Pencarian Transaksi Terbaik pada Bitcoin dengan Program Dinamis

Joseph Salimin 135160371
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
joseph_salimin@hotmail.com

***Abstrak*—Bitcoin merupakan salah satu mata uang kripto yang sedang ramai dibicarakan saat ini. Cara kerja bitcoin yaitu dengan melakukan validasi transaksi yang dilakukan secara desentralisasi, yaitu melalui metode atau protokol yang disebut Proof-of-Work. Protokol Proof-of-Work sendiri merupakan metode yang memakai kerja komputer secara terus menerus untuk menghasilkan suatu blok yang menerima dan memvalidasi sejumlah transaksi dengan ukuran tertentu dan tentunya terdapat juga biaya transaksi yang diterima. Biaya transaksi itu kemudian bersama dengan upah dari kerja keras untuk menghasilkan blok yang akan diletakkan ke dalam rantai blok kemudian diberikan kepada sekumpulan orang yang disebut sebagai miner. Proses ini kemudian disebut dengan yang namanya mining. Tentunya miner ingin mencari transaksi dengan ukuran terkecil dan menghasilkan biaya yang besar. Salah satu cara yang telah diimplementasikan yaitu dengan metode Branch and Bound. Namun cara lain untuk mendapatkan transaksi terbaik dari sekumpulan transaksi yang ada yaitu dengan algoritma program dinamis. Program dinamis merupakan metode pemecahan masalah yang menguraikan solusi menjadi sekumpulan langkah. Dengan kata lain, program dinamis merupakan metode yang mencari solusi yang kompleks dengan memecahkannya menjadi beberapa masalah yang lebih kecil**

Kata Kunci—Bitcoin; Proof-of-Work, Program Dinamis; Mining; Miner

I. PENDAHULUAN

Akhir-akhir ini, teknologi blockchain mulai dikenal dan berkembang dengan pesat berkat kenaikan harga Bitcoin dan mata uang kripto lainnya dengan pesat. Teknologi blockchain ini sendiri pun sebenarnya sudah ada sejak tahun 1991 namun kemudian dikenalkan kepada dunia dalam bentuk Bitcoin. Bitcoin itu sendiri adalah implementasi dari blockchain yang dikenalkan oleh penemu Satoshi Nakamoto melalui *paper* yang dibuatnya [1] sebagai sejarah pertama mata uang kripto muncul dan beredar seperti sekarang ini. Bitcoin adalah mata uang kripto yang terdesentralisasi dan berjalan di atas jaringan Bitcoin dan Bitcoin yang dijual belikan maupun yang digunakan sebagai transaksi merupakan aset atau *token* yang digunakan pada jaringan Bitcoin. Sistem yang berjalan pada Bitcoin merupakan sistem yang terdesentralisasi begitu rupa sehingga transaksi dapat dikonfirmasi tanpa orang penengah seperti bank, namun melalui pembentukan suatu blok melalui komputasi yang membutuhkan kerja keras dan persetujuan oleh setiap pihak pada jaringan Bitcoin.

Gambar 1 Logo Bitcoin
Sumber : https://en.wikipedia.org/wiki/Bitcoin

Metode yang dikenalkan melalui Bitcoin dipercaya sebagai salah satu metode yang revolusioner. Bitcoin berhasil mengimplementasikan visi untuk membuat suatu transaksi digital yang dilakukan antar individu tanpa menggunakan instituti finansial. Metode yang dikenalkan oleh Satoshi Nakamoto dalam *paper* yang dibuatnya yaitu protokol Proof-Of-Work. Melalui protokol ini, setiap transaksi akan dipetakan ke dalam suatu *hash* dan kemudian diletakan ke dalam suatu blok untuk selanjutnya blok itu sendiri diletakan pada rantai utama yang ditentukan oleh panjang rantai pada jaringan. Proses pembentukan blok sendiri membutuhkan komputasi yang sangat sulit dan membutuhkan kerja keras dari kumpulan node. Dengan demikian, dapat dibuat suatu rantai blok yang secara teori tidak dapat diubah tanpa membalik rantai blok yang tentunya membutuhkan biaya yang sangat mahal.

Melalui protokol ini, dikenal pula istilah yang disebut dengan *Miner* dan *Mining*. *Miner* adalah node yang melakukan komputasi untuk menghasilkan suatu blok dan kemudian mendapatkan upah Bitcoin. *Mining* adalah proses menambahkan transaksi ke dalam suatu blok dengan melakukan perhitungan komputasi secara terus menerus dengan tujuan mendapatkan upah Bitcoin. Dengan demikian, melalui proses penambangan ini, setiap penambang yang berhasil menambahkan suatu blok ke dalam rantai blok utama akan diberi upah Bitcoin.

