Penentuan Jalur Optimal pada Minecraft Any% Speedrun Menggunakan Graf Berarah Berbobot dan Algoritma Dijkstra

Davin Farel Santoso - 13525033
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: davinfarel.santoso@gmail.com , 13525033@std.stei.itb.ac.id

Abstrak*—Minecraft Any% Speedrun merupakan salah satu kategori permainan yang membutuhkan urutan strategi optimal untuk mengalahkan Ender Dragon dalam waktu sesingkat mungkin. Namun, variasi strategi dan faktor keberuntungan dalam pembuatan dunia acuan menimbulkan tantangan tersendiri dalam menentukan jalur progres yang paling efektif. Makalah ini membahas tentang pemodelan alur progres Minecraft Any% Speedrun menggunakan graf berarah berbobot dan pencarian jalur kritis dengan Algoritma Dijkstra. Data durasi waktu nyata (real-time) diambil dari lima sampel permainan rasional pada papan peringkat resmi untuk meminimalkan bias faktor keberuntungan ekstrem serta menghitung waktu aktivitas pasif strategis pemain. Didapatkan rute progres paling optimal dengan urutan tahapan tertentu beserta total estimasi waktu penyelesaian minimumnya.

***Kata Kunci*—Algoritma Dijkstra, graf berarah berbobot, jalur terpendek, Minecraft Speedrun, waktu nyata**

I. Pendahuluan

Minecraft merupakan salah satu permainan *sandbox* yang memiliki komunitas *speedrun* yang sangat aktif. Dalam kategori *Any%*, pemain dituntut untuk menyelesaikan permainan secepat mungkin dengan mengalahkan Ender Dragon tanpa harus menyelesaikan seluruh konten yang tersedia. Untuk mencapai tujuan tersebut, pemain harus menentukan strategi dan urutan tindakan yang efisien agar waktu penyelesaian dapat diminimalkan.

Dalam praktiknya, terdapat berbagai jalur yang dapat ditempuh oleh pemain selama melakukan *speedrun*. Beberapa pemain memilih mencari *village* untuk memperoleh sumber daya awal, sementara pemain lain memanfaatkan *ruined portal* atau menjelajahi gua untuk mempercepat progres permainan. Perbedaan strategi tersebut menyebabkan variasi waktu penyelesaian yang cukup signifikan antar pemain.

Permasalahan pemilihan strategi dalam Minecraft Any% Speedrun dapat dimodelkan menggunakan graf berarah berbobot. Setiap tahapan penting dalam permainan dapat direpresentasikan sebagai simpul (*vertex*), sedangkan perpindahan antar tahapan direpresentasikan sebagai sisi (*edge*) yang memiliki bobot berupa waktu yang dibutuhkan untuk mencapai tahapan berikutnya. Dengan model tersebut, pencarian jalur penyelesaian yang paling efisien dapat dilakukan menggunakan algoritma pencarian jalur terpendek.

Oleh karena itu, makalah ini membahas pemodelan Minecraft Any% Speedrun menggunakan graf berarah berbobot serta penerapan algoritma Dijkstra untuk menentukan jalur penyelesaian yang optimal berdasarkan data *speedrun* yang dianalisis.

II. Landasan Teori

A. Minecraft Any% Speedrun

Minecraft merupakan permainan *sandbox* yang dikembangkan oleh Mojang Studios dan pertama kali dirilis pada tahun 2011. Permainan ini memungkinkan pemain untuk mengeksplorasi dunia yang dihasilkan secara prosedural (*procedurally generated world*), mengumpulkan sumber daya, membuat peralatan, membangun struktur, serta menghadapi berbagai tantangan dalam permainan. Salah satu tujuan utama Minecraft adalah mengalahkan Ender Dragon yang berada di dimensi End.

