), 65–79. https://doi.org/10.1080/0264041031000140554

Oliveira, C. C., Ferreira, D., Caetano, C., Granja, D., Pinto, R., Mendes, B., & Sousa, M. (2017). Nutrition and Supplementation in Soccer. *Sports*, *5*(2), 28. https://doi.org/10.3390/sports5020028

Harding, J. (2018). *The Cost of Feeding a Bundesliga Team*. Terakhir diakses melalui https://www.dw.com/en/the-cost-of-feeding-a-bundesliga-team/a-42560126 pada tanggal 18 Juni 2024.