Penerapan Algoritma Backtracking untuk Memaksimalkan Hashrate pada Ethereum Classic Mining

Rayhan Asadel (13519196)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13519196@std.stei.itb.ac.id

***Abstract*— *Cryptocurrency* adalah suatu jenis *digital asset* yang menganut desentralisasi. Asset jenis ini tidak memerlukan lembaga pemerintah seperti bank sental dan bank umum untuk mengatur dan mengelola keberadaannya. Ethereum Classic merupakan salah satu dari *cryptocurrency* tersebut. Dalam ekosistem *cryptocurrency* terdapat peran yang bernama *miner*. *Miner* ini adalah individu atau sekelompok orang yang mengerahkan kekuatan komputasinya untuk menjaga dan memverifikasi transaksi yang terjadi di *blockchain cryptocurrency* tersebut. Atas jasanya *miner* mendapatkan intensif berupa *cryptocurrency* yang ia bantu dalam "mining" tersebut. Untuk memaksimalkan profit, *miner* melakukan optimalisasi pada GPUnya. Optimalisasi ini ditentukan dari peningkatan *hashrate* dari GPU tersebut. Dengan menggunakan algoritma *backtracking*, kita dapat mengoptimalisasikan *hashrate* dari GPU tersebut tanpa takut *overkill* dan membuat system menjadi tidak stabil (*crash*).**

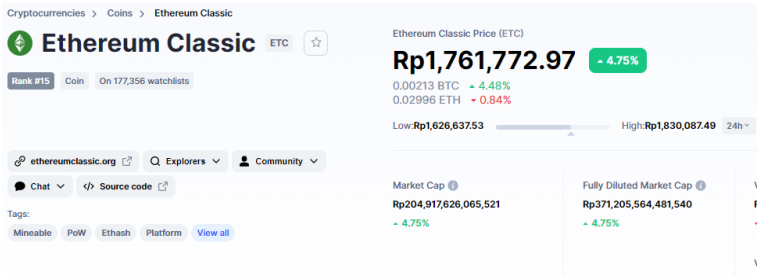
Keywords*—*Cryptocurrency, Ethereum Classic, GPU Mining, Miners, Hashrate, Overclocking, Profit.

I. PENDAHULUAN

Cryptocurrency adalah suatu jenis *digital asset* yang menganut filosofi desentralisasi, dimana asset ini tidak memerlukan lembaga pemerintah seperti bank sentral, untuk mengeluarkan, mengatur, dan menjaga keberlangsungan system keuangan. Untuk mencapai filosofi tersebut *cryptocurrency* memanfaatkan teknologi *blockchain* dan *cryptography* untuk menyimpan data transaksi, dan mengamankan serta memverifikasi transaksi baru yang terjadi di jaringan tersebut. Dalam memproses transaksinya, *cryptocurrency* menggunakan kekuatan komputasi untuk memverifikasi transaksi, hal ini mengeliminasi liabilitas jika dibandingkan dengan system keuangan yang tersentralisasi, seperti, kesalahan manusia, serta kebijakan-kebijakan otoritas yang merugikan, oleh karena itu berlaku istilah "*Code is Law*".

Cryptocurrency pertama yang diciptakan adalah Bitcoin, yang mulai dirilis ke kalangan umum secara *open source* pada tahun 2009. Pada saat itu Bitcoin tidak memiliki nilai yang berarti, namun karena Bitcoin diciptakan dengan supply yang terbatas dan dengan berkembangnya adopsi Bitcoin oleh lembaga-lembaga keuangan, 1 Bitcoin pada saat makalah ini ditulis bernilai Rp. 826.780.000, dan dengan kapitalisasi pasar $1,093,232,803,022, no 1 diantara *cryptocurrency lainnya*.

