Penerapan Program Dinamis untuk Optimisasi Taktik *Pit Stop* F1

Marchy Tio Pandapotan[1]
13509026
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13509026@std.stei.itb.ac.id

***Abstract*—Formula satu, disingkat F1(atau bernama lengkap The FIA Formula One World Championship), adalah kelas tertinggi dalam balapan mobil formula dengan tempat duduk tunggal. F1 terdiri dari sejumlah seri balapan yang dikenal dengan istilah Grand Prix. Balapan biasanya diselenggarakan di dalam sirkuit ataupun jalan umum dalam kota yang ditutup untuk umum. Hasil dari F1 adalah dua gelar juara dunia, satu untuk pembalap dan satu lagi untuk konstruktor. Nama olahraga ini, Formula Satu, mengindikasikan bahwa ini merupakan olahraga yang paling maju dan kompetitif di antara balapan mobil formula lain. Sebuah seri balapan Formula Satu dilaksanakan pada akhir minggu, dimulai dengan dua kali latihan bebas pada hari Jumat dan dua kali latihan bebas pada hari Sabtu. Balapan biasanya dilakukan pada hari Minggu. Selama balapan, pebalap bisa melakukan beberapa kali pit stop untuk mengganti ban dan memperbaiki kerusakan. Masing-masing tim beserta pebalapnya menggunakan strategi pit stop yang berbeda untuk memaksimalkan potensi dari mobil. Setiap pebalap harus melakukan minimal satu kali pit stop untuk menggunakan dua jenis ban yang disediakan. Tidak ada batas maksimal untuk pit stop. Makalah ini akan membahas tentang penerapan program dinamis untuk optimalisasi jumlah pit stop dan strategi yang dilakukan oleh masing-masing tim dan pengendara.**

***Index Terms*—program dinamis, pit stop, F1, strategi.**

I. PENDAHULUAN

Pit stop adalah salah satu kegiatan yang berlangsung selama balapan F1. Kegiatan ini biasanya terdiri dari penggantian ban dan perbaikan kerusakan yang terjadi pada mobil balap F1 (sampai dengan musim 2010, pebalap juga dapat mengisi ulang bahan bakar). Setiap tim memiliki berbagai strategi pit stop untuk memaksimalkan kemampuan dari mobil balap. Dua jenis komposisi ban dengan karakteristik yang berbeda, ketahanan dan kerekatan, tersedia untuk pebalap. Selama berlangsungnya balapan, pebalap harus menggunakan kedua jenis ban tersebut. Setiap pebalap harus menggunakan satu kali kesempatan pit stop untuk menggunakan kedua jenis ban yang tersedia. Biasanya, pit stop dilakukan sampai dengan tiga kali, namun tidak ada batas tertentu untuk jumlah pit stop yang dapat dilakukan. Setiap komposisi ban mempunyai keuntungan performansi dari jenis yang lain, dan pemilihan waktu pemakaian adalah sebuah pilihan taktis yang harus diambil.

Oleh karena itu, dalam makalah ini penulis akan mencoba untuk membahas optimasi dari strategi pit stop yang dapat digunakan tim untuk memaksimalkan kemampuan dari mobil. Masalah optimalisasi merupakan salah satu contoh masalah yang dapat diselesaikan dengan program dinamis. Yang akan dibahas dalam makalah ini adalah pemilihan strategi optimal dari permasalah tersebut dengan menggunakan algoritma program dinamis.

II. LANDASAN TEORI

A. Pengenalan Algoritma Program Dinamis

Program dinamis (*dynamic programming)* adalah metode pemecahan masalah dengan cara menguraikan solusi menjadi sekumpulan langkah (step) atau tahapan (stage) sedemikian sehingga solusi dari persoalan dapat dipandang dari serangkaian keputusan yang saling berkaitan. Pada penyelesaian persoalan dengan metode ini terdapat sejumlah berhingga pilihan yang mungkin, solusi pada setiap tahap dibangun dari hasil solusi tahap sebelumnya, dan kita menggunakan persyaratan optimasi dan kedala untuk membatasi sejumlah pilihan yang harus dipertimbangkan pada suatu tahap.

