Dynamic Programming untuk Optimasi Sponsorship Peserta Kompetisi Balap F1

Rindhu Astry Nalistia[1], Arya Widyanarko[2], Ahmad Zamakhsyari Sidiq[3]

Laboratorium Ilmu dan Rekayasa Komputasi
Departemen Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung

E-mail : if14005@students.if.itb.ac.id[1], if14030@students.if.itb.ac.id[2], if14053@students.if.itb.ac.id[3]

Abstrak

Olahraga balap mobil F1 (F1 *Racing*) dapat dikatakan sebagai olahraga "mahal". Tim-tim dalam kompetisi balap mobil F1 pada umumnya membutuhkan biaya yang besar dalam pengadaan, pengelolaan dan perawatan sarana dan prasarana yang terkait dengan tim tersebut. Selayaknya suatu instansi olahraga yang membutuhkan dana, masing-masing tim F1 tersebut harus dapat mengatur strategi untuk menyeleksi dan menentukan kombinasi *sponsorship*, untuk mendapatkan keuntungan semaksimal mungkin dari sponsor-sponsor yang tersedia. Tiap *sponsorship* yang telah terpilih untuk bekerjasama dengan tim berhak untuk ditampilkan logo atau *icon* sponsor tersebut pada mobil balap tim, dimana mobil balap tersebut hanya memiliki slot kosong dengan jumlah yang terbatas untuk diisi dengan logo sponsor. Tiap sponsor terdiri dari bidang usaha yang berbeda-beda dan tawaran keuntungan yang berbeda-beda. Permasalahan ini dapat disimulasikan sebagai salah satu wujud lain dari permasalah integer knapsack. Permasalahan optimasi seperti ini paling sesuai apabila diselesaikan dengan algoritma yang mencari solusi optimum secara *step by step,* dimana kelompok kami memutuskan untuk melakukan penelitian terhadap optimasi *sponsorship* pada kompetisi balap F1 dengan menerapkan algoritma *dynamic programming.*

***Kata kunci**: dynamic programming,optimasi,strategi, integer knapsack*

1. Pendahuluan

Dalam dunia bisnis, setiap petarung bisnis pasti mengorientasikan bisnisnya untuk mendapatkan keuntungan semaksimal mungkin dengan pengorbanan seminimal mungkin. Salah satu cara untuk mendapatkan keuntungan semaksimal mungkin adalah dengan melakukan optimasi dalam pemilihan *sponsorship*. Oleh karena itu, tiap instansi bisnis dituntut untuk memiliki suatu strategi optimasi untuk menentukan kombinasi sponsorship yang mendatangkan keuntungan yang paling maksimal. Sebagai algoritma yang memecahkan masalah dengan menguraikan solusi menjadi sekumpulan langkah, Pemrograman Dinamis (*Dynamic Programming*) merupakan algoritma yang tepat untuk menyelesaikan permasalahan optimasi tersebut.

2. Ruang Lingkup

Ruang lingkup makalah ini meliputi penentuan kombinasi *sponsorship* dengan keuntungan paling maksimal, menggunakan algoritma Pemrograman Dinamis *(Dynamic Programming)*

3. Integer Knapsack

Persoalan Integer Knapsack merupakan salah satu persoalan klasik, yaitu masalah yang banyak ditemukan dalam literatur-literatur lama. Namun hingga kini persoalan ini banyak sekali ditemukan dalam permasalahan sehari-hari, misalnya memilih barang apa saja yang akan dimasukkan ke dalam koper untuk dibawa bepergian, sampai pada masalah yang akan kita bahas, optimasi sponsorship pada kompetisi F1.

Dalam persoalan ini, kita diberikan n buah objek yang masing-masing memiliki nilai bobot dan keuntungan. Kita diminta untuk memilih objek-objek yang akan dimasukkan ke dalam *Knapsack* (karung) yang memiliki bobot maksimum W sehingga didapat keuntungan yang maksimum.

3.1 Skema Umum Permasalahan Integer Knapsack

Diberikan n buah objek dengan bobot masing-masing $w_1, w_2, ..., w_n$ dan keuntungan $p_1, p_2, ..., p_n$.
Lalu terdapat sebuah knapsack dengan bobot maksimum K. Solusi dari persoalan diatas dinyatakan dalam vektor n-tupel:

$X = \{x_1, x_2, ..., x_n\}$

dimana x_i bernilai 1 jika objek ke-i dipilih dan bernilai 0 jika objek ke-i tidak dipilih.

