Analisis Sederhana Fitur "Nemesis System" dalam Video Game "Middle-Earth: Shadow of Mordor"

Ahmad Naufal Farhan 13513049
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13513049@std.stei.itb.ac.id

***Abstract*—*Nemesis System* merupakan satu fitur unik yang ditawarkan oleh video game berjudul *Middle-Earth: Shadow of Mordor*. Fitur yang menjadi nilai tambah dari game ini, memungkinkan seluruh lingkungan pemain untuk dapat merespon sesuai dengan apa yang dilakukan oleh karakter pemain dalam game. Pada aspek AI, *Nemesis System* tidak hanya membuat AI yang cerdik, tetapi juga membuat AI yang memiliki "memori". Sebagai contoh, ketika pemain mengalahkan seorang musuh tanpa membunuhnya, maka di pertemuan selanjutnya dengan pemain, musuh tersebut akan melarikan diri. Tidak hanya berdampak pada satu musuh itu, lingkungan sekitar musuh tersebut akan terkena dampaknya juga. Pada makalah ini, penulis akan menjelaskan dan mengilustrasikan bagaimana *Nemesis System* ini bekerja, dengan menggunakan teori-teori pada Matematika Diskrit.**

Keywords—Middle-Earth: Shadow of Mordor, Nemesis, Nemesis System, Artificial Intelligence, graf, pohon, kombinatorial

I. PENDAHULUAN

Seiring dengan kemajuan dan perkembangan teknologi informasi dalam kehidupan manusia, berbagai macam perangkat lunak (*software*) pun berkembang juga fungsinya. Para produsen perangkat lunak mulai berlomba-lomba untuk menawarkan berbagai macam fitur-fitur unik yang diusung perangkat lunaknya, untuk memenuhi kebutuhan dan kepuasan para konsumennya. Tak terkecuali pada industri *video games*.

Dalam konteks *video game*, ada beberapa hal yang memengaruhi kepuasan pemain dalam memainkan *video game*, diantaranya:

1. *Gameplay*
   Gameplay yang unik dan menarik dapat membuat pemainnya memiliki rasa ketertarikan untuk terus memainkannya. Keunikan gameplay dapat ditinjau dari kepintaran *Artificial Intelligence* (AI), lingkungan permainan yang menarik, cerita atau *storyline* yang menarik untuk diikuti, atau antarmuka pengguna pada saat jalannya permainan.
2. *Replay Value*
   Replay value dapat didefinisikan sebagai nilai kepuasan yang diperoleh dalam memainkan *video game* tersebut beberapa kali. Dapat didefinisikan juga sebagai "sampai berapa lama pemain akan merasa bosan memainkan game tersebut".
3. *Game Display Graphics*
   Tampilan grafik yang memukau dari sebuah *video game* akan memanjakan mata pemain, sehingga pemain merasa nyaman untuk memainkannya.

Produsen *video game* berusaha untuk memajukan ketiga aspek tersebut, demi meraih perhatian pemain. Salah satunya adalah *video game* dengan judul *Middle-Earth: Shadow of Mordor* yang dikembangkan oleh *Monolith* ini. Salah satu keunikan yang dipasarkan oleh *video game* ini adalah fitur Nemesis System, yang dikatakan akan memberikan pengalaman bermain dengan AI yang benar-benar baru dan "*fresh*".

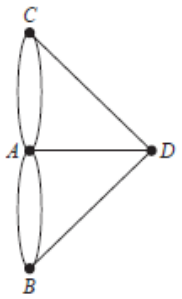
Fitur Nemesis System ini dipilih penulis karena setelah memainkan *video game* ini, penulis menemukan suatu relasi antar unsur pada Nemesis System ini, yang ternyata bisa dijelaskan menggunakan teori-teori yang dipelajari pada kuliah Matematika Diskrit ini.

II. TEORI TERKAIT

A. Graf

Graf adalah salah satu cabang topik dari matematika diskrit, yang biasanya digunakan untuk menggambarkan objek-objek diskrit dan hubungan antar objek-objek tersebut. Pada representasi graf, objek diskrit direpresentasikan oleh simpul/*vertex* dan hubungannya oleh sisi/*edge*.

Penggunaan teori graf diperkenalkan pertama kali oleh Leonhard Euler pada penyelesaiannya untuk permasalahan Jembatan Königsberg[1]. Ia memodelkan jembatan menjadi sisi dan tempat yang dihubungkan oleh jembatan tersebut menjadi simpulnya.



