ISSN (Online) 2747-0563
Volume 6 Tahun 2026

Perancangan Topologi Level Terstruktur Permainan 2D Platformer Menggunakan Metode *Graph Grammar*

Akmal Zidan Fahreza[1], Chrystia Aji Putra[2*], Yisti Vita Via[3]

[1,2,3] Informatika, Universitas Pembangunan Nasional "Veteran" Jawa Timur
[1]22081010163@student.upnjatim.ac.id
[3] yistivia.if@upnjatim.ac.id

*Corresponding author email: ajiputra@upnjatim.ac.id

***Abstrak*— Metode klasik dalam *Procedural Content Generation* sering kali menghasilkan struktur level yang terlalu acak sehingga sulit untun dikendalikan melalui batas tertentu. Penelitian ini mengusulkan untuk menggunakan metode *graph grammar* dalam membangkitkan struktur topologi level yang tidak hanya dihasilkan secara otomatis tetapi juga sepenuhnya terkendali dan terstruktur dalam batas tertentu. Dengan pendekatan ini, level permainan akan terlebih dahulu dibuat dalam bentuk graf berlabel yang kemudian bertransformasi melalui eksekusi aturan produksi. Aturan produksi mencakup *Linear Expansion*, *Branching*, dan *Key-Lock Mechanism*. Dalam melakukan eksekusi aturan, algoritma akan mendistribusikan satu aturan tiap iterasi dengan probabilitas yang proporsional (60% *linear expansion*, 30% *branching*, 10% *key-lock*) yang kemudian menghasilkan sebuah graf kompleks berisi 14 simpul dalam 10x iterasi. Melalui beberapa pengamatan, sistem terbukti dapat menghasilkan struktur level yang tidak hanya terstruktur tapi juga mampu menyisipkan teka-teki pintu dan kunci melalui penerapan aturan *Key-Lock Mechanism*. sistem juga mampu mempertahankan koneksi jalur utama permainan selama proses transformasi graf berlangsung.**

Kata Kunci— ***Procedural Content Generation*, Level, Graf, *Graph Grammar*, Topologi.**

I. PENDAHULUAN

Procedural Content Generation (PCG) adalah sebuah metode komputasi dimana sebuah konten dihasilkan secara otomatis oleh sebuah komputasi algoritma[1]. Metode ini mempermudah pengembang permainan dalam membuat konten dengan nilai variasi yang cukup tinggi secara otomatis[2]. Metode PCG dapat dimanfaatkan dalam beragam aspek seperti level, karakter, dan item [3]. Terlepas dari keunggulan PCG yang dijelaskan sebelumnya, sebuah survei yang pada referensi [4] menjelaskan bahwa metode PCG klasik seperti Perlin Noise dan Random Walk cenderung menghasilkan konten yang tidak berstruktur dan sarat akan batasan struktural. Tingkat stokastik yang tinggi pada metode tersebut menyebabkan pengembang tidak dapat mengetahui level seperti apa yang akan dihasilkan sebelum akhirnya level tersebut selesai dibangkitkan [4].

Berdasarkan masalah tersebut, dibutuhkan sebuah pendekatan berbeda yang mampu menghasilkan level dengan batasan logika tertentu. salah satu metode yang dianggap cocok adalah *graph grammar*, metode ini pertama kali digunakan dalam lingkup game oleh Joris Dorman [5]. *Graph grammar* merupakan pengembangan dari fungsi grammar dan struktur data graf, berbeda dari grammar biasa yang hanya menghasilkan sebuah *string*, *graph grammar* mampu menghasilkan sebuah graf dengan karakteristik tertentu sesuai dengan batasan aturan yang telah ditentukan. Terdapat 2 jenis metode penulisan graf dalam *graph grammar*, yaitu *edge replacement* dan *node replacement* [6].

Dalam metode *graph grammar* terdapat 3 aspek utama yaitu *Left-Hand-Side* (*Precondition*), *Right-Hand-Side* (*Operation*), dan teknik penggabungan. *Left-Hand-Side* atau bisa disebut juga LHS adalah prekondisi atau syarat yang harus dipenuhi sebuah *simpul* sebelum akhirnya dilakukan operasi *Right-Hand-Side* atau RHS [6]. Proses ini akan terus dilakukan hingga mencapai batas tertentu seperti jumlah iterasi yang telah dilakukan. Hal ini efektif untuk membuat proses pembangkitan level yang tidak hanya acak tapi juga terkendali secara struktur dan batasan tertentu.