Misalnya X = {1, 0, 0} merupakan solusi dimana objek yang dipilih ialah objek ke-1, sedangkan objek ke-2 dan ke-3 tidak dipilih.[1]

4. *Dynamic Programming*

4.1 Definisi *Dynamic Programming*

Dynamic Programming (Program Dinamis) 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[1].

Dynamic Programming biasanya digunakan untuk memecahkan masalah optimasi, yakni masalah yang memiliki banyak kemungkinan solusi, dan *Dynamic Programming* diminta untuk mencari solusi yang paling optimal.

4.2 Jenis-jenis *Dynamic Programming*

Dalam menyelesaikan persoalan dengan *Dynamic Programming*, kita dapat menggunakan dua pendekatan berbeda ,
Misalkan x_1, x_2, …, x_n menyatakan peubah (*variable*) keputusan yang harus dibuat untuk masing-masing tahap 1, 2, ..., n. Maka :

a. *Dynamic Programming Forward,* (Program Dinamis Maju) program dinamis bergerak mulai dari tahap 1, ke tahap 2, 3 dan sterusnya sampai tahap *n*. Runtunan *variable* keputusan adalah x_1, x_2, …, x_n.

b. *Dynamic Programming Backward,* (Program Dinamis Mundur) program dinamis bergerak dari tahap ke *n,* terus mundur ke tahap *n -1,* tahap *n-2* sampai tahap 1.

Pada Permasalahan ini kita akan menggunakan Program Dinamis Maju.

5. Penerapan Algoritma *Dynamic Programming*

Tabel Calon Sponsor

Code	*Company Name*	*Kind*	*Requested Slot (Rs)*	*Profit (p)*
A	Allianz	Insurance	3	$2,000
B	Lucky Strike	Cigarrete	5	$3,000
C	Siemens	Handphone	2	$1,200
D	Mild Seven	Cigarrete	4	$2,600
E	HSBC	Bank	2	$1,500
F	Vodafone	Handphone	3	$2,300
G	Marlboro	Cigarrete	3	$2,400

Kapasitas Maksimum Slot (MaxSlot) = 10

Ket : Requested Slot adalah jumlah slot stiker yang diminta perusahaan untuk dilekatkan stiker berlogo perusahaan tersebut.

Solusi permasalahan ini ditemukan dengan menggunakan *Dynamic Programming* maju.

1. Langkah (n) adalah proses memasukkan perusahaan ke dalam kombinasi *sponsorship* (ada 7 tahap, tiap tahap memasukkan perusahaan ke-n)
2. Status (y) menyatakan kapasitas slot yang tersisa setelah memasukkan perusahaan pada tahap sebelumnya

Ketika memasukkan perusahaan pada langkah ke *n*, kapasitas slot *sticker* setelah itu adalah y- Rs_n (yaitu $f_{n-1}(y-Rs_k)$. Untuk mengisi slot lain yang belum terisi, dapat diterapkan prinsip optimalitas terhadap permasalahan tersebut dengan menjadikan nilai optimum sebelumnya sebagai acuan untuk kapasitas slot yang masih belum terisi. Berikutnya, keuntungan (*profit)* yang ditawarkan perusahaan pada langkah *n* (P_n) dijumlah dengan f_{n-1} (y – Rs_n) dibandingkan dengan keuntungan pemasukkan perusahaan hanya n-1 perusahaan, $f_{n-1}(y)$. Jika $P_n + f_{n-1}(y - Rs_n)$ lebih kecil dari $f_{n-1}(y)$, maka perusahaan ke-n tidak dimasukkan kedalam kombinasi *sponsorship* , apabila sebaliknya (lebih besar) maka perusahaan ke-n dimasukkan kedalam kombinasi *sponsorship.* Selain itu, apabila didalam kombinasi *sponsorship* terdapat lebih dari 1 perusahaan yang berjenis (bidang usaha) sama, maka kombinasi tersebut ditolak walaupun keuntungan yang ditawarkan lebih besar dari keuntungan optimum pada $f_{n-1}(y)$, pada kasus ini, keuntungan optimum pada langkah sebelumnya ($f_{n-1}(y)$) –lah yang terpilih menjadi keuntungan optimum pada $f_n(y)$.