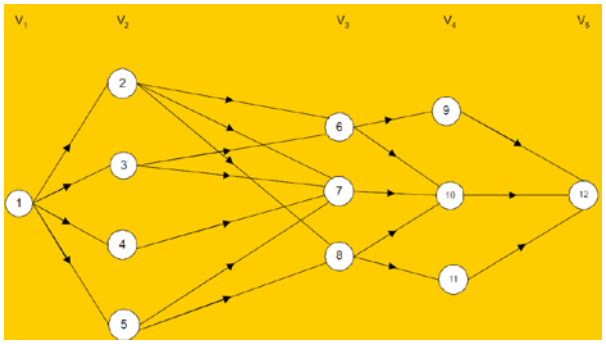
Cara penyelesaian dengan metode program dinamis mengingatkan kita pada metode *greedy*, kita membuat keputusan setiap tahap berdasarkan pilihan yang paling menarik (yang memenuhi ukuran optimasi yang digunakan). Pengambilan keputusan pada setiap tahap hanya pada informasi lokal, dan pada setiap tahap itu kita tidak pernah membuat keputusan yang salah. Pada kelas-kelas persoalan tertentu, algoritma *greedy* bekerja dengan baik, tetapi pada kebanyakan persoalan yang lain tidak. Hal ini terjadi karena pengambilan keputusan pada setiap langkah *greedy* tidak pernah mempertimbangkan lebih jauh apakah pilihan tersebut pada langkah-langkah selanjutnya merupakan pilihan yang tepat. Dari sejumlah pilihan yang menarik, kita tidak tahu pilihan mana yang

terbaik sampai kita menuju proses yang lebih jauh ke depan.

B. *Karakteristik Persoalan Program Dinamis*

Program dinamis diterapkan pada persoalan yang memiliki karakteristik sebagai berikut :

1. Persoalan dapat dibagi menjadi beberapa tahap (*stage*), yang pada setiap tahapnya diambil keputusan.
2. Masing-masing tahap terdiri dari sejumlah status (*state*) yang berhubungan degan tahap tersebut. Secara umum, status merupakan bermacam kemungkinan masukan yang ada pada tahap tersebut. Jumlah status bisa berhingga (*finite*) atau tidak berhingga (*infinite*). Gambar 1 memperlihatkan perbedaan antara tahap dan status yang diberikan pada graf multitahap. Tiap simpul di dalam graf tersebut menyatakan status, sedangkan V_1, V_2, ... menyatakan tahap.
3. Hasil dari keputusan yang diambil pada setiap tahap ditransformasikan dari status yang bersangkutan ke status berikutnya pada tahap berikutnya.
4. Ongkos (*cost*) pada suatu tahap meningkat secara teratur dengan bertambahnya jumlah tahapan.
5. Ongkos pada suatu tahap bergantung pada ongkos tahap-tahap yang sudah berjalan dan ongkos pada tahap tersebut.
6. Keputusan terbaik pada suatu tahap bersifat independen terhadap keputusan yang dilakukan pada tahap sebelumnya.
7. Adanya hubungan rekursif yang mengidentifikasikan keputusan terbaik untuk setiap status pada tahap k memberikan keputusan terbaik untuk setiap status pada tahap $k + 1$.
8. Prinsip optimalitas berlaku pada persoalan tersebut.



Gambar 1 Graf yang menyatakan tahap (*stage*) dan status (*state*)

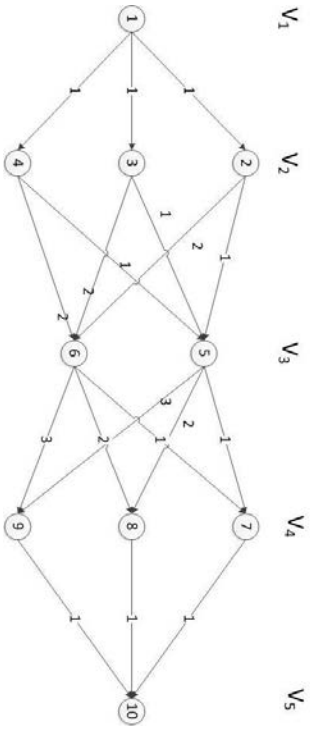
Dalam menyelesaikan persoalan dengan menggunakan algoritma ini, kita bisa menggunakan dua pendekatan berbeda: maju dan mundur. Penyelesaian secara maju atau mundur keduanya ekivalen dan menghasilkan solusi optimum yang sama. Pengalaman menunjukkan bahwa penyelesaian dengan program dinamis mundur umumnya lebih mangkus. Secara umum, ada empat langkah yang dapat dilakukan saat mengembangkan algoritma program dinamis:

1. Karakteristikkan struktur solusi optimal
2. Definisikan secara rekursif nilai solusi optimal
3. Hitung nilai solusi optimal secara maju atau mundur.
4. Konstruksi solusi optimal.

Perlu dicatat bahwa solusi optimal yang dihasilkan oleh program dinamis dapat lebih dari satu buah.