Dengan meningkatnya harga Bitcoin akhir-akhir ini, semakin banyak orang yang ingin menjadi penambang Bitcoin sehingga tentunya terjadi persaingan antara satu kumpulan penambang dengan penambang lainnya. Selain itu, transaksi yang terjadi di Bitcoin pun semakin banyak. Padahal ukuran blok Bitcoin terbatas dan komputasi semakin sulit. Dengan demikian untung yang didapat dari menambang Bitcoin semakin kecil.

Oleh karena itu, diperlukan suatu cara untuk setidaknya memperoleh keuntungan yang lebih, yaitu salah satunya dengan mengefisiensikan pengambilan transaksi yang berukuran kecil namun mempunyai biaya transaksi yang mahal. Salah satu cara

yang dapat diimplementasikan yaitu dengan menggunakan metode *Greedy* atau *Branch and Bound*. Namun melalui makalah ini akan dicoba suatu cara lain yaitu dengan menggunakan metode program dinamis untuk mendapatkan transaksi dengan biaya yang tinggi namun mempunyai ukuran yang kecil.

II. DASAR TEORI

A. Program Dinamis

A.1. Pengertian Program Dinamis

Program dinamis atau yang dalam bahasa Inggris disebut sebagai *dynamic programming* meruapakan suatu metode pemecahan masalah dengan menguraikan solusi menjadi sekumpulan langkah atau tahapan sedemikian sehingga dapat dipandang sebagai serangkaian keputusan yang saling berkaitan [2]. Dengan demikian, metode program dinamis ini menguraikan sekumpulan masalah ke dalam masalah-masalah yang lebih sederhana dan menyimpan setiap solusi pada masalah-masalah yang telah diuraikan sehingga tidak perlu dilakukan komputasi yang berulang-ulang.

Metode ini sendiri didasarkan oleh Prinsip Optimalitas yang dikemukan oleh Bellman. Prinsip ini mengatakan bahwa setiap solusi optimal mempunyai properti yang mengatakan bahwa apapun status awal dan pilihan awal yang diambil, setiap pilihan setelahnya harus merupakan solusi optimal yang dipilih berdasarkan pilihan-pilihan sebelumnya. Prinsip ini pada intinya mengatakan bahwa untuk solusi total yang optimal, maka bagian solusi sampai tahap ke-*n* juga optimal. Dengan membuat solusi bagian yang optimal, maka akan didapatkan solusi total yang lebih baik.

Dengan demikian, dapat dilihat bahwa setiap persoalan yang diselesaikan oleh program dinamis mempunyai tiga ciri sebagai berikut, yaitu :

- Terdapat sejumlah berhingga pilihan yang mungkin.
- Solusi pada setiap tahap dibangun dari hasil solusi tahap sebelumnya.
- Penggunaan persyaratan optimasi dan kendala untuk membatasi sejumlah pilihan yang harus dipertimbangkan pada suatu tahap.

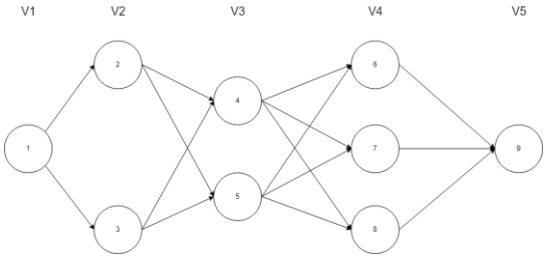
Ciri yang dimiliki oleh program dinamis ini mirip dengan algoritma *greedy* karena algoritma *greedy* juga membentuk solusi secara bertahap. Namun pada metode *greedy* tidak menggunakan prinsip optimalitas karena pada metode ini hanya menghasilkan satu rangkaian keputusan.

Program dinamis juga dapat dikatakan sebagai pengembangan dari *exhaustive search*. Pada program dinamis, pengenumerasian rangkaian keputusan yang tidak mengarah kepada solusi optimal dapat dihilangkan sehingga hanya rangakaian keputusan yang optimal, yaitu rangkaian keputusan yang memenuhi prinsip optimalitas yang akan dihasilkan. Dengan demikian, terdapat jaminan bahwa pengambilan keputusan pada suatu tahap merupakan keputusan yang benar untuk tahap-tahap selanjutnya. Karena itu, program dinamis merupakan algoritma yang lebih berkembang dan biasanya dilakukan untuk mengoptimisasi suatu permasalahan yang mempunyai beberapa solusi alternatif dan mempunyai fungsi pencarian harga.

