Penerapan Struktur Graf dan Fungsi *Hash* pada Teknologi *Blockchain*

Jimly Firdaus - 13521102
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13521102@std.stei.itb.ac.id

***Abstrak—Blockchain* merupakan teknologi yang cukup baru dalam dunia transaksi / perbankan. *Blockchain* sendiri bertujuan untuk menghilangkan *third-party* dalam transaksi dan meningkatkan keamanan dalam transaksi. Konsep *blockchain* memanfaatkan struktur data graf, pohon, dan teori bilangan (fungsi *hash*). Pada makalah ini akan dibahas pemanfaatan graf sebagai struktur data dan fungsi *hash* dalam *blockchain*.**

***Kata kunci—Blockchain,* Graf, Transaksi, *Security, Hash, Cryptocurrency*.**

I. PENDAHULUAN

Blockchain adalah kumpulan *block* dan setiap dari *block* tersebut, tersimpan suatu data yang berupa *ledger* / transaksi. Konsep dari *blockchain* sebenarnya sudah pernah disebutkan dalam sebuah *research paper* pada tahun 2008 oleh sekelompok entitas (yang disebut Satoshi Nakamoto) yang masih tidak diketahui identitas aslinya. Sejak saat itu, banyak jaringan *cryptocurrency* yang mulai dibuat / diciptakan. Beberapa aplikasi dari teknologi *blockchain* yaitu *cryptocurrency*, *banking* dan sektor finansial, *supply chain and logistics*, dan masih banyak lagi. Konsep dari *blockchain* sendiri terinspirasi dari algoritma *timestamp-ordering*, yang pada saat sebelum *blockchain* dipublikasikan ke publik, kegunaan dari algoritma *timestamp-ordering* digunakan untuk mencegah *tampering of documents*. Algoritma tersebut kemudian di-*extend* pada ruang lingkup transaksi dalam kehidupan sehari-hari untuk kepentingan keamanan sistem pembayaran. Tujuan saat teknologi *blockchain* pertama kali diperkenalkan kepada publik adalah untuk menyelesaikan 2 permasalahan besar yaitu :

1. *Double spending problem*.
2. Pihak ketiga sebagai agen *trust* dalam transaksi.

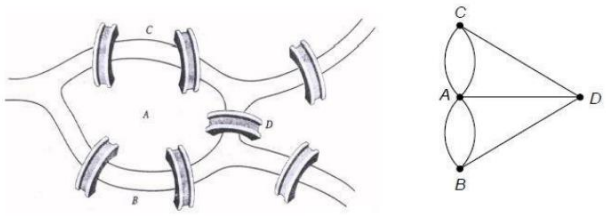
Setiap *block* pada *blockchain* mengandung suatu data yang berupa transaksi. *Block* yang paling pertama terbuat disebut sebagai *genesis block*. Setiap *block* dalam *blockchain* saling terhubung melalui *linkages* pada *block* sebelumnya. Setiap transaksi baru akan menciptakan *block* yang baru juga. *Block-block* tersebut kemudian akan disimpan ke dalam setiap *node-node* dalam suatu jaringan *peer-to-peer*. Setiap *block* juga memiliki keterhubungan dengan *block* yang lain sehingga dalam implementasinya, *blockchain* memanfaatkan struktur data berupa graf.

II. DASAR TEORI

A. GRAF

1. Pengertian Graf

Graf merupakan sekumpulan objek terstruktur dimana beberapa pasangan objek memiliki hubungan / keterkaitan tertentu. Menurut catatan sejarah, graf pertama kali digunakan untuk memecahkan permasalahan jembatan *Königsberg* pada tahun 1736. Permasalaan jembatan *Königsberg* yaitu untuk menentukan apakah mungkin melalui ketujuh buah jembatan, masing-masing tepat satu kali dan kembali ke tempat semula. Pada tahun 1736, seorang matematikawan Swiss, Leonhard Euler, adalah orang pertama yang berhasil menemukan jawaban permasalahan jembatan *Königsberg* dengan pembuktian yang sederhana. Ia memodelkan permasalahan ini ke dalam bentuk graf.