Ethereum Classic merupakan suatu *cryptocurrency* yang terinspirasi dari Bitcoin, yang memiliki fitur tambahan yaitu teknologi *smart contract*. *Smart contract* ini memungkinkan eksekusi suatu transaksi, berdasarkan kejadian atau event-event yang harus dipenuhi. Fitur ini dapat dikembangkan untuk membangun DApps (Decentralized Application) dan DeFi (Decentralized Finance) yang di dalamnya terdapat fitur-fitur seperti *peer to peer lending*, *coin exchange*, dan lain sebagainya. Hingga makalah ini ditulis, 1 Ethereum Classic memiliki nilai sebesar Rp. 1.761.772, dan memiliki kapitalisasi pasar terbesar ke 15 diantara *cryptocurrency* yang lain.



Gambar 1-1: Nilai dan kapitalisasi pasar Ethereum Classic berdasarkan coinmarketcap.com (10/05/2021) (sumber: coinmarketcap.com)

II. DASAR TEORI

A. *Ethereum Classic*

Ethereum Classic merupakan *cryptocurrency* yang diciptakan oleh Vitalik Butterin dan Gavin Wood pada 30 Juli 2015. Pada saat diciptakan pertama kali Ethereum Classic memiliki nama hanya Ethereum saja. Ethereum Classic adalah *blockchain cryptocurrency* pertama yang menunjang teknologi *smart contract*. Dengan *smart contract* suatu individu dapat mengirim atau menerima *cryptocurrency* jika memenuhi kejadian atau event-event

tertentu. Teknologi ini menjadi dasar dibuatnya DApps (*decentralized apps)* dan DeFi (*decentralized finance)*.

Gambar 2-1: Logo Ethereum Classic terbaru. (sumber: Github Ethereum Classic)

Pada tahun 2016 Ethereum Classic yang waktu itu masih disebut Ethereum, menjadi korban penyerangan oleh *hacker* yang memanfaatkan celah pada protocol Ethereum. Akibat dari insiden itu tim developer Ethereum melakukan *hard-fork* untuk mengembalikan dana yang telah dicuri. Dengan *hard-fork* tersebut network baru yang dihasilkan dan yang sudah mengembalikan kerusakan akibat insiden tersebut disebut sebagai Ethereum, sedangkan *blockchain* lamanya yang terdampak akibat serangan tersebut diberi nama baru sebagai Ethereum Classic.

B. *GPU Mining*

Mining merupakan suatu proses dimana dengan menggunakan kekuatan komputasi (biasanya GPU) untuk menyelesaikan persoalan matematis untuk memverifikasi suatu transaksi pada *blockchain cryptocurrency*. Proses mining ini memberikan insentif hadiah berupa *cryptocurrency* tersebut pada subjek atau *Miner* yang melakukan *Mining.*

Jika dibandingkan dengan system keuangan konvensional, yang membutuhkan bank sentral, dan bank umum, yang di dalamnya terdapat jaringan serta karyawan untuk menjalankan transaksi, hal tersebut digantikan oleh kekuatan komputasi yang dilakukan dalam proses *Mining* tersebut.

Mining selain memberikan insentif kepada *miners* karena telah membantu mengurus dan menjalankan jaringan dari *cryptocurrency*, juga membantu dalam mengamankan jaringan *cryptocurrency* dari serangan-serangan pihak luar, semakin banyak kekuatan komputasi dalam suatu jaringan *cryptocurrency* maka akan semakin susah dan semakin besar sumber daya (kekuatan komputasi) yang perlu dikerahkan untuk menyerang jaringan *cryptocurrency* tersebut.

C. *Ethereum Classic Mining*

Pada awalnya, untuk melakukan *Mining* Ethereum Classic dibutuhkan GPU dengan *memory* VRAM lebih dari 4GB. Hal ini dikarenakan file DAG (*directed acyclic graph)* yang digunakan untuk menyimpan data historis transaksi sudah terlalu besar.