***Gambar 2-1**: Representasi Graf untuk Jembatan Konigsberg*

Sebuah graf G, didefinisikan sebagai satu pasangan himpunan simpul (*V*) dan sisi (*E*), atau dapat didefinisikan dengan notasi *G = (V, E)*.

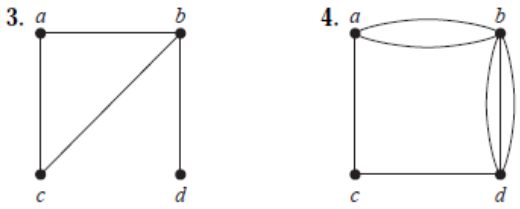
- *V* = { v_1, v_2, v_3, ... , v_n } adalah himpunan tidak kosong dari simpul-simpul.
- *E* = { e_1, e_2, e_3, ... , e_n } adalah himpunan sisi yang menghubungkan sepasang simpul.

Sebagai contoh, graf Jembatan Konigsberg pada Gambar 2-1 dapat ditulis dalam notasi G = *(V,E)*, dimana:

- *V* = { *A, B, C, D* }, dan
- *E* = {(*C,D*), (*A,C*), (*A,C*), (*A,B*), (*A,B*), (*A,D*), (*B,D*) }

Berdasarkan ada tidaknya gelang atau sisi ganda, graf dapat dikelompokkan menjadi dua, yaitu graf sederhana dan graf tidak-sederhana. Graf sederhana merupakan graf yang tidak memiliki gelang dan sisi ganda. Sedangkan graf tidak-sederhana adalah graf yang memiliki gelang atau sisi ganda, atau keduanya.

- Suatu graf dikatakan memiliki gelang apabila ada satu sisi yang menghubungan suatu simpul dengan simpul itu sendiri, dan
- Suatu graf dikatakan memiliki sisi ganda apabila ada sepasang simpul berdekatan yang dihubungkan oleh dua atau lebih sisi.



***Gambar 2-2**: Graf Sederhana dan Graf Tidak-Sederhana [1]*

Sedangkan berdasarkan ada tidaknya arah pada sisi, graf dikelompokan menjadi dua, yaitu graf berarah dan graf tidak berarah.

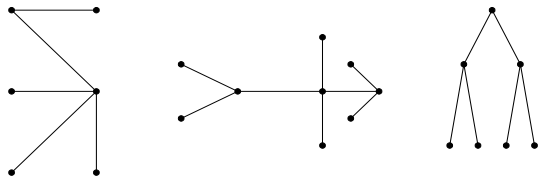
Teori graf memiliki beberapa terminologi[2], diantaranya:

1. Bertetangga (*adjacent*)
   Dua simpul dikatakan bertetangga bila keduanya dihubungkan langsung dengan sebuah sisi.
2. Bersisian (*incident*)
   Sebuah sisi dikatakan bersisian dengan 2 simpul jika kedua simpul dihubungkan oleh sisi tersebut.
3. Simpul terpencil (*isolated graph*)
   Simpul terpencil adalah simpul yang tidak mempunyai sisi yang menghubungkannya.
4. Graf kosong (*null graph* atau *empty graph*)
   Graf kosong adalah graf yang himpunan sisinya merupakan himpunan kosong.
5. Derajat (*degree*)
   Derajat suatu simpul pada graf tak-berarah adalah jumlah sisi-sisi yang bersisian dengan simpul tersebut.
6. Lintasan (*path*)
   Lintasan yang panjangnya *n* dari simpul awal v_0 ke simpul tujuan v_n di dalam graf *G* ialah barisan berselang-seling simpul-simpul dan sisi-sisi yang berbentuk v_0, e_1, v_1, e_2, v_2,... , v_{n-1}, e_n, v_n sedemikian sehingga e_1, e_2, ... , e_n adalah sisi-sisi dari graf *G*.
7. Sirkuit (*circuit*)
   Sirkuit adalah lintasan yang berawal dan berakhir pada simpul yang sama.
8. Upagraf (*subgraph*)
   Upagraf adalah bagian dari graf, yang berarti jika terdapat graf *G* = (*V,E*) maka terdapat upagraf G_1 = (V_1,E_1) dimana V_1 himpunan bagian dari *V*, dan E_1 himpunan bagian dari *E*.
9. Terhubung (*connected*)
   G = (*V,E*) disebut graf terhubung (*connected graph*) jika untuk setiap pasang simpul v_i dan v_j dalam himpunan *V*, terdapat lintasan dari v_i ke v_j.
10. Graf berbobot (*weighted graph*)
    Graf berbobot adalah graf yang setiap sisinya diberi sebuah harga/bobot.