Maka dari itu tujuan dari penelitian adalah untuk merancang sebuah arsitektur topologi level menggunakan metode *graph grammar* yang dilakukan secara prosedural. Perancangan ini berfokus pada pengimplementasian aturan-aturan berbasis LHS dan RHS ke dalam proses transformasi graf. Perancangan ini diharapkan dapat menjadi solusi dalam mengatasi keterbatasan kontrol pada metode klasik PCG, sehingga metode ini dapat menghasilkan level yang tidak hanya otomatis dan bervariasi, tapi juga terkendali secara struktur dan batasan logika tertentu.

II. KAJIAN TEORI

A. *Procedural Content Generation*

Procedural Content Generation adalah sebuah metode dalam pengembangan game yang berfungsi untuk membuat atau membangun konten game secara otomatis dan acak [7]. kemampuannya metode ini dapat digunakan dalam berbagai aspek dalam lingkungan permainan, seperti tatanan level, item, misi, music, kendaraan hingga karakter [8]. Tingkat fleksibilitas ini membuat PCG menjadi metode yang banyak digunakan dalam game populer seperti Minecraft dan Spelunky [9].

B. *Labelled Graph*

Graf berlabel adalah sebuah struktur graf yang terdapat fungsi pelabelan dalam tiap simpul, pelabelan ini berfungsi untuk memberikan identitas sebagai penanda atau pembeda peran dari tiap simpul. Label yang digunakan bisa berupa alfabet, angka

ISSN (Online) 2747-0563
Volume 6 Tahun 2026

atau himpunan label lain. Graf dengan simpul yang berlabel didefinisikan sebagai:

$$G = (V, E, \sum, \ell) \quad 1$$

Dimana,

- V = himpunan verteks/*simpul*.
- E = himpunan sisi/*edge*.
- ∑ = himpunan label yang terbatas.
- ℓ (ell) = fungsi pelabelan total.

Rozenberg dalam referensi [10] menegaskan bahwa dalam proses transformasi graf, label pada tiap *simpul* akan berperan penting saat pemetaan berlangsung, *production aturan* hanya akan diterapkan jika label pada *simpul* target sesuai dengan label yang dicari.

C. Graph grammar

Mekanisme dasar dalam metode *graph grammar* adalah melakukan penulisan ulang pada graf induk, pada dasarnya terdapat dua kategori penulisan graf dalam *graph grammar*, yaitu *node replacement* dan *edge replacement* [11]. Proses dimulai dengan menginisiasi sebuah graf sederhana yang nantinya akan dikenai perlakuan, graf ini disebut dengan graf induk, lalu sistem akan terus melakukan penulisan ulang pada graf ini hingga menjadi bentuk yang lebih kompleks [6].

D. Aturan Produksi

Aturan Produksi dalam *graph grammar* berperan sebagai aktor utama pada saat proses transformasi graf. Dalam tiap aturan produksi yang dikembangkan, terdapat dua komponen utama yang saling berinteraksi satu sama lainnya, yaitu:

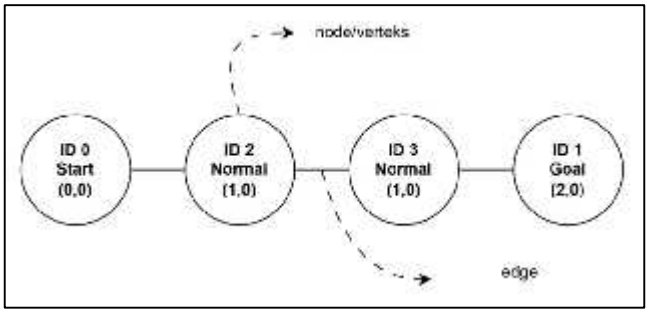
1). *Left-Hand Side* (LHS) atau dapat juga disebut dengan *The Mother* adalah sebuah pola atau syarat yang dicari dalam sebuah graf. Dalam prosesnya, sistem tidak akan melakukan operasi transformasi graf jika pola atau syarat dalam LHS tidak terpenuhi.

2). Right-Hand Side (RHS) atau dapat juga disebut dengan *The Daughter* adalah pola pengganti untuk LHS (*The Mother*). Saat LHS dalam sebuah aturan produksi telah ditemukan, sistem akan berinteraksi dengan RHS untuk melakukan transformasi graf pada pola yang ditemukan [6].