A.2. Karakterisik Persoalan dan Langkah Program Dinamis

Progran dinamis dapat diterapkan pada beberapa persoalan dengan karakteristik sebagai berikut [2] :

1) Persoalan dapat dibagi menjadi beberapa tahap dan pada setiap tahap diambil satu keputusan.
2) Masing-masing tahap terdiri dari sejumlah status yang berhubungan dengan tahap tersebut. Status merupakan bermacam kemungkinan masukan yang ada dan dapat berhingga atau bahkan tidak berhingga.
3) Keputusan yang diambil pada setiap tahap diteruskan dari status yang bersangkutan ke status berikutnya pada tahap berikutnya.
4) Biaya pada suatu tahap meningkat secara teratur dengna bertambahanya jumlah tahapan
5) Biaya pada suatu tahap bergantung pada biaya tahap-tahap sebelumnya yang sudah berjalan.
6) Keputusan terbaik pada suatu tahap bersifat independen terhadap keputusan yang telah diambil sebelumnya.
7) Terdapat hubungan rekursif yang mengidentifikasikan keputusan terbaik untuk setiap status pada tahap *i* memberikan keputusan terbaik pada tahap *i* + 1.
8) Dengan demikian prinsip optimalitas berlaku.



Gambar 2 Graf dengan tahap dan status

Dalam pengembangan program dinamis, terdapat empat langkah yang dapat diikuti [3]:

1) Cari karakteristik stuktur dari solusi optimal.
2) Mendefinisikan secara rekursif suatu nilai dari solusi optimal.
3) Hitung nilai solusi optimal, baik secara maju ataupun mundur.
4) Buat solusi optimal dari informasi yang sudah dihitung.

Dalam menyelesaikan persoalan dengan program dinamis, terdapat dua pendekatan yang berbeda, yaitu pendekatan secara maju atau pendekatan secara mundur. Misalkan terdapat $x_1, x_2, x_3, \ldots, x_n$ yang menyatakan variabel keputusan yang harus dibuat untuk masing-masing tahap, maka [2] :

- Program dinamis maju akan bergerak mulai dari tahap 1, lalu ke tahap 2, 3, dan seterusnya sampai tahap *n*.
- Program dinamis mundur akan bergerak mulai dari tahap *n*, lalu *n* - 1, dan seterusnya sampai tahap 1.

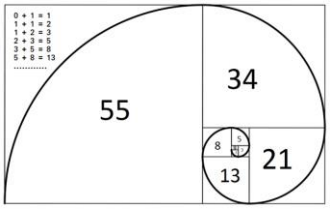
Penyelesaian dengan kedua pendekatan sebenarnya menghasilan solusi optimal yang ekivalen dan sama. Namun berdasakarn pengalaman menunjukkan bahwa penyelesaian dengan program dinamis mundur ditemukan lebih mangkus.

A.3. Contoh Fibonacci

Bilangan Fibonacci merupakan urutan angka-angka dengan karakteristik untuk angkat selain angka pertama dan kedua, nilai angka didefinisikan sebagai penjumlah angka sebelum dan kedua sebelumnya. Dapat dilihat bahwa untuk membuat urutan angka Fibonacci dapat dilakukan dengan penggunaan algoritma program dinamis.

Program dinamis yang diimplementasikan yang dengan menyimpan angka-angka sebelumnya yang sudah didefinisikan terlebih dahulu. Caranya yaitu sebagai berikut :

1) Inisialisasi nilai awal, misalkan F[0] = 0 dan F[1] = 1
2) Untuk setiap indeks ke-*i* yang dimulai dari indeks kedua, hitung nilai F[i] yaitu F[i] = F[i-1] + F[i-2]
3) Lakukan dengan cara yang sama sampai indeks ke-*n*.



Gambar 3 Bilangan Fibonacci
Sumber : https://www.geeksforgeeks.org/program-for-nth-fibonacci-number/

B. Bitcoin

B.1. Pengertian Bitcoin

Bitcoin merupakan mata uang kripto yang terdistribusi dan dapat dipindahkan secara cepat dan aman antara dua individu yang berbeda dimanapun dan kapanpun. Bitcoin itu sendiri merupakan mata uang yang berjalan pada jaringan Bitcoin dan biasanya disebut dengan singkatan "BTC". Bitcoin merupakan implementasi yang pertama kali berhasil untuk membentuk mata uang yang terdistribusi pada sistem yang dikenal sebagai blockchain. Bitcoin diciptakan oleh Satoshi Nakamoto yang sampai saat ini keberadaannya tidak diketahui orang. Nilai terkecil pada bitcoin disebut sebagai satoshi, dimana satu satoshi bernilai 0.000000001 BTC.