Gambar 2.1 Jembatan *Königsberg* dan graf yang merepresentasikan jembatan *Königsberg.*

Daratan (titik-titik yang dihubungkan oleh jembatan) dinyatakan sebagai titik yang disebut simpul (*vertex*) dan jembatan dinyatakan sebagai garis yang disebut (*edges*). Jawaban Euler menyatakan bahwa tidak mungkin melalui ketujuh jembatan itu tepat satu kali dan kembali ke tempat asal jika *derajat* setiap simpul tidak seluruhnya berjumlah genap. Derajat adalah banyaknya garis (*edges*) yang bersisian dengan simpul (*vertex*).

Secara matematis, graf didefinisikan sebagai pasangan himpunan simpul (*vertex*) dan sisi (*edges*). Secara formal, graf dinyatakan sebagai berikut:

$$G = < V, E >$$

Graf G terdiri atas himpunan simpul (V) dan himpunan sisi (E), yang dalam hal ini:

V = himpunan tidak kosong dari simpul-simpul (*vertices* atau *node*) = $v_1, v_2, v_3 \ldots v_n$

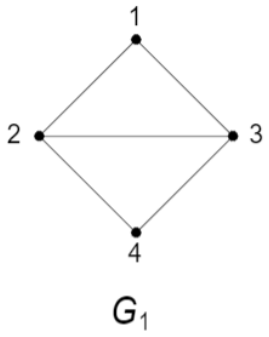
E = himpunan sisi (*edges* atau *arcs*) yang menghubungkan sepasang simpul = $e_1, e_2, e_3 \ldots e_n$

Graf G tidak boleh memiliki himpunan V yang kosong. Sebuah graf dimungkinkan tidak memiliki sisi, tetapi simpulnya (*vertex*) tetap harus ada, minimal satu. Graf yang hanya mempunyai satu buah simpul tanpa sebuah sisi pun dinamakan **graf trivial.**

2. Jenis-Jenis Graf

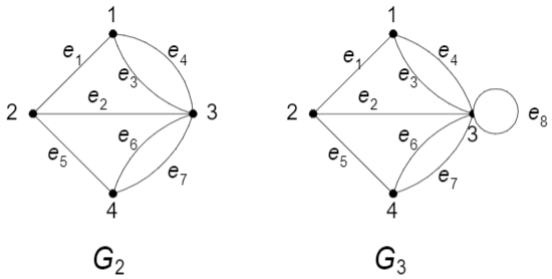
Berdasarkan ada tidaknya gelang atau sisi ganda pada suatu graf, secara umum graf dapat digolongkan menjadi dua jenis:

1. Graf sederhana (*simple graph*)
Graf yang tidak mengandung gelang maupun sisi ganda.



Gambar 2.2 Graf sederhana

2. Graf tak sederhana (*unsimple graph*)
Graf yang mengandung sisi ganda atau gelang. Terdapat 2 graf tak sederhana yaitu graf ganda (*multigraph*) dan graf semu (*pseudograph*). Graf ganda adalah graf yang mengandung sisi ganda tetapi tidak mengandung gelang. Graf semu adalah graf yang mengandung gelang.

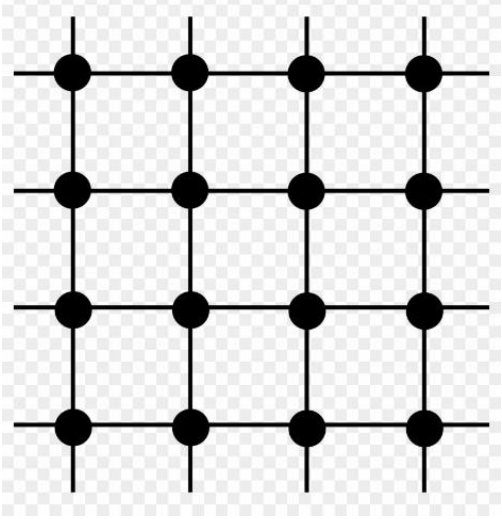


Gambar 2.3 Graf ganda (kiri) dan graf semu (kanan)
Sumber : https://2.bp.blogspot.com/-mSgSuS8GsMY/WkhPG3k-rlI/AAAAAAAAAgI/aZdnQWq1X_8z_mbL8FqVECnKP_5mr-doQCLcBGAs/s1600/20.jpg