III. METODE PENELITIAN

A. *Inisiasi Struktur Data Graf*

Hasil akhir yang diharapkan pada penelitian ini adalah sebuah struktur graf berlabel tak berarah yang nantinya dapat digunakan sebagai *blueprint* dalam pembangkitan level permainan secara fisik. Dalam sebuah graf terdapat 2 komponen utama yaitu simpul *(node)* yang digambarkan dengan bentuk lingkaran dan (sisi) *edge* yaitu garis yang menghubungkan tiap lingkaran tersebut. Simpul dalam graf akan merepresentasikan satu ruangan fisik pada permainan, sedangkan sisi merepresentasikan hubungan tiap ruangan. Contoh struktur graf yang akan dikembangkan pada penelitian ini akan ditunjukkan pada Gbr 1.



Gbr 1. Contoh Struktur Graf

Dalam Gbr 1. ditampilkan struktur graf sederhana yang pada tiap simpulnya memiliki 3 atribut meliputi: (1) ID sebagai identitas utama simpul yang diambil dari urutan simpul tersebut dibuat, (2) tipe *node* sebagai identitas peran dari simpul tersebut, (3) koordinat vektor2 sebagai atribut posisi *node* dalam peta *grid* .

B. *Pemetaan Tiap Node atau Simpul*

Dalam penelitian ini tiap simpul V pada graf akan merepresentasikan satu ruang dalam permainan, pemetaan ruang akan dilakukan dengan mencocokkan atribut tipe pada simpul. Setiap tipe simpul menentukan karakteristik atau mekanik ruangan yang berbeda-beda. Pemetaan tipe simpul dengan ruang permainan dijabarkan pada tabel berikut:

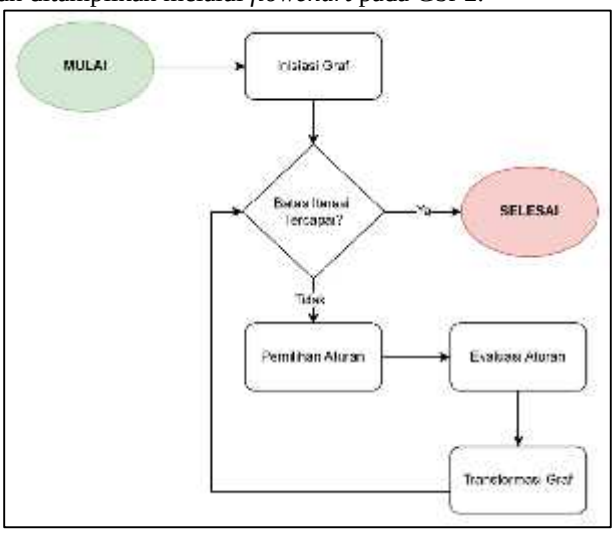
TABEL I
PEMETAAN TIAP SIMPUL

Tipe Simpul	**Nama Ruang**	**Keterangan**
START	Start_Chunk	Ruangan awal pemain saat memulai level
GOAL	Goal_Chunk	Ruangan yang harus dicapai pemain
NORMAL	Normal_Chunk	Ruangan standar berisi platform, musuh, dan tantangan
TREASURE	Treasure_Chunk	Ruangan berisi hadiah yang dapat diambil pemain
KEY	Key_Chunk	Ruangan yang berisi sebuah kunci untuk membuka pintu pada ruangan terkunci
LOCK	Lock_Chunk	Ruangan terkunci yang mengharuskan pemain untuk mencari sebuah kunci terlebih dahulu

ISSN (Online) 2747-0563
Volume 6 Tahun 2026

C. *Tahapan Pembangkitan Struktur Graf*

Proses pembangkitan struktur graf pada penelitian ini dibagi menjadi 4 proses, yaitu inisiasi graf, pemilihan aturan, evaluasi aturan, lalu transformasi graf. Alur sistem pembangkitan graf akan ditampilkan melalui *flowchart* pada Gbr 2.



Gbr 2. Flowchart Pembangkitan Struktur Graf

Berdasarkan alur sistem pada Gbr 2. berikut penjelasan lebih lanjut mengenai proses yang terjadi pada alur sistem di atas:

1) Inisiasi Graf, sistem akan memulai dengan membuat sebuah graf sederhana yang berisikan 2 simpul yang saling terhubung, yaitu simpul *start* dan *goal*.
2) Pemilihan Aturan, setelah graf dasar selesai dibentuk sistem akan melakukan pemilihan satu aturan secara acak dengan probabilitas yang telah ditentukan.
3) Evaluasi Aturan, saat satu aturan spesifik telah terpilih, program akan melakukan evaluasi LHS pada semua simpul yang ada dan memasukkannya dalam suatu himpunan, proses ini bertujuan agar sistem hanya akan memilih simpul yang telah terjamin kesesuaiannya dengan LHS.
4) Transformasi Graf, setelah proses sebelumnya selesai, sistem akan mendapatkan himpunan berisi *simpul* yang telah terjamin kesesuaiannya dengan LHS. Sistem akan mengambil secara acak satu *simpul* dalam himpunan tersebut, lalu menerapkannya dalam proses transformasi graf.