Pada 28 November 2020, tim developer dari *Ethereum Classic* melakukan *forking* dengan *codename thanos. Forking* ini akan mengurangi ukuran file DAG yang diperlukan untuk melakukan *Mining* Ethereum Classic. Inisiatif ini dilakukan agar para Miners yang masih menggunakan GPU dengan ≤ 4GB VRAM, tertarik untuk Mining Ethereum Classic dan meningkatkan *Network hashrate* yang berarti meningkatkan keamanan *network* Ethereum Classic secara keseluruhan.

Untuk melakukan mining dalam Ethereum Classic digunakan algoritma mining Etchash, yang merupakan pengembangan dari Ethash yang digunakan oleh Ethereum, sebelum Ethereum Classic melakukan Hardfork dan menghasilkan Ethereum.

Algoritma Etchash memiliki ketergantungan yang sangat tinggi terhadap *memory speed* dari VRAM (*video random access memory)* yang dimiliki oleh GPU. Karakteristik ini membuat *hashrate* (kecepatan penyelesaian solusi dari mining) akan memiliki pengaruh yang lebih besar terhadap *memory clock*, dibandingkan dengan *core clock.*

D. *Algoritma Backtracking*

Backtracking merupakan suatu algoritma pemecahan masalah, yang secara incremental membangun himpunan solusi yang mungkin, sembari memangkas kandidat-kandidat solusi yang sudah tidak mungkin mengarah ke solusi.

Algoritma *backtracking* ini merupakan pengembangan dan perbaikan dari *exhaustive search.* Berbeda dengan *exhaustive search*, algoritma *backtracking* tidak membangkitkan atau mencoba semua kemungkinan solusi yang ada. Pada Algoritma *backtracking* kemungkinan-kemungkinan solusi yang sudah tidak valid akan berhenti dieksplore dan dilanjutkan dengan mengeksplore kemungkinan solusi yang lain. Algoritma *backtracking* ini pertama kali diperkenalkan oleh D. H. Lehmer pada tahun 1950.

Algoritma *backtracking* memiliki beberapa property umum yang dapat didefinisikan sebagai berikut.

1. Solusi Persoalan

   Dapat dinyatakan sebagai vector atau n tuple X = $(x_1,x_2,\ldots,x_n)$, dengan $x_i \in S_i$, vector ini menyatakan apakah suato solusi diambil atau termasuk kedalam solusi yang mengarah ke tujuan.

2. Fungsi Pembangkit nilai x_k

   Dapat dinyatakan sebagai predikat T(), dengan $T(x_K)$ membangkitkan nilai x_k yang merupakan vector atau tuple dari solusi.

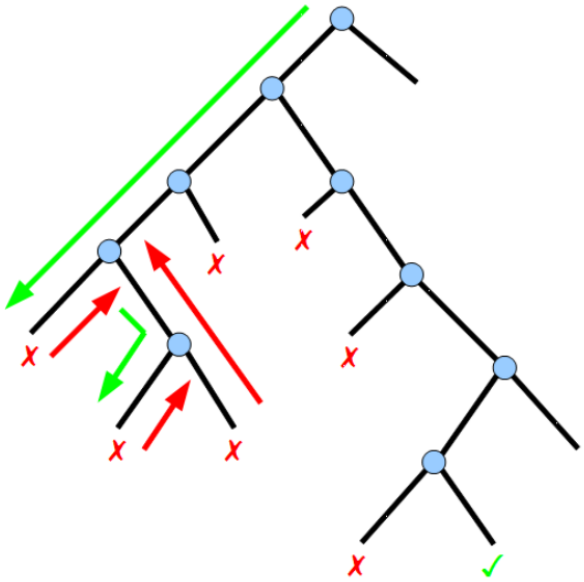
3. Fungsi Pembatas (*bounding function)*

   Dinyatakan sebagai predikat $B(x_1,x_2,\ldots,x_n)$, dan akan bernilai true jika $(x_1,x_2,\ldots,x_n)$ mengarah ke solusi dan tidak melanggar *constraint*. Jika bernilai true maka penelusuran akan diteruskan ke solusi x_{n+1}, dan jika false maka $(x_1,x_2,\ldots,x_n)$ akan dibuang dan dihilangkan dari kemungkinan solusi.