Gambar I. Visualisasi Permainan Minecraft
Sumber: https://apps.microsoft.com/detail/9ngqktxr9bzf?hl=id-ID&gl=ID

Seiring berkembangnya komunitas Minecraft, muncul aktivitas yang dikenal sebagai *speedrunning*. Speedrun adalah

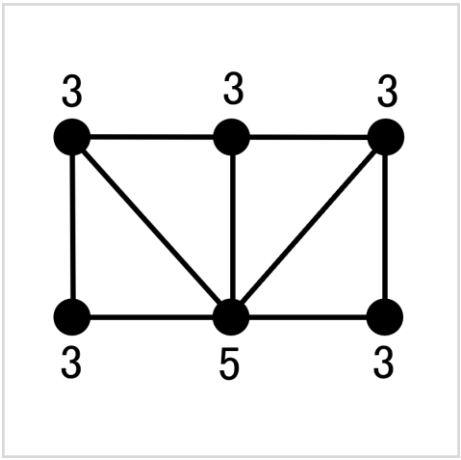
kegiatan menyelesaikan suatu permainan dalam waktu sesingkat mungkin dengan mengikuti aturan kategori tertentu. Dalam komunitas speedrun, waktu penyelesaian setiap pemain dicatat dan dibandingkan dalam papan peringkat (*leaderboard*) sehingga pemain terus berusaha menemukan strategi yang lebih cepat dan efisien.

Salah satu kategori speedrun yang paling populer pada Minecraft adalah Any% Random Seed Glitchless. Pada kategori ini, pemain hanya diwajibkan menyelesaikan permainan secepat mungkin dengan mengalahkan Ender Dragon tanpa harus menyelesaikan seluruh konten permainan. Selain itu, dunia permainan yang digunakan bersifat acak (*random seed*) sehingga setiap percobaan memiliki kondisi awal yang berbeda. Oleh karena itu, pemain harus mampu menentukan strategi yang paling efektif berdasarkan sumber daya dan struktur yang ditemukan selama permainan berlangsung [4].

Dalam proses speedrun, pemain biasanya melewati beberapa tahapan penting seperti memperoleh sumber daya awal, memasuki Nether, mengumpulkan Blaze Rod, memperoleh Ender Pearl, menemukan Stronghold, memasuki End Portal, dan akhirnya mengalahkan Ender Dragon. Setiap tahapan tersebut dapat direpresentasikan sebagai bagian dari suatu jalur penyelesaian yang dapat dianalisis menggunakan teori graf.

B. *Graf Berarah Berbobot*

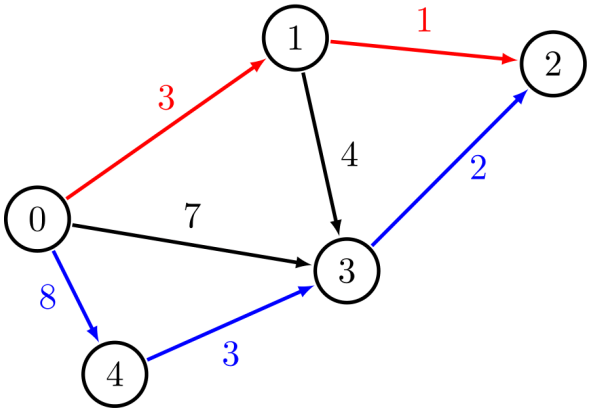
Graf merupakan salah satu struktur dasar dalam Matematika Diskrit yang digunakan untuk merepresentasikan hubungan antara objek-objek tertentu. Secara umum, graf terdiri atas himpunan simpul (*vertex*) yang mewakili objek dan himpunan sisi (*edge*) yang menunjukkan hubungan antar objek tersebut. Graf banyak digunakan dalam berbagai bidang, seperti jaringan komputer, sistem transportasi, media sosial, serta berbagai permasalahan optimasi karena mampu menyederhanakan hubungan yang kompleks ke dalam bentuk yang lebih mudah dianalisis.



Gambar II. Contoh Graf Sederhana
Sumber: https://mathcyber1997.com/materi-soal-dasar-graf-dan-terminologi/

Berdasarkan arah sisinya, graf dapat dibedakan menjadi graf tak berarah (*undirected graph*) dan graf berarah (*directed graph*). Pada graf berarah, setiap sisi memiliki orientasi atau arah tertentu dari suatu simpul menuju simpul lainnya. Jika terdapat sisi dari simpul u ke simpul v, maka hubungan tersebut belum tentu berlaku sebaliknya. Karakteristik ini membuat graf berarah cocok digunakan untuk memodelkan suatu proses yang memiliki urutan atau alur tertentu.