Proses-proses di atas akan terus diulang hingga batas iterasi telah tercapai.

D. *Perancangan Aturan Produksi*

Inti dari sistem ini terletak pada perancangan aturan produksi yang didefinisikan dengan LHS dan RHS, aturan ini akan menjadi otak penggerak saat dilakukan transformasi graf. Pada penelitian ini akan dibuat 3 aturan produksi, yaitu *Linear Expansion*, *Branching*, dan *Key-Lock Mechanism*. Berikut adalah penjelasan mekanisme ketiga aturan tersebut:

1). *Linear Expansion* : Aturan ini memungkinkan sistem untuk melakukan ekspansi level dengan menambahkan *simpul* baru tepat di belakang *simpul* goal. Dengan mekanisme ekspansi tersebut, aturan ini akan membuat struktur graf melebar ke samping dan secara bersamaan memastikan simpul *goal* akan tetap berada diujung level.

2). *Branching* : Aturan ini membuat sebuah variasi pada struktur graf dengan menambahkan satu *simpul* bertipe *treasure* pada salah satu *simpul* bertipe normal. selain itu terdapat prasyarat lain yaitu simpul yang terpilih harus memiliki *degree* < 3 dan tidak ada *simpul* bertipe *treasure* pada salah satunya.

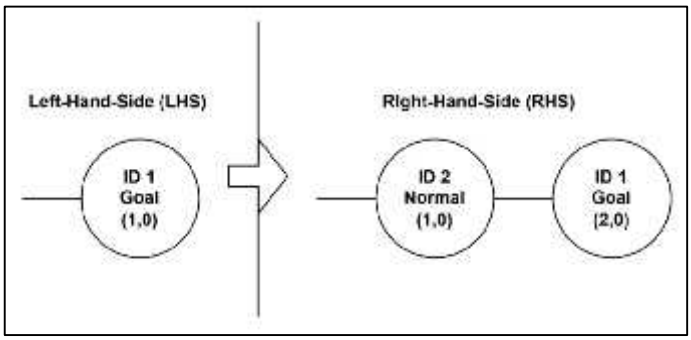
3). *Key-Lock Mechanism* : Aturan ini menambahkan variasi permainan dengan menjamin terdapat teka-teki Kunci dan Pintu pada level. LHS akan mencari dua V-normal yang saling terhubung. lalu RHS akan memutuskan hubungan tersebut untuk menyisipkan satu *simpul* baru dengan tipe *Lock*, dan menambahkan simpul baru bertipe *key* di sekitarnya.

IV. HASIL DAN PEMBAHASAN

A. *Pengembangan Aturan Produksi*

Aturan produksi menjadi inti dari mekanisme pembangkitan level penelitian ini. penulis akan mengembangkan setidaknya 3 jenis *aturan* yang memiliki karakter dan fungsi yang berbeda, yaitu Linear Expansion, Branching, dan Key-Lock Mechanism. berikut hasil pengembangan berupa LHS dan RHS dari tiap *aturan*.

1). *Linear Expansion:* aturan ini akan berperan untuk menambah durasi permainan dengan memperluas struktur graf dengan menambahkan simpul baru ke arah samping. Berikut adalah visualisasi dan uraian lebih lanjut bagaimana aturan *Linear Expansion* bekerja melalui konsep LHS RHS.