Semua kemungkinan solusi dari persoalan disebut dengan ruang solusi (*solution space*). Ruang solusi tersebut akan dituangkan ke dalam struktur pohon berakar (*rooted tree)*. Simpul-simpul dari pohon merupakan status (*state)*, sedangkan sisi dilabeli dengan nilai dari x_k.Suatu lintasan dari akar ke daun menyatakan suatu solusi, dan seluruh lintasan dari simpul akar ke daun membentuk ruang solusi secara keseluruhan.

Dalam melakukan pencarian solusi dengan menggunakan algoritma *backtracking* berlaku prinsip-prinsip sebagai berikut.

1. Solusi dicari dengan membangkitkan solusi ruang status sehingga tercipta lintasan dari simpul akar ke simpul daun
2. Pembangkitan simpul solusi dalam pohon menggunakan aturan *depth first search (DFS).*
3. Simpul yang sudah dibangkitkan dinamakan simpul hidup (*live node*)
4. Simpul hidup yang sedang kita periksa dinamakan simpul ekspan (*expand node*)
5. Jika lintasan yang sedang dibentuk dari simpul ekspan tidak mengarah ke solusi (melanggar bounding function) maka simpul tersebut dimatikan.
6. Jika lintasan yang sedang dibentuk dari simpul ekspan tidak melanggar fungsi pembatas (*bounding function)* pencarian dilanjutkan.
7. Jika pencarian berakhir pada simpul mati, maka pencarian akan *backtrack* (mundur) ke simpul diatasnya, dan memulai pencarian dengan mengekspan simpul anak lainnya.
8. Pencarian dihentikan apabila kita telah mencapai simpul tujuan.



Gambar 2-1: ilustrasi penelusuran pohon dengan *backtracking*

(sumber: http://www.w3.org/2011/Talks/01-14-steven-phenotype/)

III. PENERAPAN ALGORITMA BACKTRACKING UNTUK MEMAKSIMALKAN HASHRATE PADA ETHEREUM CLASSIC MINING

A. *Analisis masalah*

Permasalahan yang timbul dari persoalan ini adalah, bagaimana mencapai *hashrate* semaksimal mungkin pada Ethereum Classic Mining, tanpa menyebabkan system *Mining* crash, dengan kondisi GPU *out of the box* yaitu tanpa mengubah settingan GPU *core voltage* dan GPU *memory voltage*.

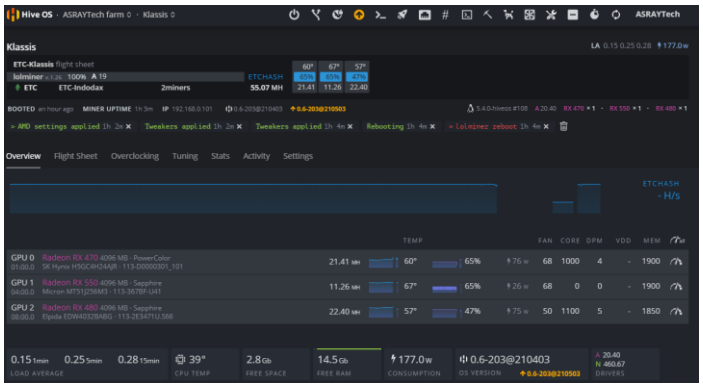
Dengan menerapkan backtracking dan melakukan inkrementasi secara berulang pada *memory clock* kita dapat mendekati frekuensi maksimal sebelum system menjadi tidak stabil atau bahkan hingga sistem crash.

B. *Ruang lingkup dan Batasan permasalahan*

Dalam melaksanakan pengujian sembari menerapkan *backtracking* ada beberapa constraint dan ruang lingkup yang harus didefinisikan terlebih dahulu, yaitu sebagai berikut.