Berdasarkan jumlah simpul pada suatu graf, secara umum graf dapat digolongkan menjadi 2 jenis yaitu:

1. Graf berhingga (*limited graph*)
Graf berhingga adalah graf yang jumlah simpulnya berhingga. Contoh graf berhingga terdapat pada gambar 2.2 dan 2.3.

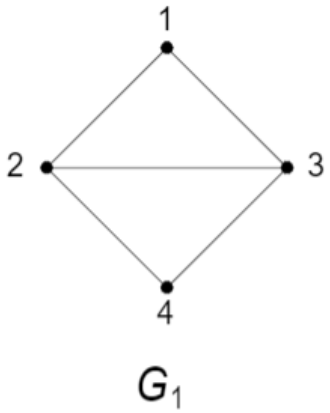
2. Graf tak-berhingga (*unlimited graph*)
Graf yang jumlah simpulnya tidak berhingga banyaknya.



Gambar 2.4 Graf tak berhingga
Sumber :
https://www.clipartmax.com/middle/m2i8m2A0H7b1b1Z5_finite-and-infinite-graph/

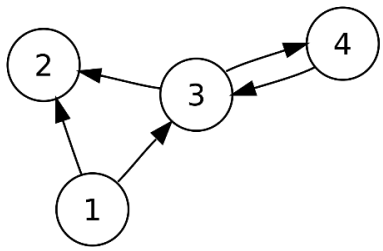
Berdasarkan orientasi arah pada sisi, secara umum graf dibedakan atas 2 jenis:

1. Graf tak-berarah (*undirected graph*)
Graf yang sisinya tidak mempunyai orientasi arah disebut graf tak-berarah. Pada graf tak-berarah, urutan pasangan simpul yang dihubungkan oleh sisi tidak diperhatikan.



Gambar 2.5 Graf tak-berarah

2. Graf berarah (*directed graph* atau *digraph*)
Graf yang setiap sisinya diberikan orientasi arah disebut sebagai graf berarah. Pada graf berarah, $(v_j, v_k) \neq (v_k, v_j)$.



Gambar 2.6 Graf berarah
Sumber :
https://en.wikipedia.org/wiki/File:Directed_graph_no_background.svg

3. Terminologi Graf

a) Bertetangga (*adjacent*)
Dua buah simpul pada graf tak-berarah bertetangga jika keduanya terhubung lansung dengan sebuah sisi.

b) Bersisian (*incident*)
Untuk sembarang sisi $e = (v_j, v_k)$, sisi e dikatakan bersisian dengan simpul v_j dan simpul v_k.

c) Simpul Terpencil (*isolated vertex*)
Simpul terpencil jika tidak memiliki sisi yang bersisian dengan simpul tersebut.

d) Graf Kosong (*null vertex*)
Graf yang himpunan sisinya merupakan himpunan kosong.

e) Derajat (*degree*)
Derajat suatu simpul pada graf tak-berarah adalah jumlah sisi yang bersisian dengan simpul tersebut.

f) Lintasan (*path*)
Barisan berselang-seling simpul-simpul dan sisi-sisi yang berbentuk $v_0, e_1, v_1, e_2, \dots, v_{n-1}, e_n$ sedemikian sehingga $e_1 = (v_0, v_1)$, $e_2 = (v_1, v_2)$, … , $e_n = (v_{n-1}, v_n)$ adalah sisi dari graf G.