Untuk menyimpan Bitcoin, diperlukan suatu dompet digital. Dompet digital pada Bitcoin merupakan koleksi dari kunci pribadi yang tertutup dan tidak boleh diketahui oleh siapapun. Jumlah bitcoin yang dimiliki dapat dilihat dengan melakukan iterasi transaksi yang telah dilakukan oleh kumpulan kunci yang ada. Kunci pribadi ini mempunyai pasangan dengan kunci terbuka yang dapat diketahui oleh orang dan kemudian melalui kunci yang terbuka ini, dapat diubah menjadi sebuah alamat Bitcoin yang terbuka dan dapat dilihat orang. Untuk menjaga keamanan dan privasi, sebaiknya dibuat alamat Bitcoin yang baru ketika melakukan transaksi.

Setiap blok pada Bitcoin mempunyai struktur data dengan ukuran maksimal 1 MB. Untuk masing-masing blok terdiri dari suatu *magic number*, ukuran block, versi dari blok tersebut, *hash* dari blok sebelumnya, *merkle root*, *timestamp*, kesulitan, nonce, dan suatu *header* dari blok. *Header* terdiri dari jumlah transaksi dan kumpulan transaksi. Dengan demikian, semakin banyak transaksi maka ukuran akan semakin besar sampai maksimal 1 MB. Penentuan rantai blok utama yang dilihat yaitu dengan melihat rantai dengan urutan terpanjang dan dengan tingkat kesulitan yang tertinggi. [4]

B.2. Proof-of-Work

Pada jaringan Bitcoin, seperti yang sudah diketahui bahwa transaksi yang ada diolah secara terdesentralisasi oleh kumpulan node sebagai penambang yang melakukan kerja keras pada kalkulasi yang mahal untuk menambahkan blok ke rantai. Proses ini disebut sebagai *mining* atau penambangan pada suatu konsep yang berjalan pada jaringan Bitcoin yang disebut sebagai Proof-of-Work, dimana konsep ini dinamakan demikian karena validasi transaksi dilakukan komputer dengan kerja keras tanpa berhenti.

Penambangan mempunyai beberapa tujuan, yaitu untuk melakukan verifikasi terhadap transaksi dan untuk menghidari permasalahan yang dinamakan *double spending*, yaitu suatu masalah dimana suatu koin digital yang sama dipakai lebih dari satu transaksi. Misalkan suatu individu yang hanya mempunyai 5 BTC mengirimkan 5 BTC ke individu yang lain dan pada saat yang bersamaan mengirimkan 1 BTC ke individu yang lain. Selain itu penambangan mempunyai tujuan lain yaitu untuk membuat suatu mata uang digital dengan memberikan upah kepada para penambang. Upah pada jaringan Bitcoin yaitu Bitcoin itu sendiri. Dengan demikian, penambangan Bitcoin turun membantu menjadi Bitcoin sebagai mata uang digital. Ibarat seperti menambang emas.

Proses pembentukan blok pada protokol Proof-of-Work yaitu sebagai berikut :

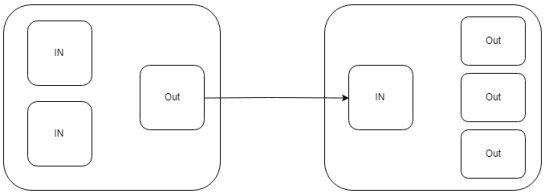
1) Saat transaksi dilakukan, transaksi akan dimasukkan ke suatu kumpulan transaksi untuk diproses.
2) Transaksi kemudian dikumpulkan ke dalam suatu blok
3) Penambangan memverifikasi setiap transaksi pada blok sesuai.
4) Proses verifikasi suatu blok dilakukan dengan melakukan komputasi yang berat dan mahal seperti memecahan suatu puzzle.
5) Blok dibuat jika berhasil memecahkan suatu masalah
6) Setelah suatu blok berhasil dibuat dengan memecahkan masalah, maka akan dikirimkan pesan kepada setiap node pada jaringan dan akan dicek apakah setiap transaksi yang ada adalah valid. Jika semua telah valid, maka blok yang dibuat akan ditambahkan ke rantai utama pada masing-masing node.

B.3. Transaksi Bitcoin

Transaksi pada Bitcoin merupakan transaksi yang aman, bahkan lebih aman pada transaksi yang dilakukan oleh penengah. Pada sistem yang tersentralisasi, setiap penyerang dapat menyerang ke sistem utama yang mengatur data transaksi. Namun pada Bitcoin, perlu biaya yang sangat mahal karena harus menyerang setiap node pada jaringan dan tentunya hal ini secara teori hampir mustahil untuk dilakukan. Selain itu, transaksi yang dilakukan juga memerlukan verifikasi melalui penggunaan kriptografi.

