APLIKASI PEMROGRAMAN LINIER SIMPLEKS DUA FASE BERBASIS *WEB* MENGGUNAKAN *ANGULAR JAVASCRIPT FRAMEWORK*

Adityo Rancaka[1)], Tri Handhika[2)]

[1)]Jurusan S1 Teknik Informatika, Fakultas Teknologi Industri, Universitas Gunadarma,
[2)]Pusat Studi Komputasi Matematika, Universitas Gunadarma
Jl. Margonda Raya 100, Depok, 16424
Email : rancakadityo@gmail.com[1)], trihandika@staff.gunadarma.ac.id[2)]

Abstrak

Pemrograman linier adalah salah satu dari beberapa metode pencarian kombinasi angka untuk memaksimumkan atau meminimumkan fungsi tujuan agar dapat menghasilkan solusi optimal. Namun demikian, perhitungan secara manual memerlukan waktu cukup lama disebabkan oleh kompleksnya rumusan dari Pemrograman Linier tersebut. Dalam makalah ini, penulis merancang suatu aplikasi untuk menyelesaikan permasalahan Pemrograman Linier Simpleks Dua Fase berbasis web menggunakan Angular JavaScript Framework. Aplikasi tersebut dapat menyelesaikan permasalahan Program Linier dengan akurat setelah model matematika dimasukkan. Karena metode yang digunakan adalah Metode Simpleks, maka aplikasi juga mampu menyelesaikan permasalahan yang memiliki lebih dari 3 variabel keputusan. Hasil yang diperoleh adalah kombinasi angka pada variabel keputusan beserta solusi optimalnya.

Kata kunci: *Pemrograman Linier, Metode Simpleks, Metode Simpleks Dua Fase, JavaScript, Angularjs.*

1. Pendahuluan

Dalam dunia usaha, seorang *entrepreneur* berharap untuk memperoleh keuntungan yang maksimum dengan biaya minimum dari bidang usaha yang digelutinya. Oleh sebab itu, pengusaha tersebut harus membuat perencanaan untuk mengoptimisasi sumber daya yang tersedia, seperti bahan baku, transportasi ataupun sumber daya manusia. Jika keputusan yang diambil tersebut tidak akurat, maka dapat berdampak buruk bahkan fatal bagi perusahaan yang mereka pimpin. Dengan demikian, dibutuhkan teknik untuk mengoptimisasi keuntungan maupun biaya tersebut melalui sumber daya yang tepat agar mampu memberikan hasil optimisasi yang akurat bagi lembaga atau perusahaan terkait [1].

Terdapat beberapa teknik yang dapat digunakan, salah satunya adalah Pemrograman Linier dengan Metode Simpleks Dua Fase. Kelebihan yang dimiliki Metode Simpleks Dua Fase mampu menyelesaikan masalah yang memiliki sembarang jumlah variabel keputusan karena metode tersebut menggunakan tabel untuk proses iterasinya. Tidak seperti Metode Grafik yang hanya dapat menyelesaikan permasalahan maksimal dengan 3 variabel keputusan [2]. Namun, untuk melakukan perhitungan Metode Simpleks Dua Fase secara manual memerlukan waktu cukup lama disebabkan oleh kompleksnya algoritma yang harus dikerjakan. Terlebih jika permasalahan memiliki banyak variabel keputusan dan kendala. Oleh sebab itu, dibutuhkan suatu alat hitung seperti aplikasi komputer untuk membantu proses perhitungan tersebut.

Pada penelitian kali ini, penulis merancang sebuah aplikasi berbasis *web* yang dapat menyelesaikan permasalahan Pemrograman Linier yang memiliki sembarang jumlah variabel. Aplikasi dibangun menggunakan 2 bahasa pemrograman. Program inti yang bertanggung jawab atas proses perhitungan, sepenuhnya ditulis dengan *JavaScript* menggunakan *Angular JavaScript Framework (Angularjs).* Sedangkan untuk tampilan ditulis menggunakan HTML5 dan *Bootstrap* [3].