B. Pohon

Pohon merupakan graf tidak berarah, yang terhubung (*connected graph*) dan tidak mengandung sirkuit di dalamnya. Pohon dapat disebut juga sebagai graf sederhana, karena pohon tidak dapat memiliki sisi ganda maupun gelang. Karena pohon adalah graf, maka pohon dapat dinotasikan sebagai *G* = (*V,E*), dengan G adalah sebuah pohon, dan beberapa terminologi graf juga berlaku pada pohon..

Pohon berakar (*rooted tree*) adalah pohon yang satu simpulnya diperlakukan sebagai akar, dan setiap sisinya diberi arah yang menjauhi akar tersebut. Pohon berakar merupakan pohon yang paling sering digunakan dalam implementasi pohon di dunia nyata. Sebagai contoh, untuk menggambarkan pohon keputusan, pencarian menggunakan *Binary Search Tree*, *Huffman coding*, silsilah keluarga, dan masih banyak lagi.



Gambar 2-3*: Beberapa Contoh Pohon. Pohon ketiga dari kiri adalah pohon berakar. [2]*

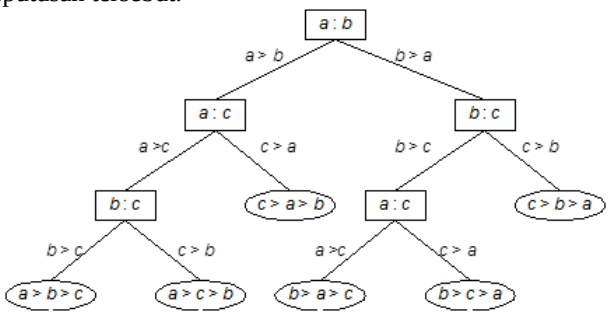
1. Pohon Merentang (*spanning tree*)

Pohon merentang adalah upagraf dari sebuah graf *G* yang merupakan pohon. Pohon merentang dapat diperoleh dengan menghilangkan sirkuit dari graf. Sebuah graf dapat memiliki lebih dari satu pohon merentang

Salah satu aplikasi pohon merentang adalah mencari jarak minimum dari sebuah rute beberapa kota sehingga setiap kota tetap terhubung satu sama lain. Dimisalkan rute kota-kota tersebut digambarkan dengan sebuah graf tidak berarah-berbobot G, maka kita harus mencari pohon merentang yang memiliki bobot seminimal mungkin, yang disebut sebagai Pohon Merentang Minimum (*Minimum Spanning Tree*)[2].

2. Pohon Keputusan (*decision tree*)

Pohon keputusan digambarkan dalam pohon berakar, dengan setiap akarnya menggambarkan suatu keputusan, dan sisi yang menuju ke sebuah akar lainnya menggambarkan kondisi yang berkorespondensi dengan keputusan tersebut.



Gambar 2-4*: Pohon Keputusan untuk Pengurutan 3 Buah Bilangan [1]*

C. Kombinatorial

Teori kombinatorial merupakan ilmu untuk menghitung jumlah penyusunan objek-objek diskrit tanpa harus mengenumerasi setiap kemungkinan penyusunannya. Prinsip dasar penghitungan pada kombinatorial ada dua, yaitu:

a. Perkalian (*rule of product*)
   Percobaan 1: *p* hasil
   Percobaan 2: *q* hasil
   Percobaan 1 **dan** percobaan 2: *p* * *q* hasil
b. Penjumlahan (*rule of sum*)
   Percobaan 1: *p* hasil
   Percobaan 2: *q* hasil
   Percobaan 1 **atau** percobaan 2: *p* + *q* hasil

1. Permutasi

Permutasi dapat didefinisikan sebagai jumlah urutan yang berbeda dari pengaturan objek-objek diskrit. Permutasi merupakan bentuk khusus dari aplikasi kaidah perkalian.