Selain memiliki arah, suatu graf juga dapat memiliki bobot (*weight*) pada setiap sisi yang menghubungkan dua simpul. Graf yang memiliki bobot pada setiap sisinya disebut sebagai graf berbobot (*weighted graph*). Bobot tersebut dapat merepresentasikan berbagai besaran, seperti jarak, biaya, kapasitas, maupun waktu yang diperlukan untuk berpindah dari satu simpul ke simpul lainnya. Dengan adanya bobot, graf tidak hanya menggambarkan hubungan antar objek, tetapi juga memberikan informasi kuantitatif yang dapat digunakan dalam proses analisis.



Ganbar III. Contoh Graf Berarah Berbobot
Sumber: https://hyperskill.org/learn/step/5645

Dalam penelitian ini, digunakan graf berarah berbobot untuk memodelkan proses penyelesaian Minecraft Any% Speedrun. Setiap simpul merepresentasikan tahapan-tahapan penting yang dilalui pemain selama permainan, sedangkan setiap sisi merepresentasikan perpindahan dari satu tahapan ke tahapan berikutnya. Bobot pada setiap sisi menyatakan estimasi waktu yang dibutuhkan pemain untuk melakukan perpindahan tersebut berdasarkan data speedrun yang dianalisis. Dengan model ini, proses speedrun dapat direpresentasikan sebagai suatu jaringan tahapan yang memungkinkan pencarian jalur penyelesaian paling optimal menggunakan algoritma Dijkstra.

C. *Algoritma Dijkstra*

Algoritma Dijkstra merupakan algoritma pencarian jalur terpendek (*shortest path algorithm*) yang diperkenalkan oleh Edsger W. Dijkstra pada tahun 1956 dan dipublikasikan pada

tahun 1959 [3]. Algoritma ini dirancang untuk menentukan lintasan dengan total bobot minimum dari suatu simpul sumber menuju simpul tujuan pada graf yang memiliki bobot sisi bernilai non-negatif.

Prinsip utama algoritma Dijkstra adalah memilih simpul yang memiliki jarak sementara terkecil dari simpul sumber, kemudian memperbarui jarak ke simpul-simpul tetangganya. Proses tersebut dilakukan secara berulang hingga seluruh simpul yang dapat dicapai telah diproses.

Secara umum, langkah-langkah Algoritma Dijkstra adalah sebagai berikut:

1. Menentukan simpul awal dan memberikan nilai jarak 0 pada simpul tersebut.
2. Memberikan nilai tak hingga pada simpul lainnya.
3. Memilih simpul yang belum dikunjungi dengan nilai jarak terkecil.
4. Memperbarui jarak ke seluruh simpul tetangga yang terhubung langsung.
5. Menandai simpul tersebut sebagai telah dikunjungi.
6. Mengulangi proses hingga simpul tujuan ditemukan atau seluruh simpul telah diproses.

Keunggulan Algoritma Dijkstra terletak pada kemampuannya menemukan jalur optimal dengan efisien pada graf berbobot non-negatif. Oleh karena itu, algoritma ini banyak digunakan pada sistem navigasi, jaringan komputer, pencarian rute transportasi, serta berbagai permasalahan optimasi lainnya.

Dalam penelitian ini, Algoritma Dijkstra digunakan untuk menentukan jalur penyelesaian Minecraft Any% Speedrun yang memiliki total waktu minimum berdasarkan graf berarah berbobot yang dibentuk dari data speedrun. Bobot setiap sisi merepresentasikan rata-rata waktu yang diperlukan untuk berpindah antar tahapan permainan. Jalur yang dihasilkan oleh algoritma kemudian dianggap sebagai jalur penyelesaian yang paling optimal berdasarkan data yang dianalisis.

III. METODE PERHITUNGAN

A. *Formulasi Simpul Graf Berarah Berbobot*

Alur permainan *Minecraft Any% Speedrun* dimodelkan ke dalam sebuah graf berarah berbobot G = (V, E). Tahapan-tahapan penting (*milestones*) yang harus dilewati oleh pemain dari awal permainan hingga naga (*Ender Dragon*) dikalahkan didefinisikan sebagai himpunan simpul (V).