Beberapa penelitian terkait dengan aplikasi metode simpleks berbasis web diantaranya menggunakan bahasa pemrograman PHP dengan iterasi maksimal adalah sebanyak 10 kali iterasi [4]. Permasalahan yang dapat terjadi pada aplikasi tersebut adalah setiap tahapan proses yang dikerjakan haruslah memuat ulang halaman karena PHP tidak dapat merubah halaman *web* secara dinamis seperti *JavaScript.* Seluruh kode program yang ditulis dengan PHP pun dimuat dan dijalankan pada sisi *server*, sehingga jika terjadi putus koneksi antara *client* dan *server* ditengah-tengah proses, aplikasi tidak dapat dijalankan ke tahap berikutnya. Berbeda dengan aplikasi yang dibangun menggunakan *JavaScript* dimana seluruh kode programnya sudah dimuat pada sisi *client* setelah halaman web dibuka. Dengan demikian, jika terjadi putus koneksi aplikasi masih tetap dapat dijalankan [5].

2. Pembahasan

2.1 Aplikasi Pemrograman Linier Simpleks Dua Fase

Penulis merumuskan bahwa Algoritma Simpleks Dua Fase dapat diklasifikasikan menjadi 4 sub bagian yaitu

Tahap Inisialisasi, Tahap Rumusan Masalah, Tahap Iterasi dan *Tahap Konklusi.*

Pada sub bab ini akan dijelaskan mengenai bagaimana keempat sub bagian algoritma tersebut dapat diimplementasikan menjadi sebuah aplikasi berbasis web secara garis besarnya menggunakan *Angularjs* dan *Bootstrap*.

Dalam tahap implementasi, *Angularjs* digunakan sebagai kode program utama dari aplikasi mulai dari inisialisasi hingga konklusi. Karena semua hal yang berkaitan dengan jalannya aplikasi *web* secara dinamis ditulis menggunakan *JavaScript* [5]. *Bootstrap* digunakan untuk mempercantik tampilan halaman *web* yang ditulis menggunakan HTML5 seperti *input box, navigation bar, button, table* dan *grid system* [3].

2.1.1 Tahap Inisialisasi

Tahap Inisialisasi adalah tahap awal dimana model matematika yang telah dibuat kemudian dimasukkan untuk ditentukan solusi awal pada fase 1.

2.1.1.1 Algoritma Tahap Inisialisasi

1. Mendeklarasikan seluruh variabel yang akan digunakan di dalam proses jalannya program.
2. Memasukkan model matematika yang telah dibuat secara manual oleh pengguna kedalam program. Variabel tersebut adalah *tujuan, jumlah variabel tujuan, jumlah fungsi kendala, koefisien fungsi tujuan, koefisien fungsi kendala, operator,* dan *ruas kanan fungsi kendala.*

2.1.1.2 Kode Program Tahap Inisialisasi

```
<select ng-model="tujuan">
<option value="max"> Maksimisasi</option>
<option value="min"> Minimisasi</option>
</select>
<input type="number" ng-model="jvt">
<input type="number" ng-model="jfk">
```

Pada blok program yang pertama di tahap inisialisasi ini digunakan untuk memasukkan tujuan permasalahan apakah maksimisasi atau minimisasi kedalam variabel *tujuan*. Serta juga jumlah variabel tujuan (*(jvt)* dan jumlah fungsi kendala *(jfk).* Atribut *ng-model* adalah API *Angularjs* yang digunakan untuk menyimpan suatu data ke dalam variabel.

```
<td ng-repeat="col in getNumber(jvt) track by $index">
<input type="number" ng-model="koef_ft[$index]">
</td>
```