Transaksi pada Bitcoin pada umumnya mempunyai referensi ke beberapa keluaran transaksi sebelumnya sebagai masukan transaksi saat ini dan mejumlahkan semua nilai Bitcoin yang ada dan menghasilkan suatu keluaran transaksi yang baru. Masukan

untuk transaksi terdiri dari total keluaran transaksi sebelumnya, sebuah indeks, dan sebuah *script* yang terdiri dari tanda tangan digital dan sebuah kunci publik yang digunakan untuk memverifikasi hasil keluaran. Selanjutnya, keluaran pada transaksi mempunyai instruksi untuk mengirimkan bitcoin ke beberapa alamat. Agar transaksi dapat diverifikasi, maka kunci public harus sesuai dengan tanda tangan digital yang dibuat berdasarkan kunci privat yang koresponden dengan kunci publik.



Gambar 4 Transaksi Bitcoin

Ukuran dari suatu transaksi dipengaruhi oleh banyaknya masukan dan keluaran dari transaksi tersebut. Semakin banyak masukan ke transaksi dan keluaran yang hasil, maka ukuran transaksi akan semakin besar. Adapun perhitungan ukuran transaksi secara sederhana yaitu sebagai berikut :

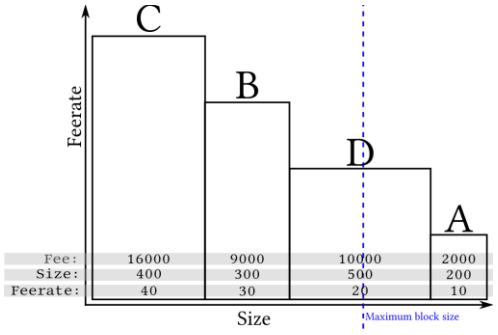
$$Total\ size = int * 148 + out * 34 + 10 \pm in \quad (1)$$

Untuk biaya transaksi, sulit dilakukan perhtiungan karena bergantung pada ukuran dan faktor-faktor lainnya yang sudah ditetapkan oleh sistem dan aplikasi dompet Bitcoin yang ada.

B.4. Pemilihan Transaksi Bitcoin

Setiap transaksi yang dibuat tidak langsung dikonfirmasi oleh jaringan Bitcoin, namun diletakkan pada suatu tempat penampungan yang menerima transaksi yang *pending*. Transaksi-transaksi tersebut akan dipilih oleh penambang untuk dimasukkan ke dalam suatu blok dan kemudian melakukan verifikasi transaksi-transaksi dan menyelesaikan masalah komputasi pada blok.

Karena penambang membutuhkan biaya yang besar sehingga tentunya yang menjadi prioritas utama bagi penambang ialah mendapatkan upah yang besar untuk setiap blok yang berhasil ditambahkan. Perlu diingat juga bahwa ukuran blok pada Bitcoin terbatas sehingga membutuhkan optimisasi bagi penambang untuk mendapatkan transaksi dengan biaya rata-rata terbaik. Biaya rata-rata yaitu besarnya biaya transaksi dibagi dengan ukuran transaksi tersebut.



Gambar 5 Grafik ukuran dengan rata-rata biaya
Sumber : https://en.bitcoin.it/wiki/Transaction_fees

Jika dilihat pada gambar di atas, tentu jika hanya melihat pada rata-rata biaya saja mengakibatkan ketidakefisien dan tidak dapat dilakukan optimisasi, sehingga perlu suatu algoritma untuk mendapatkan biaya terbesar sebaik mungkin. Salah satunya yaitu dengan menggunakan algoritma program dinamis. Perlu diingat juga bahwa pada makalah ini hanya diberikan contoh yang sederhana. Sebab pada kenyataannya jika hanya menggunakan program dinamis saja tanpa mementingkan prioritas urutan sebelumnya juga menyebabkan suatu transaksi tidak dapat dikonfirmasi, padahal seharusnya setiap transaksi harus dapat dikonfirmasi meskipun membutuhkan waktu yang cukup lama sehingga pertimbangan berikutnya yaitu dengan mempertimbangkan lama transaksi tersebut berada.

III. Rumusan Masalah Program Dinamis

Karena itu, dapat dirumuskan persoalan dimana harus melakukan optimasi dari transaksi-transaksi yang ada untuk dimasukkan ke dalam blok yaitu dengan menggunakan metode program dinamis. Dalam kenyataannya, setiap blok pada Bitcoin mampun menyimpan ratusan bahkan ribuan transaksi dan terdapat ribuan transaksi yang menunggu untuk dikonfirmasi. Oleh karena itu, pada makalah ini akan ditunjukkan contoh sederhana perhitungan dengan menggunakan metode program dinamis. Transaksi pada contoh ini menggunakan transaksi yang berukuran sangat besar.