Berdasarkan variasi strategi dari data yang dikumpulkan, ditentukan 10 simpul utama sebagai berikut:

Simpul	Keterangan
S1	Spawn (Titik awal permainan dimulai)
S2	Shipwreck (Penjelajahan kapal karam untuk suplai awal)
S3	Village (Desa untuk mencari makanan, besi, atau tempat tidur)
S4	Nether (Dimensi Nether setelah portal berhasil dibuat/aktif)
S5	*Bastion Remnant* (untuk melakukan *bartering* emas dengan *Piglin*)
S6	*Ender Pearl* (pertama kali ditemukan)
S7	*Nether Fortress* (untuk berburu *Blaze Rod*)
S8	Blaze Rod (pertama kali ditemukan)
S9	*Stronghold* (struktur tempat portal menuju dimensi *The End)*
S10	*Ender Dragon* (Dimensi *The End* hingga naga berhasil dikalahkan)

Tabel I. Keterangan Simpul pada Graf Berarah Berbobot

B. *Batasan Pengambilan Data*

- Kriteria Rasionalitas Data Leaderboard

  Data mentah yang digunakan sebagai acuan fungsi waktu tetap bersumber dari catatan waktu terbaik pada papan peringkat resmi (*speedrun leaderboard*). Meskipun demikian, dilakukan penyaringan ketat dengan hanya mengambil data dari *run* yang dikategorikan sebagai "*rational run*". *Rational run* didefinisikan sebagai runtutan permainan yang memiliki alur progres umum dan tidak didominasi oleh faktor keberuntungan ekstrem.

- Penggunaan Pengukuran *Real-Time*

  Dalam penelitian ini, bobot waktu yang digunakan merupakan waktu nyata (*real-time*). Pembatasan ini diterapkan karena terdapat aktivitas pasif penting yang dilakukan oleh pemain yang sering kali tidak terhitung di dalam IGT, seperti berhenti sejenak untuk membaca koordinat (F3 *screen analysis*), melakukan triangulasi arah struktur (*pie chart parsing*), atau merencanakan rute internal. Aktivitas-aktivitas tersebut tetap dikategorikan sebagai bagian dari strategi aktif pemain yang berkontribusi langsung terhadap progres penyelesaian permainan.

C. *Data Mentah*

Data mentah durasi waktu dalam satuan sekon (s) diekstraksi dari 5 *run* pilihan yang memenuhi kriteria rasionalitas. Pencatatan dilakukan pada setiap transisi langsung dari satu simpul ke simpul berikutnya secara *real-time*. Data lengkap dari kelima *run* tersebut disajikan pada Tabel II.

Transisi Antar Simpul	Run 1	Run 2	Run 3	Run 4	Run 5
S1 -> S2	40	-	-	-	32
S1 -> S3	-	78	66	80	-
S2 -> S4	158	-	-	-	341
S3 -> S4	-	-	519	256	-
S3 -> S6	-	112	-	-	-
S6 -> S4	-	226	-	-	-
S4 -> S5	68	-	-	-	-
S5 -> S6	175	-	-	-	-
S4 -> S7	-	125	73	-	128
S6 -> S7	101	-	-	-	-
S7 -> S8	33	32	60	48	95
S8 -> S9	229	436	-	-	-
S8 -> S7	-	-	545	451	324
S7 -> S9	-	371	-	443	1395
S9 -> S10	372	55	109	49	172

Tabel II. Data Durasi Perpindahan Antar Simpul pada 5 Run Rasional (dalam sekon)

D. *Penentuan Bobot Sisi Menggunakan Nilai Rata-Rata*

Dalam menentukan jalur penyelesaian yang paling optimal, maka diperlukan bobot waktu yang dapat merepresentasikan semua data yang diambil. Salah satu metode yang dapat digunakan adalah dengan menghitung nilai rata-rata dari seluruh sampel *run* yang valid. Formulasi yang digunakan dalam mencari nilai rata-rata adalah sebagai berikut:

$$w(u,v) = \frac{1}{k}\sum_{i=1}^{k} t_i(u,v) \quad (2)$$

Dengan keterangan sebagai berikut:

- w(u, v) = Bobot final (estimasi waktu) dari simpul u ke simpul v.
- k = Jumlah *run* yang melewati sisi berarah (u, v) secara langsung (di mana k <= 5).
- t[i](u, v) = Durasi waktu nyata pada *run* ke-i saat melakukan transisi dari simpul u ke simpul v.