1. Peningkatan memory clock, dilakukan secara bertahap dengan kelipatan 50mhz dan 100mhz.
2. Sistem komputer hanya menggunakan 1 GPU yang dihubungkan ke slot pcie menggunakan GPU Riser.
3. Tidak dilakukan *tweaking* pada *core voltage* dan *memory voltage* pada GPU.

4. Pengujian dilakukan dengan menggunakan *operating system* khusus mining untuk menambah kestabilan dan efisiensi, yaitu HiveOS.

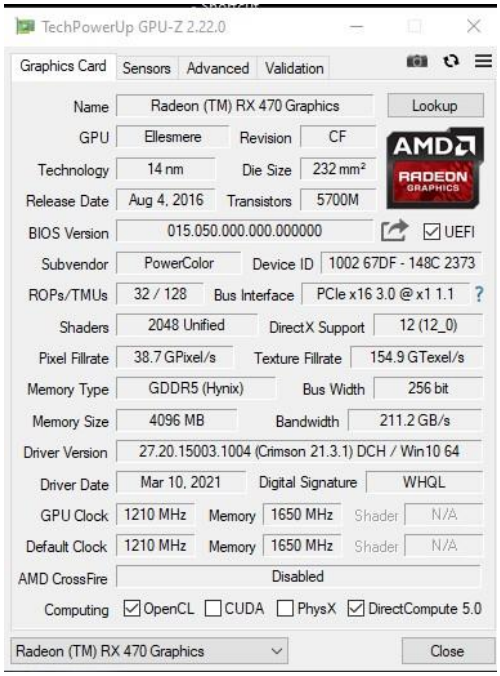


Gambar 3-1: Tampilan antarmuka remote access HiveOS (Sumber: Dokumen Penulis).

GPU yang digunakan dalam pengetesan adalah Powercolor RX 470 4GB.

Gambar 3-2: GPU Powercolor RX 470 4GB dengan GPU Riser (Sumber: Dokumen Penulis).

Dengan detail spesifikasi sebagai berikut.



Gambar 3-3: Spesifikasi Teknis Powercolor RX 470 4GB (Sumber: Dokumen Penulis).

Sesuai gambar diatas, mula-mula GPU ini memiliki *memory clock* sebesar 1650 Mhz. *Memory clock* inilah yang nilainya akan kita tingkatkan secara bertahap, untuk memaksimalkan *hashrate* dalam Ethereum Classic Mining.

C. Penerapan algoritma backtrackng

Properti dari algoritma *backtracking* dalam permasalahan memaksimalkan *hashrate* pada Ethereum Classic Mining, dapat didefinisikan sebagai berikut.

1. Solusi Persoalan

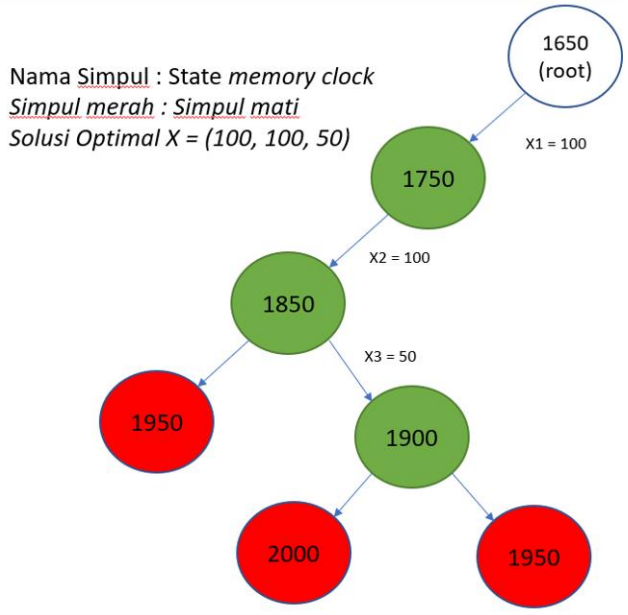
   Merupakan vector atau n-tuple, sebanyak n, dengan n adalah jumlah iterasi peningkatan *memory clock* secara bertahap seperti berikut $X = (x_1,x_2,\ldots,x_n)$.