III. ISI

Penggunaan algoritma ini akan mendukung tim untuk memilih strategi terbaik yang mungkin untuk balapan. Algoritma ini akan menentukan jumlah pit stop yang optimal berdasarkan pengukuran, statistik, dan pengetahuan dari ahli-ahli tim. Di dalam algoritma program dinamis terdapat tahap dan setiap tahap memiliki simpu yang menyatakan status. Di dalam penerapannya, tahap ini merupakan beberapa pertimbangan yang ada dalam menentukan strategi, misalnya keadaan dari mobil, keadaan cuaca, keadaan sirkuit, dan lainnya. Sedangkan simpul merupakan status dari tiap tahap tersebut. Misalnya, untuk tahap keadaan mobil (*V*1), dapat dipecah menjadi lebih detil lagi seperti, sisa bahan bakar, keadaan ban, keadaan mesin, dan lainnya. Contohnya adalah sebagai berikut :



Gambar 2 Graf multitahap yang menyatakan tahap dan status dari kasus F1

V_1, V_2, V_3, V_4, V_5, menyatakan pertimbangan-pertimbangan yang ada. Karena pada dasarnya sangat banyak pertimbangan yang ada dalam suatu balapan, maka kita hanya mengambil beberapa pertimbangan sebagai contoh dalam menghitung. Penjelasan dari tahap dan simpul dari gambar diatas adalah sebagai berikut :

- V_1 bernilai 1 untuk setiap sisinya ke simpul-simpul yang ada karena V_1 digunakan hanya sebagai state awal. Sebagai contoh, di kasus ini simpul 1 adalah keadaan mesin 100%.
- V_2 adalah keadaan atau tipe sirkuit.
  - Simpul 2 adalah tipe sirkuit yang mempunyai banyak tikungan.
  - Simpul 3 adalah tipe sirkuit yang mempunyai banyak jalan lurus.
  - Simpul 4 adalah tipe sirkuit yang mempunyai campuran antara tikungan dan jalan lurus.
- V_3 adalah keadaan cuaca
  - Simpul 5 adalah cuaca hujan.
  - Simpul 6 adalah cuaca kering.
- V_4 adalah banyaknya pit-stop yang sebaiknya diambil
  - Simpul 7 adalah pengambilan pit stop sebanyak lebih dari tiga kali.
  - Simpul 8 adalah pengambilan pit stop sebanyak tiga kali.
  - Simpul 9 adalah pengambilan pit stop sebanyak dua kali.
- V_5 adalah state akhir dari pertimbangan. Sama seperti V_1 semua simpul yang menuju simpul V_5 bernilai 1 karena hanya sebagai state akhir.

Penyelesaian dari masalah diatas adalah sebagai berikut:

Relasi rekurens berikut menyatakan lintasan terpendek dari status s ke x_4 pada tahap *k*:

$f_4(s) = c_{sx4}$ (basis)

$f_k(s) = \min\{c_{sxk} + f_{k+1}(x_k)\}$ (rekurens)

Keterangan:

a. x_k : peubah keputusan pada tahap k (k = 1,2,3).
b. c_{sx4} : bobot (*cost*) sisi dari *s* ke x_k
c. $f_k(s,x_k)$: total bobot lintasan dari *s* ke x_k
d. $f_k(s)$: nilai minimum dari $f_k(s,x_k)$.

Tujuan program dinamis mundur: mendapatkan f_1 (1) dengan cara mencari f_4 (s), f_3 (s), f_2 (s) terlebih dahulu.

Tahap 4

$f_4(s) = c_{sx4}$

S	Solusi Optimum	
	$f_4(s)$	x_4*
7	1	10
8	1	10
9	1	10

Catatan: x_k* adalah nilai x_k yang meminimumkan $f_k(s, x_k)$.

Tahap 3

$f_3(s) = \min\{c_{sx3} + f_4(x_3)\}$

x_3	$f_3(s,x_3) = c_{sx3} + f_4(x_3)$			Solusi Optimum	
s	7	8	9	$f_3(s)$	x_3*
5	2	3	4	2	7
6	2	3	4	2	7

Tahap 2

$f_2(s) = \min\{c_{sx2} + f_3(x_2)\}$

x_3	$f_2(s,x_2) = c_{sx2} + f_3(x_2)$		Solusi Optimum	
s	5	6	$f_2(s)$	x_2*
2	3	4	3	6
3	3	4	3	6
4	3	4	3	6

Tahap 1

$f_1(s) = \min\{c_{sx1} + f_2(x_1)\}$

x_3	$f_2(s,x_2) = c_{sx2} + f_3(x_2)$			Solusi Optimum	
s	2	3	4	$f_1(s)$	x_1*
1	4	4	4	4	2 atau 3 atau 4

Solusi optimum dapat dibaca pada tabel di bawah ini:

	x_1	x_2	x_3	x_4	*Panjang Lintasan Terpendek*
1	2	6	7	10	4
1	3	6	7	10	4
1	4	6	7	10	4

Jadi, ada tiga lintasan terpendek dari 1 ke 10, yaitu

1→2→6→7→10
1→3→6→7→10
1→4→6→7→10

Panjang ketiga lintasan tersebut sama, yaitu 4.