Blok program yang berikutya digunakan untuk memasukkan koefisien fungsi tujuan. Atribut *ng-repeat* adalah API dari *Angularjs* yang digunakan untuk melakukan perulangan pada elemen HTML5. Dan variabel $index adalah indeks dari perulangan tersebut.

```
<div ng-repeat="row in getNumber(jfk) track by
$index">
<td ng-repeat="col in getNumber(jvt) track by $index">
<input type="number" ng-
model="koef_fk[0][$parent.$index][$index]">
</td>
<select data-ng-model="op[$index]">
 <option value="eq">&equals;</option>
<option value="le">&le;</option>
<option value="ge">&ge;</option>
</select>
<input type="number" ng-model="rk[0][$index]">
</div>
```

Blok program selanjutnya bertanggung jawab untuk membaca masukkan koefisien fungsi tujuan, operator dan ruas kanan.Proses tersebut dilakukan menggunakan perulangan bersarang.

2.1.2 Tahap Rumusan Masalah

Tahap Rumusan Masalah adalah tahap dimana model matematika yang telah dimasukkan di verifikasi dan di format untuk di dapatkan rumusan masalahnya sehingga dapat dibuat solusi awal fase 1.

2.1.2.1 Algoritma Tahap Rumusan Masalah

1. Ruas kanan (RK) fungsi tujuan sama dengan 0.
2. Ruas kanan fungsi kendala adalah bilangan positif. Jika negatif, nilai kedua ruas dikalikan dengan -1, dan merubah simbol "≤" menjadi "≥" begitupun sebaliknya.
3. Fungsi kendala dengan pertidaksamaan "≤" diubah ke dalam bentuk persamaan "=" dengan menambahkan variabel *slack* (+1). Fungsi kendala dengan tanda "≥" diubah pula ke dalam bentuk persamaan "=" dengan menambahkan variabel *surplus* (-1) serta variabel buatan (+1). Fungsi kendala dengan tanda "=" ditambahkan dengan variabel buatan (+1).
4. Proses yang terakhir pada *Tahap Rumusan Masalah* adalah proses untuk menentukan solusi awal fase 1. Terdapat 2 macam kondisi pada penentuan solusi awal. Pertama, kondisi dimana tidak terdapat variabel buatan pada semua fungsi kendala. Solusi awal tersebut akan dibentuk dengan fungsi objektif yang sama dengan fungsi tujuannya [2].
   Untuk memaksimumkan :
   $$- C_1X_1 - C_2X_2 - \ldots - C_mX_m$$
   Untuk meminimumkan :
   $$C_1X_1 + C_2X_2 + \ldots + C_mX_m$$
   Kondisi yang kedua adalah jika terdapat variabel buatan pada salah satu atau beberapa fungsi kendala. Solusi awalnya dibentuk dengan fungsi objektif sebagai berikut.

Untuk memaksimumkan :

$$-X_{m+n+1} - X_{m+n+2} - \ldots - X_{m+n+o}$$

Untuk meminimumkan :

$$X_{m+n+1} + X_{m+n+2} + \ldots + X_{m+n+o}$$

Ket :
C = Konstanta.
X = Variabel.
m = Jumlah variabel keputusan.
n = Jumlah variabel slack.
o = Jumlah variabel buatan.

2.1.2.2 Kode Program Tahap Rumusan Masalah

```
<button class="btn btn-success" ng-click="optimize()">
   <span ng-if="tujuan == 'max'">Maksimalkan!</span>
   <span ng-if="tujuan == 'min'">Minimalkan!</span>
</button>
```

Tombol HTML5 diatas adalah satu - satunya tombol yang harus di tekan oleh pengguna untuk menghasilkan solusi optimal dari permasalahan. Dengan atribut *ng-click* dari *Angularjs,* yang akan dengan segera memanggil prosedur *optimize()* yang berisi perintah untuk mengeksekusi tahap dari mulai rumusan masalah hingga konklusi.