Relasi rekurens untuk persoalan ini adalah

$f0(y) = 0 \quad y = 0,1,2,\ldots,MaxSlot$ (basis)
$fn(y) = -\infty \quad y<0$ (basis)
$fn(y) = \max\{f_{n-1}(y), Pn + f_{n-1}(y-Rs_k)\}, n = 1,2,\ldots,x$ (rekurens)

Pada hal ini, fn(y) adalah keuntungan optimum dari persoalan optimasi *sponsorship* pada tahap n untuk kapasitas slot sticker sebesar y.

Tiap-tiap langkah pencari solusi diwakilkan dengan masing-masing satu tabel,

Keterangan Tabel :
y : kapasitas slot stiker
$f_n(y)$: keuntungan optimum pada langkah ke-n
OK : variabel yang menyatakan *acceptance (ditandai dengan √ jika tidak ada kombinasi perusahaan dengan jenis (bidang usaha) yang sama)*
Solusi : Kombinasi perusahaan dengan keuntungan yang ditawarkan paling besar dan memenuhi syarat *acceptance* .

Langkah 1

$f_1(y) = \max\{f_0(y), P_1 + f_0(y\text{-}Rs_1)\}$

				Solusi Optimum	
y	$f_0(y)$	$2000+f_0(y\text{-}3)$	OK	$f_1(y)$	Solusi
0	**0**	-∞	√	0	-
1	**0**	-∞	√	0	-
2	**0**	-∞	√	0	-
3	0	**$2,000**	√	$2,000	A
4	0	**$2,000**	√	$2,000	A
5	0	**$2,000**	√	$2,000	A
6	0	**$2,000**	√	$2,000	A
7	0	**$2,000**	√	$2,000	A
8	0	**$2,000**	√	$2,000	A
9	0	**$2,000**	√	$2,000	A
10	0	**$2,000**	√	$2,000	A

Langkah 2

$f_2(y) = \max\{f_1(y), P_2 + f_1(y\text{-}Rs_2)\}$

				Solusi Optimum	
y	$f_1(y)$	$3000+f_1(y\text{-}5)$	OK	$f_2(y)$	Solusi
0	**0**	-∞	√	0	-
1	**0**	-∞	√	0	-
2	**0**	-∞	√	0	-
3	**$2,000**	-∞	√	$2,000	A
4	**$2,000**	-∞	√	$2,000	A
5	$2,000	**$3,000**	√	$3,000	B
6	$2,000	**$3,000**	√	$3,000	B
7	$2,000	**$3,000**	√	$3,000	B
8	$2,000	**$5,000**	√	$5,000	A, B
9	$2,000	**$5,000**	√	$5,000	A, B
10	$2,000	**$5,000**	√	$5,000	A, B

Langkah 3

$f_3(y) = \max\{f_2(y), P_3 + f_2(y\text{-}Rs_3)\}$

				Solusi Optimum	
y	$f_2(y)$	$1200+f_2(y\text{-}2)$	OK	$f_3(y)$	Solusi
0	**0**	-∞	√	0	-
1	**0**	-∞	√	0	-
2	0	**$1,200**	√	$1,200	C
3	**$2,000**	$1,200	√	$2,000	A
4	**$2,000**	$1,200	√	$2,000	A
5	$3,000	**$3,200**	√	$3,200	A, C
6	$3,000	**$3,200**	√	$3,200	A, C
7	$3,000	**$4,200**	√	$4,200	B, C
8	**$5,000**	$4,200	√	$5,000	A, B
9	**$5,000**	$4,200	√	$5,000	A, B
10	$5,000	**$6,200**	√	$6,200	A, B, C