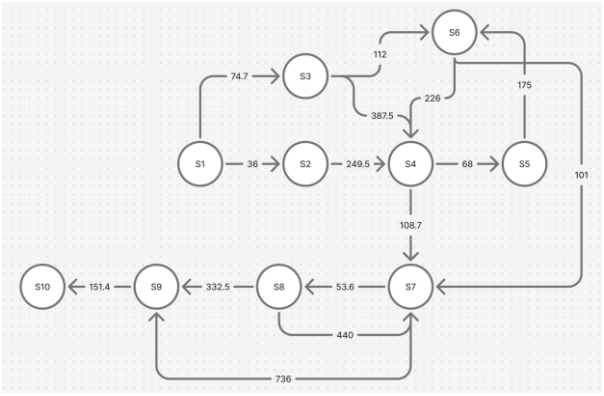
Melalui Formulasi (2) maka didapat hasil bobot final seperti yang telah disajikan pada Tabel III.

Transisi Antar Simpul	Bobot Final (dalam sekon)
S1 -> S2	36
S1 -> S3	74.7
S2 -> S4	249.5
S3 -> S4	387.5
S3 -> S6	112
S6 -> S4	226
S4 -> S5	68
S5 -> S6	175
S4 -> S7	108.7
S6 -> S7	101
S7 -> S8	53.6
S8 -> S9	332.5
S8 -> S7	440
S7 -> S9	736.3
S9 -> S10	151.4

Tabel III. Bobot Durasi Rata-Rata (Final)

E. *Visualisasi Graf Berarah Berbobot*

Berdasarkan simpul-simpul yang telah didefinisikan dan bobot durasi waktu final yang telah diformulasikan, struktur tahapan penting dalam proses penyelesaian permainan dapat divisualisasikan menjadi sebuah graf berarah berbobot, seperti pada Gambar IV.



Gambar IV. Representasi Graf Berarah Berbobot Tahapan Strategi Minecraft Any% Speedrun

Pada Gambar IV, setiap simpul yang dinotasikan dengan S1 hingga S10 menggambarkan tahapan kronologis atau milestone yang harus dicapai oleh seorang pemain. Garis berarah (sisi) menunjukkan pilihan orientasi keputusan strategis yang diambil, sedangkan nilai numerik di atas setiap garis merepresentasikan estimasi beban waktu transisi nyata dalam satuan sekon. Struktur interkoneksi ini memperlihatkan adanya beberapa alternatif rute bercabang yang dapat ditempuh untuk mencapai simpul tujuan akhir (S10).

F. *Penentuan Jalur Optimasi Penyelesaian Permainan*

Untuk mengetahui rute progres Minecraft Any% Speedrun yang paling efektif dan efisien, dilakukan pencarian jalur kritis menggunakan bantuan Algoritma Dijkstra. Berdasarkan visualisasi graf berarah dan bobot rata-rata yang telah dihitung sebelumnya, terdapat beberapa alternatif rute logis yang dapat ditempuh oleh pemain dari titik awal permainan hingga Ender Dragon dikalahkan seperti yang telah disajikan pada Tabel IV.

No.	Rute (dari *Spawn* Hingga *Ender Dragon*)	Hasil Perhitungan Waktu Rata-Rata (dalam sekon)	Total Waktu (dalam sekon)
1.	S1 -> S3 -> S6 -> S7 -> S8 -> S9 -> S10 *(Spawn -> Village -> E-Pearl -> Fortress -> Blaze Rod -> Stronghold -> Ender Dragon)*	74.7 + 112.0 + 101.0 + 53.6 + 332.5 + 151.4	825.2
2.	S1 -> S2 -> S4 -> S7	36.0 + 249.5 +	931.7