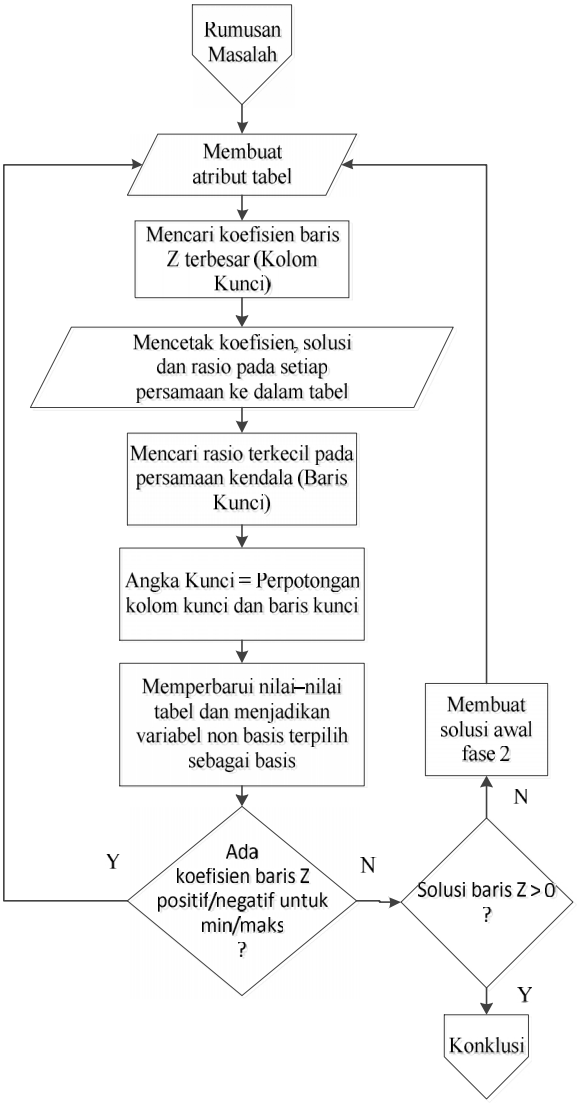
```
Z{{ tujuan }} =
<span ng-repeat="col in getNumber((jvt+jss+ja[1]))
track by $index">
{{ koef_Z[1][$index] }}
</span>
= {{ rk_Z[1] }}

<div ng-repeat="row in getNumber(jfk) track by
$index">
<span ng-repeat="col in getNumber((jvt+jss+ja[1]) track
by $index">
{{ koef_fk[1][$parent.$parent.$index][$index] }}
</span>
<span ng-if="op[$index] == 'eq'">&equals;</span>
<span ng-if="op[$index] == 'le'">&le;</span>
<span ng-if="op[$index] == 'ge'">&ge;</span>
{{ rk[1][$index] }}
</div>
```

Elemen - elemen HTML5 yang telah di tanamkan atribut *Angularjs* diatas digunakan untuk menampilkan rumusan masalah setelah tombol di tekan.

2.1.3 Tahap Iterasi

Setelah diperoleh solusi awal, kemudian masukkan solusi tersebut ke dalam tabel awal simpleks sebagai baris Z. Algoritma pun masuk ke dalam *Tahap Iterasi* yang akan diselesaikan secara terurut dan sistematis.

2.1.3.1 Algoritma Tahap Iterasi



Gambar 1. *Diagram Alir Tahap Iterasi*

1. Menentukan kolom kunci (KK), yaitu kolom dengan nilai negatif terbesar pada baris Z, jika tujuannya adalah memaksimumkan atau kolom dengan nilai positif terbesar pada baris Z, jika tujuannya adalah meminimumkan.
2. Menghitung rasio tiap baris dengan rumus: Rasio = RK / Koef. KK.
3. Menentukan baris kunci (BK) yaitu baris dengan rasio positif terkecil.
4. Menentukan angka kunci (AK) yaitu perpotongan antara KK dan BK.
5. Membuat baris kunci baru (BKB) dengan rumus BKB = BK / AK.
6. Memperbarui nilai tiap baris selain baris kunci dengan rumus Baris Baru = Baris Lama – Koef. KK x BKB
7. Mengeluarkan variabel basis yang terpilih dari baris kunci, dan menjadikan variabel non basis yang terpilih dari kolom kunci sebagai basis.
8. Memasukkan seluruh nilai pada BKB dan Baris Baru ke dalam tabel simpleks baru.