Permutasi *r* dari *n* elemen adalah jumlah kemungkinan urutan *r* buah elemen yang dipilih dari *n* buah elemen, dengan $r \leq n$, yang dalam hal ini, pada setiap kemungkinan urutan tidak ada elemen yang sama[2]. Sehingga dapat dinotasikan sebagai:

$$P(n,r) = n(n-1)(n-2)...(n-(r-1)) = \frac{n!}{(n-r)!}$$

2. Kombinasi

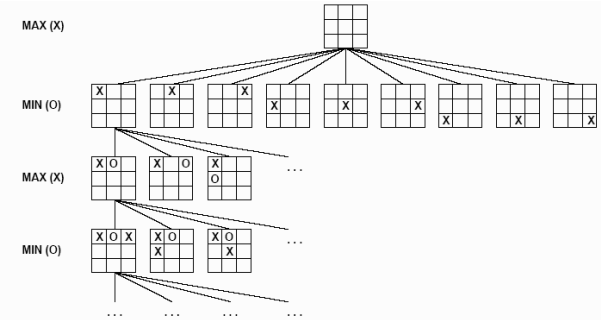
Kombinasi merupakan suatu bentuk khusus dari permutasi yang tidak memperhitungkan peletakan objek yang diacu, berbeda dengan permutasi yang memperhitungkan peletakan objek. Sebagai contoh, permutasi memperhitungkan (a,b) dan (b,a) sebagai dua penyusunan yang berbeda, sedangkan kombinasi memperhitungkannya sebagai satu kesatuan yang sama (mengandung a dan b). Perbedaan tersebut membuat perhitungan kombinasi memiliki jumlah yang lebih sedikit dibandingkan dengan perhitungan permutasi.

$$C(n,r) = \frac{n(n-1)(n-2)...(n-(r-1))}{r!} = \frac{n!}{r!(n-r)!}$$

D. Artificial Intelligence

Artificial Intelligence (AI) dapat didefinisikan sebagai sebuah proses komputasi yang memiliki sifat "pintar". Parameter dari "pintar" ini dijelaskan sebagai sebuah sistem yang dapat berpikir dan bertindak secara rasional, layaknya manusia berpikir dan bertindak[3].

Pada *video game*, AI dapat berinteraksi dengan pemain manusia melalui objek "hidup", atau berinteraksi di belakang layar (*background*). Interaksi AI melalui objek hidup biasanya dikaitkan dengan *bot* atau NPC (*Non-Player Character*), yang menggantikan peran pemain manusia sebagai penentu alur (*flow*) permainan. Dalam game yang melibatkan dua pihak, AI memiliki kemampuan untuk merespon sesuai dengan apa yang dilakukan player, atau bahkan menyusun strategi agar pemain dapat terkecoh oleh aksi yang dilakukan. Kemampuan AI yang semakin "pintar" akan membuat permainan yang menantang, dan membuat permainan menjadi semakin sulit. AI dalam permainan harus memperhitungkan berbagai kemungkinan yang ada, dan melakukan aksi sesuai perhitungan optimalnya.



***Gambar 2-5**: Permainan Tic Tac Toe, melibatkan dua pihak sebagai penentu alur permainan [4]*

Pada gambar di atas, diilustrasikan bagaimana kerja AI dalam sebuah permainan *Tic Tac Toe*. Jika pemain manusia mengambil langkah awal, hal yang dilakukan oleh AI adalah mengantisipasi langkah pemain dengan memblokir kolom yang memungkinkan pemain untuk menang, atau memulai strategi menangnya sendiri dengan menempatkan diri di kolom yang strategis. Jika AI mengambil langkah awal, hal yang dilakukannya hanya membuat strategi optimal untuk menang dengan menempatkan diri di kolom yang strategis.

AI tentunya memegang peran penting dalam *value* sebuah permainan. Permainan yang memiliki AI yang optimal dan unik akan menambah *value* dan memberikan kepuasan bermain kepada pemainnya. Berlaku juga untuk sebaliknya, apabila AI permainan tersebut jelek, maka *value*-nya akan berkurang dan menjadikan permainan tersebut tidak menarik. Saat ini banyak *video game developer* yang menawarkan sistem AI yang unik dan lebih cerdik, agar dapat meraih pangsa pasaran.