   Dengan $x_i \in S_i$, dan S_i merupakan besar penambahan frekuensi pada *memory clock*, maka $S_i = \{50, 100\}$.

2. Fungsi Pembatas (*bounding function)*

   Dinyatakan sebagai predikat $B(x_1,x_2,\ldots,x_k)$, dan akan bernilai true jika $X = (x_1,x_2,\ldots,x_k)$ tidak membuat system crash, dan $X = (x_1,x_2,\ldots,x_n)$. $> X_{k-1} = (x_1,x_2,\ldots,x_n)$. atau bisa dibilang jika besar frekuensi *memory clock* pada X_k lebih besar dari pada X_{k-1}, dan X_k tidak membuat *system crash* maka bounding function untuk X_k bernilai true.

Setelah dilakukan pengetesan dengan menerapkan algoritma *backtracking tersebut*, didapat pohon solusi sebagai berikut.



Gambar 3-4: Pohon solusi dari penelusuran permasalahan Pemaksimalan *hashrate* pada Ethereum Classic Mining (Sumber: Dokumen Penulis)

Dari pohon solusi tersebut jika Langkah-langkahnya diuraikan dapat diilustrasikan dengan table berikut.

Simpul Ekspan	Simpul Hidup	Nilai Bounding Function
1650	1750	True
1750	1850	True
1850	1950	True
1950	-	False (*system crash*)
1850	1900	True
1900	2000	True
2000	-	False (*system crash*)
1900	1950	False (*system crash*)
1950	-	False (*system crash*)
1900	-	True

(Tabel 3-1 : Langkah-langkah penelusuran dengan *backtracking*)

Dari hasil percobaan diatas didapatkan solusi optimal X = (100,100,50), yang berarti *memory clock* maksimal yang bisa dicapai sebelum *system* menjadi tidak stabil adalah 1650 + 100 +100+ 50 = 1900Mhz.

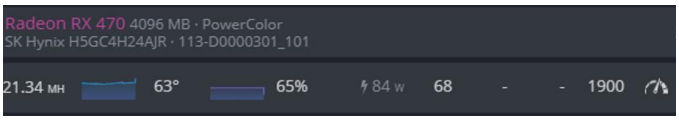
Peningkatan *memory clock* tersebut *membuat* perubahan *hashrate* sebagai berikut.

1. *Hashrate sebelum* (1650mhz)

Gambar 3-5: Hasil *hashrate* pada HiveOS sebelum dioptimisasi. (Sumber:Dokumen penulis)

Dari hasil percobaan didapatkan *hashrate* sebesar 20.65 MH/s (megahash/s), saat GPU belum dioptimisasi *memory clocknya*.

2. *Hashrate sesudah* (1900mhz)



Gambar 3-6: Hasil *hashrate* pada HiveOS setelah dioptimisasi. (Sumber:Dokumen penulis)

Setelah dilakukan optimisasi dengan menaikan *memory clock* dari 1650Mhz menjadi 1900Mhz, *hashrate* pun meningkat menjadi 21.34 MH/s. Peningkatan ini terjadi dengan konsumsi daya yang sama yaitu 84 watt.

Dari hasil diatas didapatkan selisih peningkatan *hashrate* sebesar 0.69 MH/s dengan tingkat konsumsi daya yang sama yaitu 84 Watt.

Peningkatan *hashrate* ini jika dikalkulasikan *profit*nya melalui situs https://minerstat.com/mining-calculator akan memberikan perbedaan sebesar 0.06 USD per harinya, atau sekitar Rp. 870 rupiah perhari, atau Rp. 26.100 per bulannya.