Setiap tahap pada nomor 1 – 8 adalah proses iterasi yang dijalankan secara kontinu hingga diperoleh solusi optimal atau ditemukan bahwa solusi awal Pemrograman Linier tersebut tidak memiliki solusi optimal sama sekali. Berikut adalah beberapa keadaan yang menentukan sampai kapan proses iterasi perlu dijalankan atau kapan algoritma masuk ke dalam fase kedua.

1. Jika masih ada kolom dengan nilai negatif pada baris Z (memaksimumkan), atau kolom dengan nilai positif pada baris Z (meminimumkan), algoritma mengulang iterasi dari tahap pertama.
2. Jika sudah tidak ada, periksa RK pada baris Z. Jika RK lebih dari 0, maka solusi optimal telah berhasil diperoleh dan program masuk ke dalam tahapan konklusi. Sebaliknya, jika RK kurang dari atau sama dengan 0, maka diperoleh solusi layak pada fase 2 sehingga algoritma pun masuk ke dalam fase 2.

Pada fase 1, telah dihilangkan variabel buatan dan membuat solusi layak. Pada fase 2, algoritma tersebut akan menyelesaikan solusi yang dibentuk pada fase 1 untuk kemudian menemukan solusi optimal. Berikut adalah proses pada fase 2:

1. Menghapus semua kolom variabel buatan yang terdapat pada tabel simpleks.
2. Membuat solusi awal fase 2 dengan fungsi objektif yang sama dengan fungsi tujuan.
   Untuk memaksimumkan :
   $$- C_1X_1 - C_2X_2 - \ldots - C_mX_m$$
   Untuk meminimumkan :
   $$C_1X_1 + C_2X_2 + \ldots + C_mX_m$$
3. Memasukkan solusi awal fase 2 dan semua fungsi kendala pada tabel akhir fase 1 kedalam tabel awal fase 2.
4. Kembali ke tahap awal iterasi.

2.1.3.2 Kode Program Tahap Iterasi

```
<div ng-repeat="table in getNumber(countIteration)
track by $index">
Z{{ tujuan }}
<td ng-repeat="col in getNumber((jvt+jss+ja[1])) track
by $index">
{{ koef_Z[$parent.$index+1][$index].toFixed(2) }}
</td>
{{ rk_Z[$index+1].toFixed(2) }}
<tr ng-repeat="row in getNumber(jfk) track by $index">
{{ basis[$parent.$index+1][$index]+1 }}</small></var>
<td ng-repeat="col in getNumber((jvt+jss+ja[1])) track
by $index">
{{
koef_fk[$parent.$parent.$index+1][$parent.$index][$ind
ex].toFixed(2) }}
</td>
{{ rk[$parent.$index+1][$index].toFixed(2) }}
{{ ratio[$parent.$index+1][$index].toFixed(2) }}
</tr></div>
```

Pada blok program *Tahap Iterasi*, untuk menampilkan koefisien fungsi kendala yang akan ditampilkan pada tabel iterasi, harus menggunakan perulangan *ng-repeat* sebanyak 3 lapis. Lapisan terluar digunakan untuk perulangan tabel, lapisan didalamnya digunakan untuk perulangan baris, dan lapisan yang paling dalam digunakan untuk perulangan kolom. Maka dari itu, variabel *koef_fk* yang bertanggung jawab untuk menyimpan nilai koefisien fungsi kendala adalah variabel array 3 dimensi.