	-> S8 -> S9 -> S10 *(Spawn -> Shipwreck -> Nether -> Fortress -> Blaze Rod -> Stronghold -> Ender Dragon)*	108.7 + 53.6 + 332.5 + 151.4	
3.	S1 -> S2 -> S4 -> S5 -> S6 -> S7 -> S8 -> S9 -> S10 *(Spawn -> Shipwreck -> Nether -> Bastion -> E-Pearl -> Fortress -> Blaze Rod -> Stronghold -> Ender Dragon)*	36.0 + 249.5 + 68.0 + 175.0 + 101.0 + 53.6 + 332.5 + 151.4	1167
4.	S1 -> S3 -> S4 -> S7 -> S8 -> S9 -> S10 *(Spawn -> Village -> Nether -> Fortress -> Blaze Rod -> Stronghold -> Ender Dragon)*	74.7 + 387.5 + 108.7 + 53.6 + 332.5 + 151.4	1108.4

Tabel IV. Hasil Perhitungan Rute Alternatif Penyelesaian Permainan

IV. KESIMPULAN

Berdasarkan hasil penelitian mengenai alur progres *Minecraft Any% Speedrun*, pendekatan teori graf berarah berbobot terbukti berhasil merepresentasikan variasi keputusan taktis dan *milestones* penting dalam permainan secara objektif. Penggunaan pembatasan data berbasis *rational run* dari papan peringkat resmi efektif dalam menyaring komponen data mentah, sehingga hasil formulasi rata-rata bobot sisi terbebas dari anomali faktor keberuntungan ekstrem (*extreme RNG*). Hal ini membuat model graf yang dihasilkan dapat menjadi acuan evaluasi rute yang valid dan dapat direplikasi oleh pemain.

Melalui analisis pencarian jalur kritis menggunakan Algoritma Dijkstra pada model graf tersebut, rute S1 -> S3 -> S6 -> S7 -> S8 -> S9 -> S10 *(Spawn -> Village -> E-Pearl -> Fortress -> Blaze Rod -> Stronghold -> Ender Dragon)* secara konsisten terpilih sebagai rute yang paling efektif dan efisien. Rute ini menghasilkan akumulasi estimasi waktu nyata (*real-time*) paling minimum, yaitu sebesar 825.2 sekon atau setara dengan 13 menit 45 detik. Hasil perhitungan matematis tersebut mengungguli tiga alternatif kombinasi rute progres logis lainnya yang diuji dalam penelitian ini.

Secara teoritis, keunggulan rute utama ini menegaskan bahwa pemilihan urutan strategi memegang peranan yang sangat krusial dalam efisiensi sebuah *run*. Keputusan taktis untuk mengamankan kebutuhan logistik awal di desa (*Village*) serta mengumpulkan *Ender Pearl* yang dicicil sejak awal permainan terbukti memberikan efisiensi waktu yang jauh lebih signifikan. Sebaliknya, rute alternatif yang langsung memaksakan masuk ke dimensi *Nether* tanpa persiapan matang atau rute yang terlalu memutar melewati *Bastion* justru memicu pembengkakan akumulasi waktu nyata yang cukup besar pada keseluruhan progres *speedrun*.

REFERENCES

[1] Munir, Rinaldi. 2020. "Teori Graf (Bagian 1)". https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2025-2026/20-Graf-Bagian1-2026.pdf, diakses 18 Juni 2026.

[2] Munir, Rinaldi. 2020. "Teori Graf (Bagian 2)". https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2025-2026/21-Graf-Bagian2-2026.pdf, diakses 18 Juni 2026.

[3] Wolfram Research, "Graph." https://mathworld.wolfram.com/Graph.html, diakses 18 Juni 2026.

[4] Minecraft Wiki, "Tutorial: Speedrun." https://minecraft.wiki/w/Tutorial:Speedrun, diakses 19 Juni 2026.

[5] Minecraft Wiki, "Speedrun." https://minecraft.wiki/w/Speedrun, diakses 17 Juni 2026.

[6] Speedrun.com, "Minecraft Leaderboards." https://www.speedrun.com/mc, diakses 17 Juni 2026.

[7] Wolfram Research, "Directed Graph." https://mathworld.wolfram.com/DirectedGraph.html, diakses 18 Juni 2026.

[8] NetworkX Developers, "Dijkstra's Algorithm." https://networkx.org/documentation/latest/reference/algorithms/shortest_paths/dijkstra.html, diakses 18 Juni 2026.

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, 1 Juni 2025

Davin Farel Santoso | 13525033