g) Sirkuit (*circuit*)
Lintasan yang berawal dan berakhir pada simpul yang sama.

h) Terhubung (*connected*)
Setiap pasang simpul v_i dan v_j dalam himpunan V terdapat lintasan dari v_i ke v_j (juga harus terdapat lintasan dari v_j ke v_i). Jika tidak, maka disebut graf tak-terhubung (*disconnected graph*).

i) Upagraf (*subgraph*)
Jika terdapat $G = (V,\ E)$ maka $G_1 = (V_1,\ E_1)$ adalah upagraf dari G jika $V_1 \subseteq V$ dan $E_1 \subseteq E$.

j) *Cut-Set*
Cut-set dari graf terhubung G adalah himpunan sisi yang bila dibuang dari G menyebabkan Gtidak terhubung.

k) Graf Berbobot (*weighted graph*)
Graf yang setiap sisinya diberi sebuah harga / bobot.

B. Fungsi Hash

Fungsi hash merupakan fungsi yang digunakan untuk melakukan *map* dari *key data* yang diberikan menjadi suatu *arbitrary size value*. Salah satu bentuk hash yang paling umum yaitu:

$$h(K) = K \bmod m$$

Kriptografi merupakan salah satu implementasi dari fungsi *hash*, dimana setiap *key* akan dilakukan *encryption* menjadi sebuah informasi / data yang tidak memiliki makna lagi. Hasil *encryption* tersebut harus dilakukan *decrypt* sehingga kembali lagi menjadi *key* semula.

Kriteria fungsi *hash* dalam kriptografi berupa:

1. *Pre-image resistance* : setiap *output* dari fungsi *hash* harus menghasilkan *hash value* yang membuat *value* asli menjadi sangat sulit untuk didapatkan kembali.

2. *Second pre-image resistance* : setiap *output* dari fungsi *hash* memiliki kemungkinan yang sangat kecil untuk memiliki *hash value* yang sudah pernah dihasilkan untuk *value* yang berbeda.

3. *Collision Resistance* : setiap *output* dari fungsi *hash* memiliki kemungkinan yang sangat kecil untuk menghasilkan *hash value* yang persis sama dengan *hash value* yang lain.

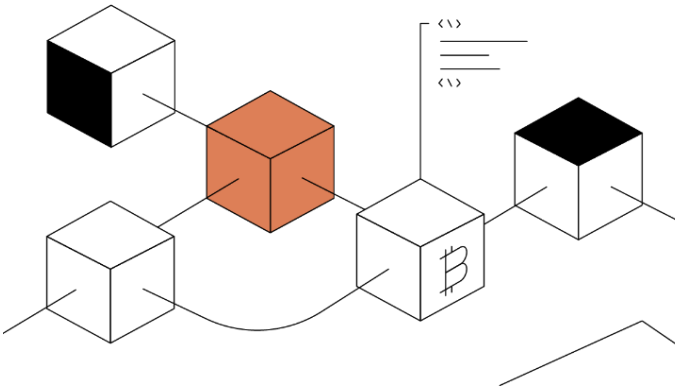
Beberapa contoh fungsi *hash* adalah *RSA*, *Message Digest* (MD), *Secure Hash Function* (SHA), *RACE Integrity Primitives Evaluation Message Digest* (RIPEMD), *Whirlpool*.