2.1.4 Tahap Konklusi

Tahap Konklusi adalah tahap terakhir yang dieksekusi apabila pencarian kombinasi angka untuk memaksimumkan atau meminimumkan fungsi tujuan telah berhasil didapatkan baik itu melalui fase 1 ataupun fase 2.

2.1.4.1 Algoritma Tahap Konklusi

1. Mensubtitusikan nilai dari masing-masing variabel keputusan ke dalam fungsi tujuan untuk memperoleh solusi optimal.
2. Mencetak koefisien fungsi tujuan, variabel keputusan serta solusi optimal.

2.1.4.2 Kode Program Tahap Konklusi

```
<span ng-repeat="col in getNumber(jvt) track by
$index">
<var>X<small>{{ $index + 1 }}</small></var> = {{
X[$index][jvt+jss].toFixed(2) }}
</span>
```

Blok program pada tahap konklusi diatas untuk menampilkan nilai dari variabel tujuan yang telah didapat dari iterasi. Nilai variabel tujuan tersebut telah disimpan dalam variabel X.

```
Z{{ tujuan }} =
<span ng-repeat="col in getNumber(jvt) track by
$index">
{{ koef_ft[$parent.$index] }} ({{
X[$parent.$index][jvt+jss].toFixed(2) }})
</span>
= {{ Z.toFixed(2) }}
```

Blok program yang terakhir masih dalam *Tahap Konklusi,* yang digunakan untuk menampilkan solusi optimal dari permasalahan yang dicari.

2.2 Uji Coba Aplikasi Pemrograman Linier Simpleks Dua Fase

Selain algoritma yang efisien untuk diimplementasikan kedalam program komputer [6], kelebihan lain yang dimiliki oleh Metode Simpleks Dua Fase adalah dapat

menyelesaikan permasalahan Pemrograman Linier yang memiliki sembarang jumlah variabel keputusan. Tidak seperti Metode Grafik yang hanya dapat menyelesaikan permasalahan dengan maksimal 3 variabel keputusan. Berikut ini adalah contoh uji coba aplikasi terhadap permasalahan yang memiliki 4 variabel keputusan.

Fungsi Tujuan :

Maksimumkan $Z = 4X_1 + 6X_2 + 7X_3 + 8X_4$

Fungsi Kendala :

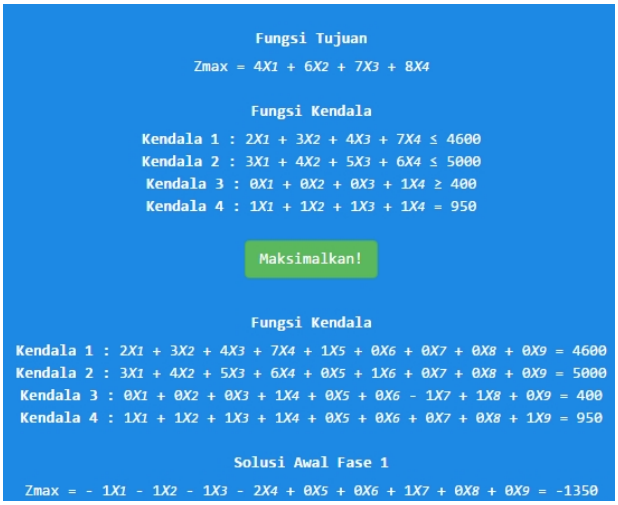
1) $2X_1 + 3X_1 + 4X_3 + 7X_4 \leq 4600$
2) $3X_1 + 4X_2 + 5X_3 + 6X_4 \leq 5000$
3) $0X_1 + 0X_2 + 0X_3 + 1X_4 \geq 400$
4) $1X_1 + 1X_2 + 1X_3 + 1X_4 = 950$

Fungsi tujuan pada permasalahan di atas memiliki 4 variabel keputusan, yakni $4X_1$, $6X_2$, $7X_3$ dan $8X_4$. Berikut pada **Gambar 2a** dan **Gambar 2b** adalah *Tahap Insisialisasi* yang diawali dengan memasukkan model matematika yang telah dibuat secara manual. Pada **Gambar 2a** adalah proses memasukkan ciri dari permasalahan yang ingin di cari solusi optimalnya. Ciri tersebut meliputi jenis optimisasi apakah itu maksimisasi atau minimisasi, jumlah variabel tujuan dan jumlah fungsi kendala.