Langkah-langkah diatas merupakan cara untuk menghitung biaya terkecil yang akan menjadi pilihan para ahli dari setiap tim. *Pseudocode* dari algoritma program dinamis dalam memecahkan masalah strategi F1 ini adalah

```
   funtion                minWeight(input
node:integer,   set:ListofInteger)   ->
integer
          hasil : ListofInteger
          if   (set   =   tidak   berisi)
//basis
               -> bobot(node,1);

          else
```

```
//rekursif
                  iter        traversal
[0..ukuran(set)]
              for(int
iter=0;iter<set.size();iter++)

                  setTemp             :
ListofInteger

                  copy(setTemp,set)

      remove(setTemp,iter)

hasil.add(bobot(node,set.get(iter))+bo
bot(set.get(iter),setTemp));

              -> min(hasil);    .
```

IV. ANALISIS

Biaya-biaya yang ada pada contoh diatas merupakan permisalan saja. Pada kenyataannya, bobot-bobot diatas nilainya akan didapatkan dari masukan ahli – ahli dari setiap tim. Permasalahan dari membangun sebuah strategi balap F1 adalah tidak ada yang mengetahui dengan pasti strategi dari tim lain dan apa yang akan mereka lakukan. Algoritma ini dapat diterapkan dalam sebuah sistem jika ada pengukuran secara *real-time* yang mengubah data mentah menjadi sebuah informasi yang dapat memudahkan tim untuk membacanya. Sistem *real-time* ini juga membantu sebagai sensor dari berbagai macam keadaan yang akan digunakan sebagai tahap-tahap dari *multistage graph*.

Real Time System (RTS) disebut juga sistem waktu nyata. Sistem ini harus menghasilkan respon yang tepat dalam batas waktu yang telah ditentukan. Jika respon komputer melewati batas waktu waktu tersebut, maka terjadi degradasi performansi atau kegagalan sistem. Sebuah RTS adalah sistem yang kebenarannya secara logis didasarkan pada kebenaran hasil-hasil keluaran sistem dan ketepatan waktu hasil-hasil tersebut dikeluarkan. Berdasarkan batasan waktu yang dimilikinya, RTS dibagi atas :

1. *Hard Real Time*
   Sistem ini dibutuhkan untuk menyelesaikan pekerjaan yang penting dengan jaminan waktu tertentu. Jika kebutuhan waktu tidak terpenuhi, maka aplikasi akan gagal. Dalam definisi lain disebutkan bahwa kontrol sistem *hard real-time* dapat mentoleransi keterlambatan tidak lebih dari 100 mikro detik. Secara umum, sebuah proses di kirim dengan sebuah pernyataan jumlah waktu yang dibutuhkan untuk menyelesaikan atau menjalankan I/O.

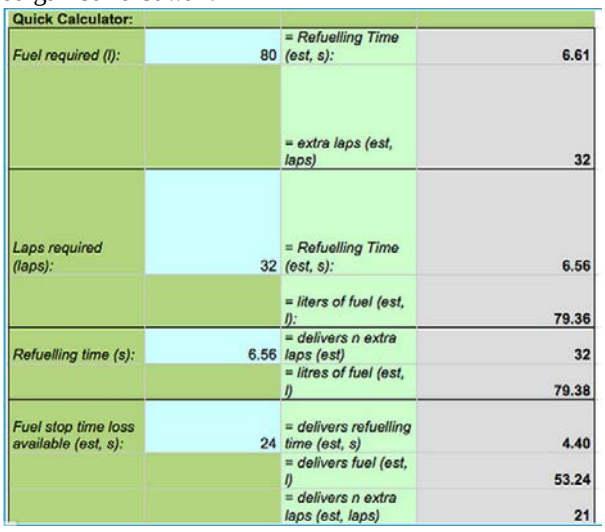
2. *Soft Real Time*
   Komputasi dari *soft real-time* memiliki sedikit kelonggaran. Dalam sistem ini, proses yang kritis menerima prioritas lebih daripada yang lain. Walaupun menambah fungsi *soft real-time* ke sistem *time sharing* mungkin akan mengakibatkan ketidakadilan pembagian sumber daya dan mengakibatkan delay yang lebih lama, atau mungkin menyebabkan *starvation*, hasilnya adalah tujuan secara umum sistem yang dapat mendukung multimedia, grafik berkecepatan tinggi, dan variasi tugas yang tidak dapat diterima di lingkungan yang tidak mendukung komputasi soft real-time.