III. *Blockchain & Validator Node*

A. Pengertian

1. Blockchain

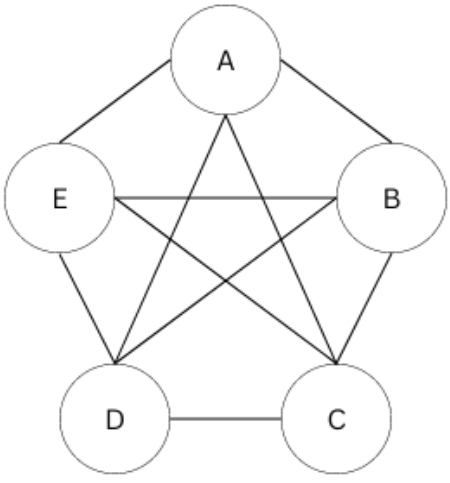
Blockchain merupakan teknologi yang masih tergolong baru dalam dunia transaksi dan perbankan. Konsep *blockchain* diperkenalkan oleh seorang / entitas yang bernama Satoshi Nakamoto. *Blockchain* sendiri dibangun atas kumpulan-kumpulan *block-block* yang membawa beberapa data. Data tersebut berupa nomor *block*, data transaksi, *nonce*, *block hash*, dan *hash block* sebelumnya. Setiap *block* bersifat unik karena memiliki data transaksi yang berbeda, dan setiap *block* memiliki *hash block* sendiri. *Hash block* ini didapat dari *hash function* yang sudah cukup umum dalam kriptografi, seperti dengan SHA -256. Setiap *block* akan menggunakan *hash block* tersebut untuk me-*link* dengan block yang lain, membentuk sebuah *chain*. Kegunaan utama dari *blockchain* sendiri adalah untuk me-*record* setiap transaksi yang terjadi. Oleh karena itu, sebuah *blockchain* akan terus bertumbuh, sesuai dengan *record* transaksi yang telah terjadi yang disimpan dalam sebuah *block*.



Gambar 3.1 Ilustrasi *block* pada *blockchain*
Sumber : https://www.gemini.com/cryptopedia/blockchain-technology-explained

2. *Validator Node*

Validator node adalah sebuah *node* yang bertindak sebagai "partisipan" dalam sebuah konsensus (*blockchain cryptographic proof*). Dengan "berpatisipasi" dalam konsensus, setiap *validator node* bertanggung jawab untuk mem-*verify* setiap transaksi baru, voting, dan menjaga *record* dari setiap transaksi. Pada makalah ini, fungsi *validator node* yang akan dibahas hanya bagaimana *validator nodes* mem-*verify* setiap transaksi baru. Setiap *node-node* yang berfungsi sebagai *validator nodes* dalam sebuah konsensus / *blockchain* terhubung antara satu dengan yang lain dengan *edges* membentuk sebuah graf lengkap. Dengan *edges* tersebut, *node-node* dapat saling "berkomunikasi" dan mengirim data / informasi. Setiap *validator-node* akan memiliki *copy*-an utuh dari *blockchain.* Jika terdapat sebuah transaksi baru, maka salah satu *validator node* akan melakukan verifikasi dan kemudian informasi dan data tersebut akan disebar kepada *validator node* yang lain dalam konsensus tersebut.



Gambar 3.2 Ilustrasi *validator node* pada sebuah *peer-to-peer network*

B. Hubungan Validator Node dengan Blockchain

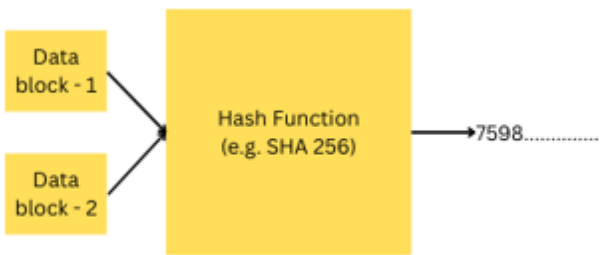
Pada suatu jaringan *peer-to-peer*, akan terdapat beberapa *validator node* dan satu *blockchain* yang merepresentasikan *ledger* dari seluruh sistem di jaringan *peer-to-peer* tersebut. Jika ada transaksi baru, salah satu *validator node* pada jaringan tersebut, pertama akan melakukan verifikasi terhadap data transaksi yang akan direcord. Jika verifikasi gagal, maka *block* baru akan gagal terbentuk. Jika verifikasi sukses, maka *block* baru akan terbentuk dengan data transaksi saat itu beserta *hash block. Block* ini disebut sebagai *genesis block.* Karena setiap *validator node* memiliki *copy*-an utuh dari *blockchain*, maka *validator node* yang pertama kali memverifikasi dan membuat *block* baru akan mengirimkan informasi kepada *validator node* yang lain untuk membuat *block* yang baru sehingga *state blockchain* yang disimpan tetap sama. Namun setiap *validator node* yang lain akan melakukan verifikasi terhadap *block* baru tersebut.