Langkah 4

$f_4(y) = \max\{f_3(y), P_4 + f_3(y\text{-}Rs_4)\}$

				Solusi Optimum	
y	$f_3(y)$	$2600+f_3(y\text{-}4)$	OK	$f_4(y)$	Solusi
0	**0**	-∞	√	0	-
1	**0**	-∞	√	0	-
2	**$1,200**	-∞	√	$1,200	C
3	**$2,000**	-∞	√	$2,000	A
4	$2,000	**$2,600**	√	$2,600	D
5	**$3,200**	$2,600	√	$3,200	A, C
6	$3,200	**$3,800**	√	$3,800	C, D
7	$4,200	**$4,600**	√	$4,600	A, D
8	**$5,000**	$4,600	√	$5,000	A, B
9	$5,000	**$5,800**	√	$5,800	A, C, D
10	**$6,200**	$5,800	√	$6,200	A, B, C

Langkah 5

$f_5(y) = \max\{f_4(y), P_5 + f_4(y\text{-}Rs_5)\}$

				Solusi Optimum	
y	$f_4(y)$	$1500+f_4(y\text{-}2)$	OK	$f_5(y)$	Solusi
0	**0**	-∞	√	0	-
1	**0**	-∞	√	0	-
2	$1,200	**$1,500**	√	$1,500	E
3	**$2,000**	$1,500	√	$2,000	A
4	$2,600	**$2,700**	√	$2,700	C,E
5	$3,200	**$3,500**	√	$3,500	A,E
6	$3,800	**$4,100**	√	$4,100	D,E
7	$4,600	**$4,700**	√	$4,700	A,C,E
8	$5,000	**$5,300**	√	$5,300	C,D,E
9	$5,800	**$6,100**	√	$6,100	A,D,E
10	$6,200	**$6,500**	√	$6,500	A,B,E

Langkah 6

$f_6(y) = \max\{f_5(y), P_6 + f_5(y\text{-}Rs_6)\}$

				Solusi Optimum	
y	$f_5(y)$	$2300+f_5(y\text{-}3)$	OK	$f_6(y)$	Solusi
0	**0**	-∞	√	0	-
1	**0**	-∞	√	0	-
2	**$1,500**	-∞	√	$1,500	E
3	$2,000	**$2,300**	√	$2,300	F
4	**$2,700**	$2,300	√	$2,700	C,E

5	$3,500	**$3,800**	√	$3,800	E,F
6	$4,100	**$4,300**	√	$4,300	A,F
7	**$4,700**	$5,000	-	$4,700	A,C,E
8	$5,300	**$5,800**	√	$5,800	A,E,F
9	$6,100	**$6,400**	√	$6,400	D,E,F
10	**$6,500**	$7,000	-	$6,500	A,B,E

Langkah 7
$f_7(y) = \max\{f_6(y), P_7 + f_6(y\text{-}Rs_7)\}$

				Solusi Optimum	
y	$f_6(y)$	$2400+f_6(y-3)$	OK	$f_7(y)$	Solusi
0	**0**	-∞	√	0	-
1	**0**	-∞	√	0	-
2	**$1,500**	-∞	√	$1,500	E
3	$2,300	**$2,400**	√	$2,400	G
4	**$2,700**	$2,400	√	$2,700	C,E
5	$3,800	**$3,900**	√	$3,900	E,G
6	$4,300	**$4,700**	√	$4,700	F,G
7	$4,700	**$5,100**	√	$5,100	C,E,G
8	$5,800	**$6,200**	√	$6,200	E,F,G
9	$6,400	**$6,700**	√	$6,700	A,F,G
10	$6,500	**$7,100**	√	$7,100	A,C,E,G

Solusi optimum dari persoalan optimasi sponsorship diatas adalah :
(A,C,E,G) atau **(Allianz, Siemens, HSBC dan Marlboro)** dengan jumlah keuntungan = **$7,100.**

6. Kesimpulan

Hasil penerapan algoritma *Dynamic Programming* diatas menunjukan bahwa kita dapat melakukan optimasi *sponsorship* sehingga bisa memberikan keuntungan *financial* yang lebih besar (optimal).
Permasalahan optimasi sponsorship ini juga merupakan modifikasi dari permasalahan Integer Knapsack namun dengan tambahan constrain (batasan) tidak boleh ada sponsor yang bergerak dibidang usaha yang sama.

Daftar Pustaka

1. Munir, Rinaldi. *Diktat Kuliah IF2251 Strategi Algoritmik*, Penerbit PRODI Informatika, ITB, Bandung, 2005.
2. *http://www.macs.hw.ac.uk/~alison/ds98/node122.html*