Peningkatan ini memang tidak signifikan, namun disisi lain peningkatan ini tidak menimbulkan *extra cost* lainnya sehingga penulis rasa dapat diterima.

IV. KESIMPULAN

Cryptocurrency adalah suatu jenis *digital asset* yang menganut filosofi desentralisasi, dimana asset ini tidak memerlukan lembaga pemerintah seperti bank sentral, untuk mengeluarkan, mengatur, dan menjaga keberlangsungan system keuangan. Untuk mencapai filosofi tersebut *cryptocurrency* memanfaatkan teknologi *blockchain* dan *cryptography* untuk menyimpan data transaksi, dan mengamankan serta memverifikasi transaksi baru yang terjadi di jaringan tersebut.

Ethereum Classic merupakan cryptocurrency yang diciptakan oleh Vitalik Butterin dan Gavin Wood pada 30 Juli 2015. Pada saat diciptakan pertama kali Ethereum Classic memilki nama hanya Ethereum saja. Ethereum Classic adalah blockchain cryptocurrency pertama yang menunjang teknologi smart contract. Dengan smart contract suatu individu dapat mengirim atau menerima cryptocurrency jika memenuhi kejadian atau event-event tertentu. Teknologi ini menjadi dasar dibuatnya DApps (decentralized apps) dan DeFi (decentralized finance).

Mining merupakan suatu proses dimana dengan menggunakan kekuatan komputasi (biasanya GPU) untuk menyelesaikan persoalan matematis untuk memverifikasi suatu transaksi pada *blockchain cryptocurrency*. Proses mining ini memberikan insentif hadiah berupa *cryptocurrency* tersebut pada subjek atau *Miner* yang melakukan *Mining.*

Proses Mining pada Ethereum Classic menggunakan algoritma Etchash, yang sangat bergantung pada *memory clock* GPU. Dengan menerapkan algoritma *backtracking* kita dapat mengoptimalkan *memory clock* maksimum yang dapat dicapai tanpa membuat system menjadi *crash* atau tidak stabil. Peningkatan *memory clock* ini akan meningkatkan *profitability* dari GPU yang kita miliki.

UCAPAN TERIMAKASIH

Puji Syukur kehadirat Allah SWT karena atas berkat dan rahmatNya lah saya dapat menyelesaikan makalah ini dengan baik. Ucapan terima kasih juga saya ucapkan kepada dosen pengajar mata kuliah strategi algoritma, Bapak Ir. Rinaldi Munir. yang dengan tanpa lelah, telah mengajarkan saya dalam mata kuliah ini selama satu semester. Semoga, makalah yang saya buat ini dapat menebar manfaat walaupun hanya sedikit sekalipun.

VIDEO LINK AT YOUTUBE

https://youtu.be/Ia7XzWTmBpQ

REFERENCES

[1] Rinaldi Munir, Slide Kuliah Strategi Algoritma: Algoritma Runut-Balik (backtracking) (bagian 1), Teknik Informatika ITB, 2021.

[2] Satoshi Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System", www.bitcoin.org

[3] Vitalik Butterin, "Ethereum Whitepaper: A Next-Generation Smart Contract and Decentralized Application Platform", https://ethereum.org/en/whitepaper/

[4] "Ethereum Classic: Principles and Philosophy ", https://www.ethereumclassic.org/knowledge/theory

[5] "Ethereum Classic: History and Future Development ", https://www.ethereumclassic.org/knowledge/roadmap

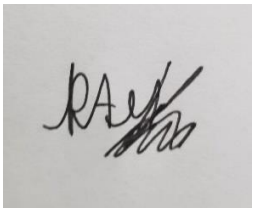
[6] "Cryptocurrency Prices, Charts And Market Capitalizations | CoinMarketCap", www.coinmarketcap.com

[7] "Crypto mining profitability calculator | minerstat ", https://minerstat.com/mining-calculator

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.

Bogor, 10 Mei 2021



Rayhan Asadel (13519196)