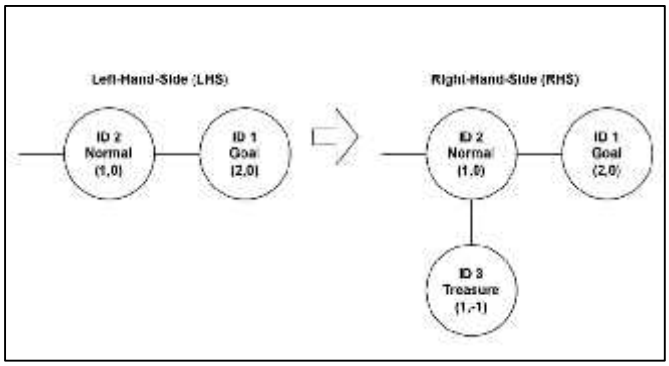
Gbr 3. LHS & RHS *Linear Expansion*

- LHS (*Pre-Condition*) : Terdapat hubungan langsung antara dua simpul yang salah satunya bertipe goal.
- RHS (*Operation*) : (1) sistem akan menghitung nilai D untuk mengetahui ke mana arah simpul baru akan diletakkan. (2) simpul *goal* akan digeser sebesar nilai D yang telah ditemukan. (3) sistem membuat simpul baru bertipe normal dan meletakkan tepat di ruang kosong yang ditinggalkan oleh simpul *goal*. (4) sistem memperbarui hubungan sisi dari simpul baru.

2) *Branching*: Aturan ini berfungsi untuk membuat jalur non-linear untuk meningkatkan variasi permainan, sistem akan membuat percabangan jalur dengan menambahkan simpul bertipe *treasure* yang hanya terhubung pada satu simpul

ISSN (Online) 2747-0563
Volume 6 Tahun 2026

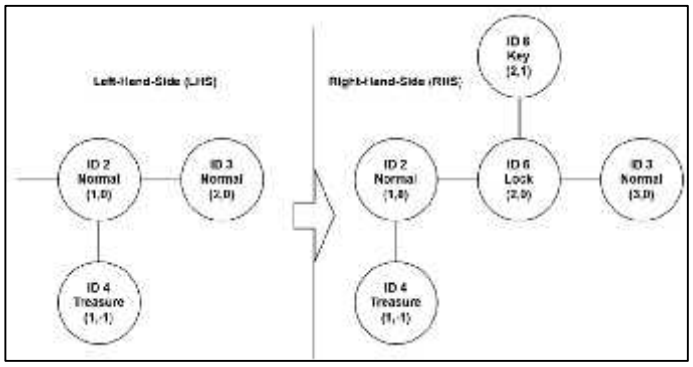
normal. Berikut adalah ilustrasi yang dilanjut dengan penjelasan bagaimana aturan *Branching* bekerja melalui konsep LHS RHS.



Gbr 4. LHS & RHS *Branching*

- LHS (*Pre-Condition*) : terdapat simpul aktif bertipe normal v dengan *degree* < 3, dan di dalamnya tidak terdapat hubungan langsung dengan simpul bertipe *treasure*.
- RHS (*Operation*) : (1) sistem membuat simpul baru bertipe *treasure*. (2) Jika simpul normal memiliki *degree* < 3, maka sistem akan memilih salah satu dari sisi kosong secara acak dengan probabilitas yang sama, namun jika telah memiliki *degree* = 3, maka sistem akan langsung meletakkan simpul *treasure* pada posisi yang tersedia. (3) sistem menghubungkan sisi pada kedua *simpul* yaitu simpul normal dan simpul *treasure*.

3) *Key-Lock Mechanism:* Aturan ini bertujuan untuk meningkatkan kompleksitas level dengan menambahkan mekanisme teka-teki logika sederhana dengan menambahkan *simpul* dalam graf dengan mekanisme kunci dan pintu. Berikut adalah visualisasi dan uraian lebih lanjut aturan *Key-Lock Mechanism* melalui konsep LHS RHS



Gbr 5. LHS & RHS *Key-Lock Mechanism*

- LHS (*Pre-Condition*) : terdapat dua *simpul* bertipe normal (N1, N2) yang terhubung secara langsung.
- RHS (*Operation*) : (1) hubungan pada kedua *simpul* N1 dan N2 diputus. (2) sistem menggeser N2 dan *simpul* lain setelahnya ke arah kanan untuk memberi ruang kosong. (3) sistem membuat simpul baru dengan tipe *Lock* dan menyisipkannya diruang kosong antara N1 dan N2. (4) sistem membuat simpul baru dengan tipe *Key*, lalu menghubungkannya pada sisi kosong simpul *Lock*.

B. *Pembobotan Aturan Produksi*

Ketiga aturan yang dikembangkan kemudian diberikan bobot untuk kemudian digunakan saat pemilihan aturan. Pembobotan ini akan mengambil *range* angka dari 0,0 hingga 1,0. Pembobotan ini berfungsi untuk menghasilkan level yang memiliki konsistensi persebaran aturan di tiap generasinya. Pada tabel II diperlihatkan bagaimana persebaran bobot untuk tiap aturan.