Berikut ini akan diberikan contoh ukuran transaksi dan biaya total dengan perhitungan (1) dengan mengabaikan penambahan atau pengurangan in. Perlu diingat bahwa biaya transaksi mempunyai suatu nilai minimum dan dihitung sesuai aplikasi yang dimiliki serta prioritas dari pengguna.

Tabel 1 Contoh Ukuran dan Biaya Transaksi

Idx	In	Out	Size (byte)	Harga (sat)
1	100	100	18210	1000
2	200	100	33010	1500
3	150	100	25610	1700
4	170	120	29250	1200
5	100	50	16510	1300

Penyelesaian program dinamis di atas akan diselesaikan dengan program dinamis untuk permasalahan knapsack, yaitu sebagai berikut :

$$f_0(y) = 0, \quad y = 0, 1, 2, \ldots, M$$

$$f_k(y) = -\infty, \quad y < 0$$

$$f_k(y) = \max\{f_{k-1}(y), p_k + f_{k-1}(y - w_k)\}, \quad k = 1, 2, \ldots, n$$

Dalam hal ini, $f_k(y)$ adalah keuntungan optimum yang didapat pada tahap k untuk kapasitas sebesar y. $f_0(y) = 0$ merupakan nilai dari persoalan *knapsack* kosong dengan kapasitas y. $f_k(y) = -\infty$ adalah nilai dari persoalan umtuk kapasitas yang negatif.

IV. Pemecahan Masalah dan Implementasi Program

A. Pemecahan Masalah

Asumsikan sebuah blok berukuran 100 KB (Dibandingkan dengan blok yang berukuran 1 MB pada Bitcoin) dan abaikan struktur lainnya kecuali transaksi. Ukuran yang dimiliki transaksi juga dibulatkan menjadi 20 KB terdekat untuk percobaan, meskipun pada kenyataannya diusahakan pendekatan dilakukan sekecil mungkin. Dengan menggunakan pendekatan demikian dan ukuran menjadi beban sedangkan harga menjadi proyeksi keuntungan, tentu diperoleh jumlah transaksi seperti berikut :

Tabel 2 Contoh Pembulatan Ukuran dan Biaya Transaksi

Idx	Size (KB)	Harga (sat)
1	20	1000
2	40	1500
3	40	1700
4	40	1200
5	20	1300

Penyelesaian di atas dilakukan secara lima tahap dengan menggunakan pendekatan secara maju sehingga :

Tahap 1 : $f_1(y) = \max\{f_0(y), 1000 + f_0(y - 20)\}$

y			Solusi Optimum	
	$f_0(y)$	$1000 + f_0(y - 20)$	$f_1(y)$	(x1*, x2*, x3*, x4*, x5*)
0	0	-∞	0	(0, 0, 0, 0, 0)
20	0	1000	1000	(1, 0, 0, 0, 0)
40	0	1000	1000	(1, 0, 0, 0, 0)
60	0	1000	1000	(1, 0, 0, 0, 0)
80	0	1000	1000	(1, 0, 0, 0, 0)
100	0	1000	1000	(1, 0, 0, 0, 0)

Tahap 2 : $f_2(y) = \max\{f_1(y), 1500 + f_1(y - 40)\}$

y			Solusi Optimum	
	$f_1(y)$	$1500 + f_1(y - 40)$	$f_2(y)$	(x1*, x2*, x3*, x4*, x5*)
0	0	-∞	0	(0, 0, 0, 0, 0)
20	1000	-∞	1000	(1, 0, 0, 0, 0)
40	1000	1500	1500	(0, 1, 0, 0, 0)
60	1000	2500	2500	(1, 1, 0, 0, 0)
80	1000	2500	2500	(1, 1, 0, 0, 0)
100	1000	2500	2500	(1, 1, 0, 0, 0)

Tahap 3 : $f_3(y) = \max\{f_2(y), 1700 + f_2(y - 40)\}$

y			Solusi Optimum	
	$f_2(y)$	$1700 + f_2(y - 40)$	$f_3(y)$	(x1*, x2*, x3*, x4*, x5*)
0	0	-∞	0	(0, 0, 0, 0, 0)
20	1000	-∞	1000	(1, 0, 0, 0, 0)
40	1500	1700	1700	(0, 0, 1, 0, 0)
60	2500	2700	2700	(1, 0, 1, 0, 0)
80	2500	3200	3200	(0, 1, 1, 0, 0)
100	2500	4200	4200	(1, 1, 1, 0, 0)

Tahap 4 : $f_4(y) = \max\{f_3(y), 1200 + f_3(y - 40)\}$

y			Solusi Optimum	
	$f_3(y)$	$1200 + f_3(y - 40)$	$f_4(y)$	(x1*, x2*, x3*, x4*, x5*)
0	0	-∞	0	(0, 0, 0, 0, 0)
20	1000	-∞	1000	(1, 0, 0, 0, 0)
40	1700	1200	1700	(0, 0, 1, 0, 0)
60	2700	2200	2700	(1, 0, 1, 0, 0)
80	3200	2900	3200	(0, 1, 1, 0, 0)
100	4200	3900	4200	(1, 1, 1, 0, 0)