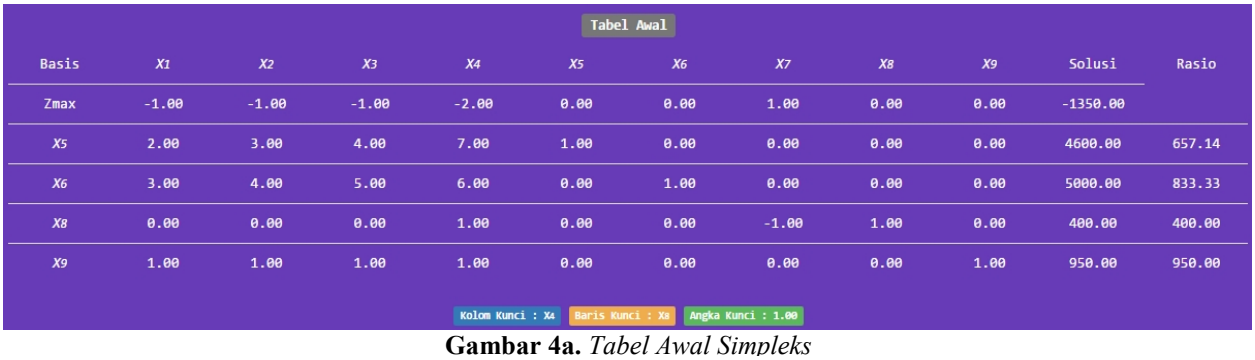
Gambar 2a. *Inisialisasi Ciri Permasalahan*

Pada **Gambar 2b** adalah tampilan untuk memasukkan fungsi tujuan dan fungsi kendala yang langsung terbentuk setelah ciri permasalahan dimasukkan. Setelah itu, **Gambar 3** adalah *Tahap Rumusan Masalah* yang menampilkan jika tombol *"Maksimalkan!"* di tekan, *Angularjs* secara langsung akan memulai proses eksekusi permasalahan Pemrograman Linier.

Gambar 2b. *Inisialisasi Koefisien Fungsi Tujuan dan Fungsi Kendala*



Gambar 3. *Rumusan Masalah Fase 1*



Gambar 4a. *Tabel Awal Simpleks*

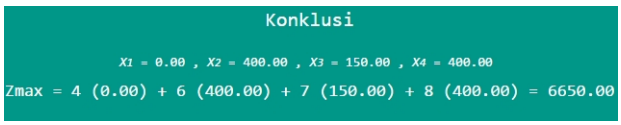
Iterasi 6 & Tabel Akhir

Basis	X1	X2	X3	X4	X5	X6	X7	X8	X9	Solusi	Rasio
Zmax	1.00	0.00	0.00	0.00	1.00	0.00	2.00			6650.00	
X3	-1.00	0.00	1.00	0.00	1.00	0.00	4.00			150.00	
X6	0.00	0.00	0.00	0.00	-1.00	1.00	-2.00			250.00	
X4	0.00	0.00	0.00	1.00	0.00	0.00	-1.00			400.00	
X2	2.00	1.00	0.00	0.00	-1.00	0.00	-3.00			400.00	

Proses iterasi berakhir pada iterasi ke 6. Solusi optimal telah ditemukan!