III. MIDDLE-EARTH: SHADOW OF MORDOR

Middle-Earth: Shadow of Mordor adalah sebuah *game* berjenis *action adventure role-playing-game* yang dirilis oleh *Warner Bros Interactive* dan dibuat oleh *Monolith Production*. Permainan yang mengambil tema dari dua karya J.R.R. Tolkien, *The Hobbit* dan trilogi *The Lord of The Rings* ini menceritakan kisah Talion, yaitu seorang *ranger* yang terbunuh setelah menyaksikan keluarganya dibantai secara mengenaskan. Ia dihidupkan kembali oleh *Spirit of Vengeance*, yang juga memberikannya kekuatan mahadahsyat nan gaib, untuk membalas dendam kepada siapa saja yang bertanggung jawab atas kejadian yang dialaminya sebelum dihidupkan kembali.

***Gambar 3-1**: Screenshot dari Middle-Earth: Shadow of Mordor [5]*

Berbeda dengan *video game* bertemakan *Lord of The Rings* sebelumnya, *Shadow of Mordor* memberikan kebebasan kepada pemainnya untuk menjelajah dan mengeksploitasi lingkungannya secara *open-world*, dan memberikan unsur aksi yang sangat memacu ketangkasan pemain. Latar tempat game ini, Mordor, dapat diakses secara keseluruhan oleh pemain, dan tentunya menyimpan berbagai macam *landmark* yang memukau khas Mordor. Selain itu, musuh yang menjadi ciri khas, yaitu *orcs* dan *uruks*, dapat ditemukan berkoloni seperti layaknya penduduk asli, tidak hanya muncul saat menyerang pemain[5].

Selain fitur-fitur yang telah disebutkan sebelumnya, fitur yang sangat ditonjolkan oleh *Monolith* pada video game ini adalah fitur Nemesis System, yang akan dibahas pada bagian selanjutnya.

IV. NEMESIS SYSTEM

Nemesis (jamak: *Nemeses*) adalah musuh dalam *Shadow of Mordor* yang dikontrol oleh AI. Nemesis ini muncul seiring dengan jalannya permainan, dan juga berkembang seperti halnya karakter pemain, Talion. Setiap Nemesis yang dihadapi oleh Talion pada permainan merupakan individu yang unik satu sama lainnya. Mereka memiliki karakteristik, kekuatan, dan kelemahannya sendiri. Karakteristik dari Nemesis mencakup 8 buah faktor, yaitu:

1. Nama (*Names*)
   Setiap nemesis memiliki nama dan gelar unik yang dibangkitkan secara random.
   Setidaknya ada 102 nama yang terdaftar pada Nemesis System, dan 172 gelar untuk setiap nama (dapat dilihat pada referensi [7]). Sehingga total kombinasi nama dan gelar yang dibangkitkan adalah 102 x 172 = 17544 pasangan nama dan gelar.
   Pada satu Nemesis System untuk satu game, setidaknya melibatkan sekitar 90 hingga 100 Nemesis. Kita dapat menghitung jumlah enumerasi yang diambil oleh game dengan kombinasi:
   $$C(17544,90) = \frac{17544!}{90!(17544-90)!} = 5.01\times10^{243}$$
   $$C(17544,100) = \frac{17544!}{100!(17544-100)!} = 2.08\times10^{266}$$
   Jadi ada sekitar $5.01\text{x}10^{243}$ hingga $2.08\text{x}10^{266}$ cara penyusunan nama untuk Nemesis dalam game.
2. *Visuals*
   Faktor ini mewakili tampilan fisik, suara, dan tingkah laku dari Nemesis, yang juga dibangkitkan secara random dan unik.
3. *Power Levels*
   Kekuatan dasar dari Nemesis yang dituliskan dengan angka. Semakin besar angkanya, maka semakin kuat

Nemesis tersebut. Power level akan meningkat apabila Nemesis berhasil membunuh Talion, atau menyelesaikan suatu misi.

4. *Rank*
   Semakin lama Nemesis berada dalam Nemesis System, rank yang didapat semakin tinggi, dan status rank yang tinggi dapat mempengaruhi *power level*.
5. *Relationships*
   Menggambarkan hubungan antar Nemesis dengan Nemesis lainnya pada Nemesis System. Contohnya, hubungan *rival* antara A dan B.
6. *Fighting Styles and Traits*
   Traits berisi kekuatan dan kelemahan dari Nemesis, yang nantinya berkorespondensi dengan *fighting style*. Contohnya, Nemesis yang memiliki kelemahan pada pelindung diri, akan lebih sering menggunakan penyerangan jarak jauh.
7. *Locations*
   Setiap Nemesis memiliki basis markasnya masing-masing.