Sebuah *block* akan terbentuk dan masuk ke dalam *chain* jika terdapat *vote* di atas 50% dari total *validator node* pada jaringan *peer-to-peer* tersebut. Hal ini akan menyebabkan *attacker* kesulitan dalam memanipulasi data transaksi karena setiap *validator node* akan melakukan verifikasi terhadap *block-block* baru. Setiap *block* baru yang terbentuk akan mendapatkan *hash block. Hash block* ini bergantung pada data transaksi karena *value* yang di-*hash* yaitu data transaksi itu sendiri sehingga jika terdapat *attacker* yang ingin memanipulasi data transaksi, hal ini akan menyebabkan *hash block* tersebut menjadi berubah. Karena setiap *block* dihubungkan dengan *hash block* sebelumnya, *block* tersebut akan terputus dari *chain* yang sudah terbentuk dan jika *attacker* ingin memperbaiki *hash block*, kemungkinannya sangat kecil karena hasil dari setiap *hash block* berbeda dengan sebelumnya sehingga sangat kecil kemungkinan untuk menyambungkan *block* yang sudah diubah kembali ke dalam *chain.*

IV. *Hashing*

A. Pengertian

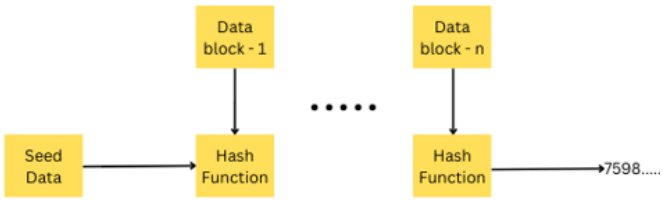
Hashing dilakukan pada dua *blocks* yang memiliki ukuran data yang sama. Ukuran dari setiap *block* data bervariasi, bergantung pada algoritma *hashing* yang digunakan. Umumnya, ukuran data *ranging* pada rentang 128 – 512 *bits.* Adapun algoritma *hashing* adalah proses untuk melakukan *hashing* dengan setiap data pada masing-masing *block* dan fungsi *hash* me-*hash* setiap data dan menghasilkan *hash value* yang akan digunakan selama algoritma *hashing* dilakukan.



Gambar 3.3 Ilustrasi *hashing* pada *blocks*

Algoritma dari *hashing* sendiri adalah dengan mengulang ilustrasi pada gambar 3.3. Setiap putaran, fungsi *hash* menerima

input berukuran tetap, umumnya berupa kombinasi dari data yang di-*hash* sebelumnya. *Hashing* dilakukan untuk keseluruhan data yang di-*input.* Ilustrasi dari kerja algoritma *hashing* ini adalah sebagai berikut:



Gambar 4.2 Ilustrasi algoritma *hashing*

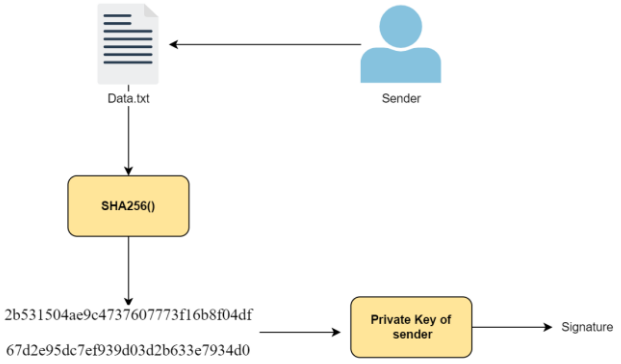
Setiap *block* akan di-*hash* berdasarkan hasil *hashing* dari *block* sebelumnya. Proses ini berulang hingga mencapai *hashing* pada *block* terakhir. Ini dikenal dengan sebutan *avalanche effect of hashing.*