TABEL II
PEMBOBOTAN TIAP ATURAN

No	Aturan	Bobot
1	Linear Expansion	0,0 – 0,6 (60%)
2	Branching	0,6 – 0,9 (30%)
3	Key-Lock Mechanism	0,9 – 1,0 (10%)

Bobot pada tabel II digunakan saat proses pemilihan aturan. Pada proses ini sistem akan menggunakan sebuah fungsi bernama RNG (*Random Number Generation*), RNG akan menghasilkan sebuah nilai acak bertipe *float* dengan rentang 0,0 - 1,0. Nilai tersebut kemudian dicocokkan dengan persebaran angka pada tabel II, hingga akhirnya terpilih satu aturan.

C. *Eksekusi Pembangkitan Struktur Graf*

Pada bagian ini penelitian akan masuk pada tahap implementasi program. Ketiga aturan yang telah dirancang sebelumnya akan diterapkan dalam pembangkitan struktur graf. Proses ini bertujuan untuk menghasilkan struktur graf yang kompleks melalui penerapan aturan-aturan yang telah dirancang pada bab sebelumnya.

1) *Log Eksekusi*

Sebagai bukti bahwa sistem telah berjalan dengan baik saat proses pembangkitan graf, dilakukan sebuah perekaman data melalui debug log pada tiap iterasi. Dalam log eksekusi ini akan menampilkan *value* RNG yang dihasilkan dan tiap aturan yang terpilih pada tiap iterasinya. Berikut adalah tabel berisi log eksekusi pembangkitan graf dengan iterasi sebanyak 10 kali.

TABEL III
LOG EKSEKUSI PEMBANGKITAN GRAF

Iterasi	RNG	*Aturan* Terpilih	Keterangan
1	0.05	*Linear Expansion*	Menambahkan *simpul normal* di belakang goal.
2	0.80	*Branching*	Membuat percabangan simpul *Treasure*.
3	0.28	*Linear Expansion*	Menambahkan *simpul normal* di belakang goal.

ISSN (Online) 2747-0563
Volume 6 Tahun 2026

4	0.46	*Linear Expansion*	Menambahkan *simpul normal* di belakang goal.
5	0.63	*Branching*	Membuat percabangan simpul *Treasure*
6	0.36	*Linear Expansion*	Menambahkan *simpul normal* di belakang goal.
7	0.92	*Key-Lock Mechanism*	Menyisipkan rintangan Gembok dan Kunci.
8	0.45	*Linear Expansion*	Menambahkan *simpul normal* di belakang goal.
9	0.75	*Branching*	Membuat percabangan simpul *Treasure*
10	0.33	*Linear Expansion*	Menambahkan *simpul normal* di belakang goal.

Seperti yang terlihat pada tabel III, sistem secara konsisten mampu menerapkan pembobotan yang telah ditetapkan sebelumnya. Pada rentang 0,0 - 0,6 sistem berhasil menjalankan aturan *Linear Expansion* untuk memperpanjang struktur graf. Lalu pada rentang 0,6 - 0,9 sistem mengeksekusi aturan *Branching* untuk percabangan dengan menambahkan ruang bertipe *treasure*. Dilanjut pada rentang 0,9 - 1,0 sistem berhasil menjalankan aturan *Key-Lock* untuk menyisipkan rintangan pada struktur graf.

2) *Analisis Kinerja Eksekusi Aturan*

Berdasarkan data yang telah disajikan pada tabel II, terlihat bahwa sistem melakukan distribusi aturan dengan proporsi yang sangat baik. Aturan *Linear Expansion* terlihat mendominasi dengan total tereksekusi sebesar 60% (6 dari 10 iterasi), angka ini menunjukkan bahwa struktur level yang dibangun memiliki panjang lintasan yang cukup sebelum menemukan teka-teki pintu dan kunci. Selanjutnya aturan *Branching* yang tereksekusi sebesar 30% (3 dari 10 iterasi) menunjukkan keseimbangan yang cukup dalam memberikan eksplorasi bercabang tanpa mengganggu rute utama. Lalu aturan *Key-Lock* tereksekusi sebesar 10% (1 dari 10 iterasi). Tabel IV akan menyajikan total persebaran aturan yang terjadi.