Selain memiliki karakteristik, tiap Nemesis juga memiliki hierarkinya masing-masing pada Nemesis System, yang terbagi menjadi 3:

1. *Warchief*
   Hierarki tertinggi dari Nemesis, paling kuat, dan biasanya dikawal oleh 2-4 *Captain*. Jenis musuh ini tidak muncul secara langsung pada Nemesis System, melainkan harus "dipancing keluar" dengan membunuh *Captains* yang melindunginya.
2. *Captain*
   Hierarki Nemesis yang paling sering ditemukan. Jenis musuh ini lebih kuat bila dibandingkan dengan musuh lainnya, karena dia dapat mengakses *traits* yang dapat memberinya tambahan kekuatan, dan juga beberapa kelemahan.

***Gambar 4-1**: Atas: Interface Nemesis System [5]*
Bawah: Representasi Pohon Berakar untuk Nemesis System

3. *Soldier/Grunt*
   Jenis musuh yang paling sering ditemui oleh Talion, terkadang bisa dianggap tidak masuk dalam Nemesis System karena ada beberapa *grunt* yang tidak memiliki karakteristik Nemesis. Seiring berjalannya permainan, *grunt* bisa dipromosikan menjadi *captain* dengan melakukan sejumlah aksi, misalnya membunuh Talion atau Nemesis lainnya.

Salah satu hal yang menjadi perhatian dari adanya Nemesis System ini adalah bagaimana Nemesis di dalamnya dapat merespon sesuai dengan aksi yang dilakukan Talion di dalam Mordor. Respon yang dimaksud adalah perubahan karakteristik dari Nemesis yang terkait.

Ambil sebuah contoh, Nemesis dengan karakteristik dibawah mengalami *encounter* dengan Talion.

Name	Uruk The Slayer
Visuals	Normal
Hierarchy	Captain
Power Level	8
Strength	Invulnerable to Ranged, Poisoned Weapon
Weakness	Fear of Burning, Vulnerable to Stealth

Kemudian pada akhir *encounter* tersebut, Nemesis berhasil mengalahkan Talion dengan membunuhnya. Di bagian sebelumnya, telah dijelaskan bahwa setiap Nemesis memiliki karakteristik atau *stats* yang akan berubah sesuai *event* dari game. Maka setelah membunuh Talion, dapat diprediksi power level dari Nemesis akan bertambah.

Namun tidak hanya itu saja. Ketika pemain dihidupkan kembali, waktu di permainan telah berlalu cukup lama, dan dengan Nemesis System, pemain dapat melihat karakteristik Nemesis tadi, yang ternyata telah berubah.

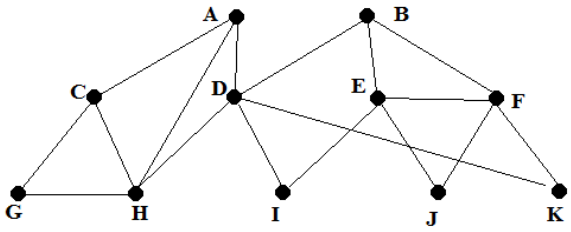
Name	Uruk The Slayer
Visuals	Normal
Hierarchy	Captain
Power Level	10
Strength	Invulnerable to Ranged, Poisoned Weapon, Immune to Fire, Bloodlust
Weakness	Vulnerable to Stealth

Jika kita membandingkan kedua tabel tadi, ternyata setelah membunuh Talion, tidak hanya power level yang berubah, ternyata *strength* dari Nemesis bertambah banyak. Hal ini tentunya memberi tantangan strategis dalam bermain, karena semakin sering pemain kalah dalam pertarungan, Nemesis akan semakin kuat, dan semakin sulit untuk dikalahkan.

Selanjutnya, kita analisis kasus kedua: Talion memenangkan pertempuran karena Nemesis lari dari pertempuran. Hal ini tentunya tidak membuat Nemesis tersebut hilang dari Nemesis System. Tetapi, *stats*-nya

bisa berubah, atau tetap seperti *stats* awal. Selain pada Nemesis, kekuatan Talion akan bertambah. Hal ini membuat jarak kekuatan antara Talion dan Nemesis semakin besar. Jarak yang sangat besar akan memengaruhi orientasi Nemesis ketika bertemu kembali dengan Talion. Contohnya, Nemesis lebih mudah melarikan diri, atau menyerang dengan ketakutan.