Algoritma *hashing* yang paling umum digunakan dalam *blockchain* adalah algoritma SHA-256. SHA-256 adalah fungsi *hash* kriptografi tanpa *key* dengan mengambil *input* panjang *variable* dan menghasilkan *hash output* sepanjang 256-*bit.* SHA-256 banyak digunakan dalam proses *hashing* dalam *blockchain,* terutama dalam hal:

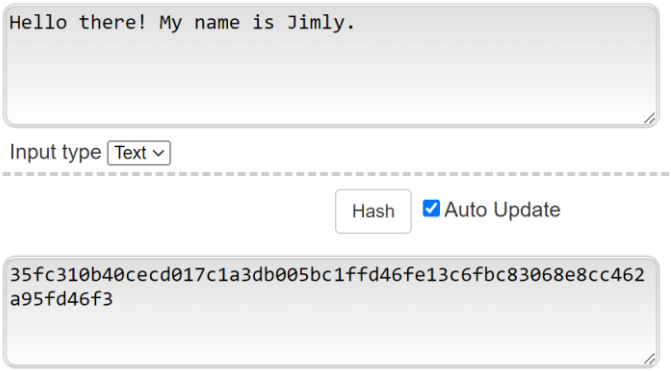
1. Mekanisme dalam konsensus : penambang *cryptocurrency* menghitung hasil *hash* dari setiap *block* baru yang terbentuk dengan menggunakan SHA-256 dan memvariasikan *value* dari *nonce* (jumlah *miner* yang melakuakan yang melakukan *solving* pada *blockchain*) sampai hasil *hash* mencapai dibawah batas tertentu. Hal ini dilakukan untuk meningkatkan keamanan pada *block* karena hasil *hash* lebih random / acak.

2. *Chains of Block* : setiap *block* dalam *blockchain* mengandung *hash* yang di-*generate* dengan menggunakan algoritma SHA-256.

3. *Digital Signatures* : setiap transaksi yang dilakukan, juga disertai dengan *signature* yang dilakukan secara *digital* untuk menjaga integritas. Informasi dalam setiap transaksi akan di-*hash* dengan menggunakan algoritma *hashing* SHA-256, dengan *key encryption* berupa sebuah *private key* yang dimiliki oleh pemilik transaksi. Informasi ini kemudian akan divalidasi oleh *miner* melalui *validator node.*



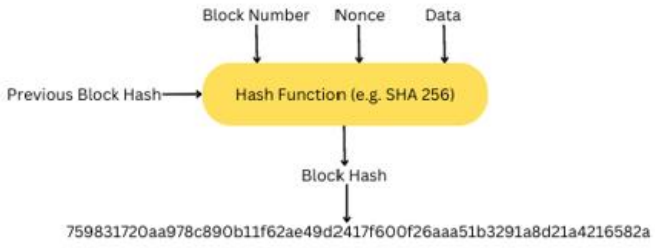
Gambar 4.3 Ilustrasi Proses *hashing* data dari pemilik transaksi
Sumber : https://www.educative.io/answers/how-is-sha-256-used-in-blockchain-and-why



Gambar 4.4 Contoh hasil *hashing* data dengan algoritma SHA-256
Sumber : https://emn178.github.io/online-tools/sha256.html

B. Cara Kerja Hashing

Setiap data transaksi yang sukses mendapatkan *vote* di atas 50%, akan dibentuk sebuah *block* baru untuk menampung *record* data transaksi tersebut. Pada proses pembuatan *block* baru, algoritma *hashing* dilakukan. Pertama akan dicari hasil *hash value* dari *block* terakhir dalam *blockchain.* Kemudian hasil *hash value* tersebut akan menjadi *seed value* untuk digunakan oleh algoritma *hashing* untuk melakukan *hashing* pada *block* baru. Setelah itu *block* baru akan dilakukan *hash* pada data yang disimpannya dengan *hash value block* terakhir yang sudah didapat. Hasil dari *hashing* ini akan menjadi *hash block* untuk *block* yang baru.