Tahap 5 : $f_5(y) = \max\{f_4(y), 1300 + f_4(y - 20)\}$

y			Solusi Optimum	
	$f_4(y)$	$1300 + f_4(y - 20)$	$f_5(y)$	(x1*, x2*, x3*, x4*, x5*)
0	0	-∞	0	(0, 0, 0, 0, 0)
20	1000	1300	1300	(0, 0, 0, 0, 1)
40	1700	2300	2300	(1, 0, 0, 0, 1)
60	2700	3000	3000	(0, 0, 1, 0, 1)
80	3200	4000	4000	(1, 0, 1, 0, 1)
100	4200	4500	4500	(0, 1, 1, 0, 1)

Dari hasil di atas, ternyata didapatkan bahwa solusi optimum yaitu dengan memilih transaksi kedua, ketiga, dan kelima dengan total biaya transaksi yang didapat sekitar 4500 satoshi.

B. Implementasi Program

Melalui tahapan dengan menggunakan algoritma program dinamis, dapat dibuat suatu program untuk mencari transaksi dengan ukuran terkecil namun mempunyai biaya transaksi yang besar. Tingkat ketelitian juga akan ditingkatkan dan transaksi juga akan diperbanyak. Program yang dibuat dengan menggunakan bahasa pemograman python.

Program yang dibuat pertama-tama akan membaca suatu file transaksi misalkan diberi nama tx_pool.txt yang berisi

suautu data transaksi. Dari pembaca file tersebut kemudian akan dibuat suatu struktur data transaksi yang mempunyai suatu indeks, ukuran, dan biaya transaksi tersebut.

Selanjutnya program akan melakukan pencarian dengan menggunakan proses rekursif. Proses rekursif mempunyai basis yaitu saat transaksi yang terakhir atau saat ukuran maksimal sudah tidak dapat ditambahkan lagi. Implementasi yang dibuat cukup berbeda dengan algoritma di atas karena menggunakan metode *bottom-up*. Selanjutnya akan dilakukan perbandingan setiap rekursif untuk mencari nilai terbesar dan menentukan apakah pada suatu transaksi ke-*i* akan diambil atau tidak.

```
procedure find_transactions(input size : integer,
                           input txs : array[1..n] of Transaction,
                           input n : integer)
    # Base Case
    if n == 0 or size == 0 then
        return 0, []
    # Get the transaction
    tx = txs[n-1]
    # If weight of nth items > than available size
    if (tx.r_size > size) then
        return find_transactions(size, txs, n-1)
    # If still available
    else
        # Get the first and second choice
        total_1, chs_1 = find_transactions(size - tx.r_size, txs, n - 1)
        total_2, chs_2 = find_transactions(size, txs, n - 1)
        # Add to fee
        total_1 = total_1 + get_fee(tx)
        # Add the transaction to list
        chs_1 = add_transaction(chs_1, tx)

        # Check the max and return the maximum profit
        if (total_1 >= total_2) then
            return total_1, chs_1
        else
            return total_2, chs_2
```

Gambar 6 *Pseudocode* Program

Dengan melakukan beberapa percobaan, berikut ada tampilan hasil yang didapatkan :

```
TRANSACTION POOL
Transaction 0, size 20, fee 1000
Transaction 1, size 40, fee 1500
Transaction 2, size 30, fee 1700
Transaction 3, size 30, fee 1200
Transaction 4, size 30, fee 1300

==========

Total Fee :
4500
Index Transaction Choices :
['1', '2', '4']
```

```
TRANSACTION POOL
Transaction 0, size 20, fee 1000
Transaction 1, size 40, fee 1500
Transaction 2, size 30, fee 1700
Transaction 3, size 30, fee 1200
Transaction 4, size 30, fee 1300
Transaction 5, size 30, fee 1500
Transaction 6, size 10, fee 1500
Transaction 7, size 10, fee 1000
Transaction 8, size 20, fee 1300

==========

Total Fee :
7000
Index Transaction Choices :
['2', '5', '6', '7', '8']
```

```
TRANSACTION POOL
Transaction 0, size 20, fee 1000
Transaction 1, size 40, fee 1500
Transaction 2, size 30, fee 1700
Transaction 3, size 30, fee 1200
Transaction 4, size 30, fee 1300
Transaction 5, size 30, fee 1500
Transaction 6, size 10, fee 1500
Transaction 7, size 10, fee 1070
Transaction 8, size 20, fee 1340
Transaction 9, size 20, fee 1400
Transaction 10, size 30, fee 1400
Transaction 11, size 30, fee 1560
Transaction 12, size 10, fee 1000
Transaction 13, size 20, fee 1250
Transaction 14, size 20, fee 1350
Transaction 15, size 20, fee 1750

==========

Total Fee :
8420
Index Transaction Choices :
['2', '6', '7', '9', '12', '15']
```