Gambar 4b. *Tabel Akhir Simpleks*

Pada **Gambar 4b,** adalah tabel simpleks yang terakhir. Setelah dilakukan 6 kali iterasi, pada iterasi ke 6 program mendapatkan kombinasi angka yang tepat pada keempat variabel keputusan yaitu $X_1 = 0$, $X_2 = 400$, $X_3 = 150$, dan $X_4 = 400$. Dan jika nilai tersebut di subtitusikan ke dalam fungsi tujuan, akan menghasilkan solusi optimal untuk memaksimumkan Z = 6650 seperti ditampilkan pada **Gambar 5** yaitu *Tahap Konklusi* untuk mencetak hasil dari permasalahan Program Linier.



Gambar 5. *Tahap Konklusi*

Aplikasi Pemrograman Simpleks Dua Fase ini memungkinkan seorang pengguna yang ingin mencari kombinasi angka untuk memaksimumkan atau meminimumkan suatu permasalahan Pemrograman Linier hanya dengan satu kali tekan tombol setelah model matematika dimasukkan.

3. Kesimpulan

Berdasarkan hasil penelitian ini diketahui bahwa Algoritma Simpleks Dua Fase dapat diimplementasikan ke dalam aplikasi komputer berbasis *web* menggunakan *Angularjs Framework*. Algoritma yang digunakan pun mampu menghasilkan kombinasi angka pada permasalahan Pemrograman Linier yang memiliki 4 variabel keputusan. Setelah kombinasi angka tersebut disubtitusikan kedalam fungsi tujuan, akan menghasilkan angka yang sama dengan solusi optimal pada baris Z, sehingga hasil dari aplikasi sudah akurat.

Pada pembuatan tampilan aplikasi ini, masih terdapat beberapa hal yang harus dikembangkan lebih jauh untuk membuat aplikasi menjadi lebih interaktif seperti memberikan blok warna pada kolom kunci, baris kunci dan angka kunci pada setiap tabel iterasi. Aplikasi juga belum memiliki fitur untuk menampilkan bilangan dalam bentuk pecahan. Pada iterasi fase 2 pun kolom variabel buatan masih tetap ditampilkan padahal sudah tidak digunakan.

Daftar Pustaka

[1]. Hillier F.S dan Gerald J. Lieberman, *Introduction to Operations Research Tenth Edition*, McGraw-Hill Education, New York, 2015.

[2]. Hamdi A. Taha, *Operations Research an Introduction Eight Edition,* Pearson Education, New Jersey, 2007.

[3]. Stephen Radford, *Learning Web Development with Bootstrap and AngularJS,* Pack Publishing, Birmingham, 2015.

[4]. Indrawirawan N.G, Darmawiguna Mahendra G.I, dan Sunarya Gede M.I, "Aplikasi Simulasi Metode Simpleks Untuk Pembelajaran Riset Operasional Berbasis Web" *Kumpulan Artikel Mahasiswa Pendidikan Teknik Informatika (KARMAPATI)* Volume 2 Nomor 6, 2013.

[5]. Bevacqua Nicolas, *JavaScript Application Design,* Manning Publications, New York, 2015.

[6]. Arsham H, "A Computationally Stable Solution Algorithm for Linier Programs" *Applied Mathematics and Computation* 188 1549-1561 University of Baltimore, Baltimore, 2007.

Biodata Penulis

Adityo Rancaka, Mahasiswa Jurusan Teknik Informatika Universitas Gunadarma. Saat ini menjadi Asisten Lembaga Pengembangan Komputerisasi di Universitas Gunadarma.

Tri Handhika, memperoleh gelar Sarjana Sains (S.Si) dan Magister Sains (M.Si) Jurusan Matematika Universitas Indonesia, masing-masing lulus tahun 2008 dan 2011. Gelar Doktor (Dr) Teknologi Informasi diperolehnya dari Universitas Gunadarma pada tahun 2015. Saat ini menjadi Dosen di Pusat Studi Komputasi Matematika, Universitas Gunadarma dan Magister Manajemen Aktuaria, Universitas Indonesia. Selain itu, saat ini pun menjadi *researcher* di PT Mediatrac Sistem Komunikasi dan Image Power Communications.