Gambar 4.3 Ilustrasi pembuatan *block*

V. KESIMPULAN

Blockchain merupakan teknologi yang masih baru dalam dunia transaksi dan perbankan. Konsep *blockchain* menawarkan

sistem transaksi yang berbeda dari sistem yang masih digunakan sekarang yaitu sistem perbankan yang berfungsi sebagai pihak ketiga (*agent of trust*) dalam setiap transaksi yang dilakukan dengan alasan keamanan. *Blockchain* bertujuan untuk menghilangkan *third party* dalam transaksi yang dilakukan. Oleh karena itu *blockchain* harus memiliki sistem yang lebih aman sehingga bisa menjamin transaksi dapat dilakukan secara *peer-to-peer*. *Blockchain* memberikan kemanan itu dengan beberapa cara seperti dengan me-*record* setiap transaksi ke dalam sebuah *block* untuk dilakukan *tracking* dan setiap transaksi yang terjadi harus melewati beberapa tahapan untuk diverifikasi oleh *nove validator*. Hal ini akan mencegah adanya *fraud* dalam transaksi. Hal ini ditambah dengan keamanan dari data transaksi yang telah dibuat, dimana setiap data akan disimpan dalam sebuah *block* kemudian akan dilakukan *hashing* kriptografi sehingga menghasilkan *hash value* yang sedikit kemungkinan untuk bisa ditebak data yang di-*hash.* Dengan struktur data berupa graf yang lebih kompleks serta fungsi *hash* yang digunakan, *blockchain* dapat menjadi teknologi yang lebih aman dibandingkan dengan sistem perbankan.

VI. Ucapan Terima Kasih

Dengan selesainya makalah ini dibuat, saya mengucapkan rasa syukur sebesar-besarnya kepada Tuhan, atas hikmat yang diberikan-Nya untuk menyelesaikan makalah ini. Saya juga berterima kasih kepada ibu Dr. Nur Ulfa Maulidevi, S.T, M.Sc dan bapak Dr. Ir. Rinaldi Munir, M.T selaku dosen yang membimbing saya dalam mata kuliah Matematika Diskrit.

Referensi

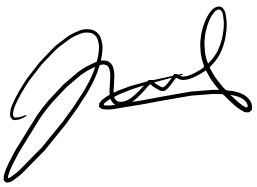
[1] https://www.javatpoint.com/blockchain-hash-function, diakses pada 2 Desember 2022.
[2] https://learn.bybit.com/blockchain/what-is-hashing-in-blockchain/, diakses pada 2 Desember 2022.
[3] https://learn.radixdlt.com/article/what-is-a-validator-node#:~:text=%EF%81%9A,maintaining%20a%20record%20of%20transactions., diakses pada 2 Desember 2022.
[4] https://www.gemini.com/cryptopedia/blockchain-technology-explained, diakses pada 2 Desember 2022.
[5] https://www.educative.io/answers/how-is-sha-256-used-in-blockchain-and-why, diakses pada 2 Desember 2022.
[6] https://coinmarketcap.com/alexandria/glossary/sha-256, diakses pada 2 Desember 2022.
[7] https://www.investopedia.com/terms/n/nonce.asp#:~:text=difficulty%20level%20restrictions.-,The%20nonce%20is%20the%20number%20that%20blockchain%20miners,to%20receive%20the%20block%20reward., diakses pada 2 Desember 2022.
[8] Rinaldi Munir, Diktat Kuliah Matematika Diskrit Edisi Keempat

Link Video : https://youtu.be/nhq4shk06EY

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, 2 Desember 2022



Jimly Firdaus, 13521102