Gambar 6 Contoh hasil dari program

V. ANALISIS

Dari hasil yang telah didapat, ternyata dengan menggunakan algoritma program dinamis mampu meghasilkan biaya transaksi terbesar yang dapat dimasukkan ke dalam blok berukuran tertentu sehingga penambang dapat mendapatkan keuntungan terbesar. Selain itu, algoritma ini juga cukup cepat dibandingkan algoritma yang lainnya.

Namun perlu dipertimbangkan juga bahwa pada kenyataannya, faktor penentu pengambilan suatu transaksi tidak hanya ditentukan oleh ukuran dan biaya transaksi tersebut, melainkan perlu memperhatikan juga waktu transaksi dimulai sehingga transaksi yang lama dapat didahulukan untuk mencegah fenomena *starving* untuk transaksi. Selain itu biaya transaksi juga dipertimbangkan oleh besar energi yang dibutuhkan serta nilai prioritas suatu transaksi. Data transaksi yang ada masih dapat diolah lebih dalam lagi dan harus mempelajari proses transaksi yang berjalan pada sistem teknologi Blockchain yang menggunakan metode Proof-of-Work seperti Bitcoin.

Oleh karena itu, tentunya diperlukan rumusan baru yang mempertimbangkan berbagai faktor yang ada dalam pengambilan transaksi pada Bitcoin. Ketelian ukuran transaksi

juga harus diperbesar untuk memperkecil kemungkinan fenomena *internal fragmentation*, yaitu masih tersedianya ruang kosong untuk suatu blok. Diperlukan juga optimisasi algoritma karena pada kenyataannya transaksi yang ada pada jaringan Bitcoin banyak dan satu blok pada Bitcoin dapat menampung ratusan bahkan ribuan transaksi.

VI. KESIMPULAN

Algoritma program dinamis merupakan suatu metode pemecahan masalah dengan menguraikan solusi menjadi sekumpulan langkah atau tahapan sedemikian sehingga dapat dipandang sebagai serangkaian keputusan yang saling berkaitan. Dengan memecahkan masalah menjadi masalah-masalah yang lebih kecil, membuat algoritma ini lebih cepat dibandingkan algoritma-algoritma seperti *greedy* maupun *exhaustive search*. Algoritma program dinamis mempunyai banyak implementasi pada kehidupan nyata, salah satunya yaitu untuk mencari keuntungan terbesar bagi para penambang di jaringan Bitcoin. Dengan algoritma program dinamis dan rumus yang tepat, maka hasil yang didapatkan optimal.

UCAPAN TERIMA KASIH

Penulis ingin mengucapkan terima kasih kepada Ibu Masayu Leylia Khodra, Ibu Nur Ulfa Maulidevi, dan Bapak Rinaldi Munir selaku dosen mata IF2211 kuliah Strategi Algoritma program studi Teknik Informatika yang telah membimbing penulis dengan baik selama satu semester dan telah memberikan contoh dasar dalam membuat sebuah makalah.

REFERENSI

[1] Nakamoto, Satoshi. 2008. Bitcoin : A Peer-to-Peer Electronic Cash System. Diambil dari https://bitcoin.org/bitcoin.pdf tanggal 12 Mei 2018.

[2] Munir, Rinaldi. 2018. Diktat Kuliah IF2211 Strategi Algoritma. Bandung : Institut Teknologi Bandung.

[3] Cormen, T. H, Leiserson, C. E., Rivest, R. L., Stein, C. 2009. Introduction to Algorithms, Third Edition. London : Massachusetts Institute of Technology.

[4] Vaidya, Kiran. 2016. Bitcoin's implementation of Blockchain. Diambil dari https://medium.com/all-things-ledger/bitcoins-implementation-of-blockchain-2be713f662c2 tanggal 12 Mei 2018.

[5] Rosic, A. 2017. Proof of Work vs Proof of Stake: Basic Mining Guide. Blockgeeks. Diambil dari https://blockgeeks.com/guides/proof-of-work-vs-proof-of-stake/ tanggal 12 Mei 2018.

[6] Bitcoinwiki. 2017. Transaction Fees. Diambil dari https://en.bitcoin.it/wiki/Transaction_fees tanggal 12 Mei 2018

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, 12 Mei 2018

Joseph Salimin 13516037