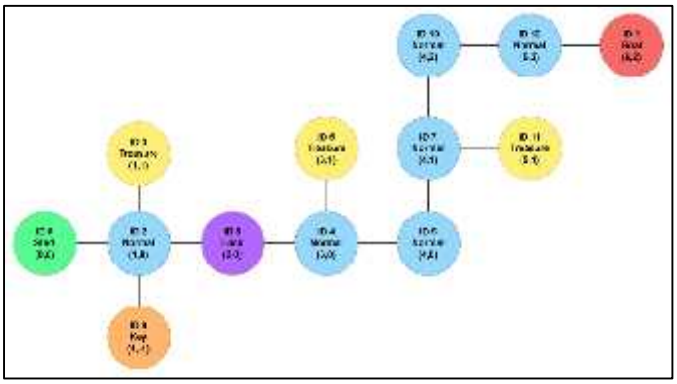
TABEL IV
ANALISIS ITERASI PADA TIAP ATURAN

Aturan Produksi	**Iterasi Ke-**	**Total**
Linear Expansion	1, 3, 4, 6, 8, 10	6
Branching	2, 5, 9	3
Key-Lock	7	1
	Total Iterasi	10

Terlihat pada tabel IV puncak transformasi graf terjadi pada iterasi ke-7 saat RNG menghasilkan nilai 0.92 dan mengeksekusi aturan Key-Lock. Aturan *Key-Lock* sukses dieksekusi karena syarat pada LHS *Key-Lock* yaitu terdapatnya dua simpul bertipe normal yang saling terhubung telah sukses terbentuk pada iterasi-iterasi sebelumnya.

D. *Hasil Akhir Transformasi Graf*

Hasil akhir dalam perancangan ini adalah sebuah struktur graf berlabel yang di dalamnya terdapat *simpul* yang saling terhubung. tiap *simpul* pada graf akan merepresentasikan satu ruangan dalam permainan, sehingga struktur graf secara keseluruhan akan membangun sebuah denah ruangan secara keseluruhan. Gbr 6. akan menampilkan hasil akhir pembangkitan graf yang dijalankan dengan iterasi sebanyak 10 kali.



Gbr 6. Hasil Akhir Transformasi Graf 10 Iterasi

Terlihat pada Gbr 6. bagaimana graf terbentuk dari 14 simpul yang saling terhubung dan terstruktur hingga membentuk susunan level yang utuh dari titik awal hingga akhir. Mengacu pada Gbr 6, terdapat setidaknya 3 poin utama yang dapat diperhatikan guna memvalidasi keberhasilan dalam perancangan:

1) *Konsistensi Jalur Utama*

Sistem berhasil mempertahankan alur utama permainan saat ekspansi graf dilakukan, hal ini dapat dilihat pada *simpul goal* (ID 2) yang digeser saat ekspansi graf hingga berada dipaling ujung struktur graf

2) Distribusi Ruang

Dengan adanya *simpul treasure* (ID 3, ID 6 & ID 11) menunjukkan bahwa algoritma pembangkit graf tidak hanya cenderung melakukan ekspansi graf tapi juga memberikan jalur percabangan untuk menambahkan nilai variasi, hal ini selaras dengan pembobotan aturan yang telah didefinisikan sebelumnya. Jalur percabangan ini ditambahkan langsung dari jalur utama sehingga tidak akan mengganggu jalur utama level.

ISSN (Online) 2747-0563
Volume 6 Tahun 2026

3) Eksekusi Key-Lock

Fokus utama pada pembangkitan struktur ini terletak pada bagaimana sistem menempatkan simpul rintangan pada level. Seperti pada gambar, sistem sukses meletakkan simpul *lock* (ID 8) pada jalur utama *simpul* terakhir. Selain itu sistem juga mampu meletakkan simpul *key* (ID 9) sebagai solusi rintangan di tempat sebelum simpul *lock* (ID 8) berada. hal ini menunjukkan bahwa sistem yang dirancang menggunakan *graph grammar* mampu menyediakan solusi rintangan tepat sebelum pemain menghadapi rintangan tersebut

V. KESIMPULAN

Berdasarkan hasil rancangan dan analisis yang dilakukan, dapat disimpulkan bahwa metode *graph grammar* telah berhasil dalam membangkitkan topologi graf yang terstruktur. Perancangan dan pengimplementasian tiga aturan produksi mulai dari *Linear expansion*, *branching*, hingga *Key-Lock Mechanism* terbukti mampu mengubah struktur graf sederhana yang hanya terdiri dari dua simpul dasar menjadi struktur graf yang lebih kompleks. Struktur yang dihasilkan tidak hanya menjadi kompleks tapi juga memiliki persebaran eksekusi aturan, hal ini terwujud berkat pembobotan probabilitas aturan yang menjaga distribusi aturan sehingga terciptanya struktur graf yang seimbang.