3. *Interactive Deadline*
   Pada interaktif *real-time*, maka waktu deadlinenya bisa ditawar, artinya tidak secara mutlak pada titik tertentu.

Untuk membantu memberikan masukan kepada algoritma ini, sebaiknya menggunakan sistem *real-time* dengan jenis *Soft Real Time*. Alasannya adalah karena sistem melakukan tugasnya sebagai sensor cukup setiap 2-5 detik.

Sistem yang menggunakan algoritma ini hanya membantu dalam strategi pit stop. Kemenangan dari sebuah tim juga ditentukan oleh kemampuan dari pebalap untuk memenangkan lomba. Selain itu, ketahanan dari mobil juga dapat mempengaruhi keputusan.

Berbagai jenis pengukuran harus dilakukan untuk mengumpulkan data yang dibutuhkan untuk optimasi. Salah satu jenis pengukuran yang mungkin dilakukan ada pada gambar dibawah:



Quick Calculator:			
Fuel required (l):	80	= Refuelling Time (est, s):	6.61
		= extra laps (est, laps)	32
Laps required (laps):	32	= Refuelling Time (est, s):	6.56
		= liters of fuel (est, l):	79.36
Refuelling time (s):	6.56	= delivers n extra laps (est)	32
		= litres of fuel (est, l)	79.38
Fuel stop time loss available (est, s):	24	= delivers refuelling time (est, s)	4.40
		= delivers fuel (est, l)	53.24
		= delivers n extra laps (est, laps)	21

Gambar 3 Salah satu pengukuran terhadap bahan bakar

Karena itu, algoritma ini cocok digunakan dengan bantuan sebuah sistem yang *real-time* untuk mendapatkan masukan yang dibutuhkan oleh algoritma. Algoritma ini dapat membantu sebuah sistem DSS (*Decision Support System).*

DSS adalah bagian dari sebuah sistem informasi berbasis komputer (termasuk sistem berbasis

pengetahuan) yang dipakai untuk pengambilan keputusan dalam suatu organisasi atau perusahaan, dalam kasus ini organisasinya adalah tim F1. DSS dapat digambarkan sebagai sistem yang berkemampuan mendukung analisis ad hoc data, dan pemodelan keputusan, berorientasi keputusan, orientasi perencanaan masa depan, dan digunakan pada saat-saat yang tidak biasa.

Tujuan dari DSS adalah membantu menyelesaikan masalah semi-terstruktur, mendukung pihak tertentu dalam mengambil keputusan, meningkatkan efektifitas bukan efisiensi dari pengambilan keputusan.

V. CONCLUSION

1. Algoritma program dinamis digunakan sebagai salah satu cara untuk memecahkan masalah optimasi.
2. Dalam menentukan strategi pit stop F1 dapat digunakan algoritma program dinamis agar pemilihan jumlah pit stop dan pengaturan mobil tepat.
3. Algoritma ini lebih baik jika mendapatkan inputan dari sensor sebuah sistem *real-time*.
4. Hasil dari algoritma ini dapat digunakan sebagai DSS (*Decision Support System)*.

REFERENSI

[1] Munir, Rinaldi. "Diktat Kuliah IF3051 Strategi Algoritma", Program Studi Teknik Informatika STEI ITB. Bandung. 2007.

[2] http://dwiishartono.wordpress.com/2008/09/17/real-time-systemrts/, diakses 8 Desember 2011, 0:23

[3] http://en.wikipedia.org/wiki/Formula_One, diakses 7 Desember 2011, 20:14

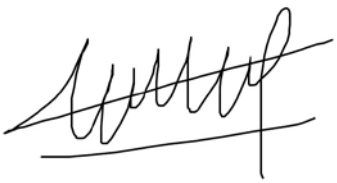
[4] http://id.wikipedia.org/wiki/Formula_Satu/ , diakses 7 Desember 2011, 20:18

[5] http://id.wikipedia.org/wiki/Sistem_pendukung_keputusan, diakses 8 Desember 2011, 1:44

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 8Desember 2011



Marchy Tio Pandapotan
13509026