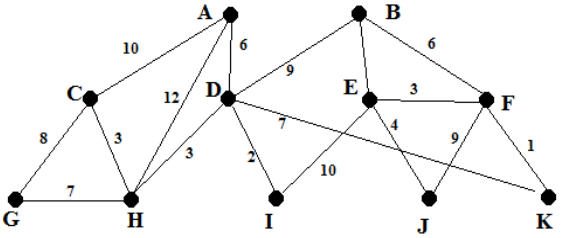
Lain halnya apabila Talion berhasil membunuh Nemesis. Selain hilangnya Nemesis dari Nemesis System dan peningkatan kekuatan Talion, pemain akan mendapatkan akses informasi karakteristik kepada setiap Nemesis yang memiliki *relationship* dengan Nemesis yang berhasil dibunuh. *Relationship* antar Nemesis dapat digambarkan dalam sebuah graf terhubung, dengan Nemesis sebagai simpul dan *relationship* sebagai sisi.



***Gambar 4-2**: Graf Relationship antar Nemesis*

Dari graf tersebut, dikatakan bahwa simpul yang berderajat besar memiliki akses informasi yang lebih banyak kepada simpul lain. Sehingga dapat disimpulkan bahwa semakin banyak suatu Nemesis memiliki *relationship*, semakin banyak juga akses informasi yang pemain dapat. Dalam permainan, tentunya seorang pemain memiliki keinginan untuk mendapatkan keuntungan sebesar-besarnya, baik dengan memanipulasi lingkungan, atau menyusun strategi. Dalam kasus ini, pemain ingin mendapatkan akses informasi sebanyak-banyaknya agar dapat mengalahkan seluruh musuhnya dengan cepat. Maka pemain harus membunuh Nemesis (simpul) yang terhubung dengan *relationship* (sisi) yang banyak, seperti simpul berlabel H pada Gambar 4-2, yang memiliki derajat 4. Dengan informasi tersebut, pemain dapat melanjutkan prosesnya untuk membunuh *Nemesis warchief* (A dan B).

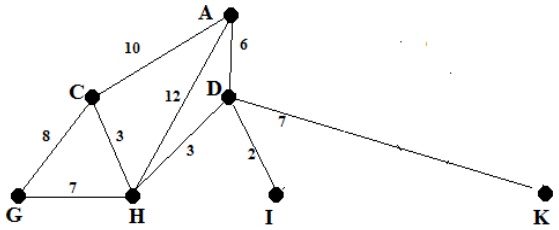
Akan tetapi, setiap *relationship* memiliki nilainya masing-masing terhadap sepasang Nemesis. Andaikan pemain ingin mendapatkan informasi mengenai seorang *warchief* dengan cara membunuh *captain* yang merupakan tangan kanan setia *warchief* tersebut. Maka ia akan memberikan respon yang beragam, misalnya peningkatan *power level* yang sangat tajam dan menantang pemain untuk bertarung. Namun, apabila pemain mengambil jalur lain, misalnya membunuh Nemesis yang ternyata menjadi musuh bebuyutan *warchief* tersebut, maka ia akan mengabaikannya, dan tidak akan ada peningkatan kesulitan yang berarti.



***Gambar 4-3**: Graf Relationship antar Nemesis, yang telah diberi bobot pada setiap sisinya, yang melambangkan nilai relationship*

Setiap nilai *relationship* pada Nemeses bisa dianalogikan sebagai bobot pada sisi graf. Sehingga graf pada Gambar 4-2 menjadi graf berbobot, yang digambarkan pada Gambar 4-3. Bobot yang tinggi melambangkan *relationship* yang penting, sedangkan bobot rendah melambangkan *relationship* yang renggang, atau nyaris menjadi *rival*.

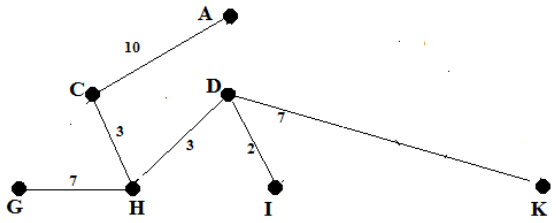
Sekarang, dalam suatu permainan, pemain ingin membunuh *warchief* A. Pada Gambar 4-3, untuk mencapai A, kita dapat melewati 4 buah simpul paling bawah (*captain* atau *grunt*), yaitu G, H, I, dan K. Jika kita ambil upagraf yang mewakili hubungan Nemesis tersebut dari graf, akan didapatkan upagraf seperti berikut.