Penelitian ini memperlihatkan bahwa algoritma dapat menghasilkan struktur graf yang memiliki teka-teki pintu dan kunci dengan memanfaatkan metode *Left-Hand-Side* (LHS) dan *Right-Hand-Side* (RHS) pada *graph grammar*. Sistem berhasil melakukan eksekusi aturan Key-Lock yang menambahkan rintangan teka-teki dengan menjamin bahwa simpul kunci akan selalu berada sebelum simpul pintu berada. Pengujian menggunakan iterasi sebanyak 10x telah menghasilkan topologi dengan jumlah 14 simpul yang dapat diselesaikan dari titik awal hingga akhir.

UCAPAN TERIMA KASIH

Penulis mengucapkan banyak terima kasih kepada seluruh pihak yang telah memberikan dukungan dan kontribusi dalam proses penyusunan penelitian ini khususnya kepada pihak SANTIKA sehingga penelitian ini dapat diselesaikan dengan baik.

REFERENSI

[1] D. A. Ramadhan dan A. D. Indriyanti, “Procedural Content Generation pada Game World Exploration Sandbox Menggunakan Alogoritma Perlin Noise,” *Journal of Informatics and Computer Science (JINACS)*, vol. 4, no. 01, hlm. 86–91, Jul 2022, doi: 10.26740/jinacs.v4n01.p86-91.

[2] T. Asrori dan M. Imron, “Model Konseptual Value-Driven Procedural Content Generation (PCG) untuk Pengembangan Game Edukasi Karakter,” 2025.

[3] N. A. Ranggianto, A. P. Segara, D. Wijonarko, A. Andrianto, dan M. Habibullah Arief, “Procedural Content Generation pada Level Gim Sokoban Menggunakan Model Hybrid GPT2 dan,” *Remik: Riset dan E-Jurnal Manajemen Informatika Komputer*, vol. 9, no. 3, 2025, doi: 10.33395/remik.v9i3.15188.

[4] Y. Zhang, G. Zhang, dan X. Huang, “A Survey of Procedural Content Generation for Games,” dalam *Proceedings - 2022 International Conference on Culture-Oriented Science and Technology, CoST 2022*, Institute of Electrical and Electronics Engineers Inc., 2022, hlm. 186–190. doi: 10.1109/CoST57098.2022.00046.

[5] J. Dormans, “Adventures in level design,” dalam *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, New York, NY, USA: ACM, Jun 2010, hlm. 1–8. doi: 10.1145/1814256.1814257.

[6] J. Vijayakumar dan L. Mathew, “On Graph Grammars and Games,” Mar 2024, [Daring]. Tersedia pada: http://arxiv.org/abs/2403.07607

[7] A. Wahyudi, A. I. Pawelloi, dan N. Sirimorok, “Implementasi Procedural Generation Pada Game Aksi Survival,” *Jurnal Informatika dan Teknologi Pendidikan*, vol. 5, no. 2, hlm. 144–151, 2025, doi: 10.59395/jitp.v5i2.142.

[8] A. Vatresia, F. Putra Utama, dan A. Yulianto, “DESIGNING A 3D ROGUELIKE GAME WITH PROCEDURAL CONTENT GENERATION USING THE GRAPH GRAMMARS METHOD,” *Jurnal Teknik Informatika (Jutif)*, vol. 4, no. 6, hlm. 1437–1446, Des 2023, doi: 10.52436/1.jutif.2023.4.6.546.

[9] N. Lee dan J. Morris, “A Procedural Generation Platform to Create Randomized Gaming Maps using 2D Model and Machine Learning,” dalam *Signal Image Processing and Multimedia*, Academy and Industry Research Collaboration Center (AIRCC), Mei 2023, hlm. 147–155. doi: 10.5121/csit.2023.130916.

[10] Grzegorz. Rozenberg, *Handbook of graph grammars and computing by graph transformation*. World Scientific, 1997. Diakses: 15 Desember 2025. [Daring]. Tersedia pada: https://books.google.co.id/books?hl=id&lr=&id=P3r82gfRf8MC&oi=fnd&pg=PA1&dq=graph+grammar&ots=SaC-MKkt4Y&sig=DtFajrXwMYIGyJZJULXSi3pqzKk&redir_esc=y#v=snippet&q=labelled&f=false

[11] S. Cooper dan M. Bazzaz, “A Constraint-Based Graph Grammar Approach Unifying Level and Playthrough Generation,” 2025.