***Gambar 4-4**: Upagraf dari Graf Relationship Nemesis, yang hanya menyertakan simpul-simpul yang dapat meraih A dari simpul-simpul bawah (captain atau grunt)*

Kita dapat menemukan 10 cara untuk meraih A dari simpul-simpul bawah (GCA, GHCA, GHA, GHDA, HGCA, HCA, HA, HDA, IDA, dan KDA). Namun, yang kita perlukan hanya lintasan yang memiliki jumlah bobot yang sedikit, sehingga *warchief* A bisa dikalahkan dengan cara yang mudah. Dari upagraf Gambar 4-4, lintasan terpendek adalah lintasan IDA, yang memiliki total bobot 2 + 6 = 8. Lintasan tersebut menawarkan cara yang lebih mudah untuk mengalahkan A, dibandingkan lintasan-lintasan lainnya.

Masih pada upagraf Gambar 4-4, sekarang pemain ingin mencari langkah termudah untuk mengalahkan seluruh Nemesis yang ada. Ini berarti pemain harus mencari lintasan yang mencakup seluruh simpul, tetapi harus memiliki total bobot yang minimum. Dengan menggunakan pohon merentang minimum (*Minimum Spanning Tree*), pemain dapat menemukan solusinya dengan mudah.



***Gambar 4-5**: Dengan menggunakan MST, pemain dengan mudah dapat membunuh seluruh Nemesis dari warchief A*

V. Conclusion

- Nemesis System merupakan fitur yang menawarkan tantangan kepada pemainnya untuk tidak hanya bermain secara *hack-and-slash* (asal menyerang), tetapi juga menyusun strategi dan taktik yang mangkus untuk menyerang musuh, karena setiap aksi dari pemain akan memengaruhi dunia permainan secara *open-world*.
- Beberapa teori pada matematika diskrit dapat digunakan untuk menyusun strategi dan taktik dalam mengalahkan setiap Nemesis yang ada pada permainan.

VI. Ucapan Terima Kasih

Pertama, penulis mengucapkan syukur dan terima kasih kepada Allah SWT yang telah memberikan akal, kesehatan, dan keberkahan kepada penulis, sehingga penulis bisa menyelesaikan makalah ini. Penulis juga ingin mengucapkan terima kasih kepada orang tua dan teman-teman penulis yang selalu memberi motivasi baik langsung maupun tidak langsung, dan juga kepada Pak Rinaldi Munir dan Ibu Harlili selaku dosen IF2120 Matematika Diskrit yang telah mengajarkan ilmunya dengan baik kepada penulis dan mahasiswa lainnya.

Referensi

[1] Rosen, Kenneth H. 2012. *Discrete Mathematics and Its Applications (Seventh Edition).*Singapura : McGraw-Hill.
[2] Munir, Rinaldi. 2003. *Matematika Diskrit Edisi Kedua*. Bandung : Penerbit Informatika.
[3] Russell, Stuart J., Norvig, Peter. 1995. *Artificial Intelligence: A Modern Approach*. New Jersey : Prentice-Hall, Inc.
[4] http://homepages.ius.edu/RWISMAN/C463/html/Chapter6.htm. Diakses pada tanggal 10 Desember 2014, pukul 17:35
[5] http://www.ign.com/games/middle-earth-shadow-of-mordor/pc-20008394. Diakses pada tanggal 10 Desember 2014, pukul 18:12
[6] http://shadowofmordor.wikia.com/wiki/Nemeses. Diakses pada tanggal 10 Desember 2014, pukul 18:20
[7] http://www.ign.com/wikis/middle-earth-shadow-of-mordor/The_Nemesis_System. Diakses pada tanggal 10 Desember 2014, pukul 18:20
[8] http://venturebeat.wordpress.com/2014/10/19/a-look-at-shadow-of-mordors-unique-nemesis-system/. Diakses pada tanggal 10 Desember 2014, pukul 20:03

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, 10 Desember 2014

Ahmad Naufal Farhan/ 13513049