Penggunaan Pohon Berakar untuk Mendesain Keseimbangan Pertarungan dalam Game

Muhammad Diaztanto Haryaputra, 13514002
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13514002@std.stei.itb.ac.id

***Abstract*—Desain video game adalah suatu hal yang kompleks, dan merupakan suatu hal yang membutuhkan ilmu multidisiplin untuk dibuat sesuai target tipe pemain yang ada. Video game berbasis *Player versus Enemies* sama kompleksnya dengan tipe video game lainnya. Penggunaan pohon berakar dapat menjadi alternatif untuk mengatur kemunculan musuh dalam game-game bertipe ini.**

***Keywords*—Borderlands 2, musuh, pohon, RNG.**

PENDAHULUAN

Sejak awal kemunculannya, video game telah menumbuhkan berbagai hal seperti budaya dan hobi, bahkan tidak sedikit yang menganggap karya multi disiplin ini sebagai layaknya keilmuan secara serius, seperti pada studi yang menunjukkan bahwa video game dengan kekerasan menumbuhkan faktor kriminal dan agresi [1], ataupun bagaimana video game dapat meningkatkan kemampuan kognitif seseorang [2].

Di samping segala studi tentang video game, studi tentang pengembangan video game sendiri sangatlah kurang. Tidak sedikit pengembang video game melalui jalan trial and error dalam mengukir karya yang "sempurna" untuk tipe pemain yang ditargetkan. Sebut saja video game *Diablo III* yang banyak disebut tidak pantas sebagai sequel dari *Diablo II* sebelum Blizzard, pengembang serial game *Diablo*, merilis update dan konten ekstra 2 tahun kemudian. Atau *Destiny* milik Bungie yang dianggap sebagai game yang mencoba meraih kesuksesan yang sama dengan serial *Borderlands* milik Gearbox dan 2K, hanya membosankan dan repetitif, sebelum mereka akhirnya merilis *Downloadable Content* (DLC) *The Taken King* setahun kemudian.

Begitu banyak aspek dari sebuah video game yang dapat dibahas satu per satu dan bagaimana mereka berinteraksi satu sama lain. Dalam makalah ini, saya membahas bagaimana kemunculan (spawn) musuh dalam game *Player versus Enemies* mempengaruhi pengalaman bermain pemain. Tidak sedikit dari sekian pemain yang sering melakukan *rage quit* atau menyelesaikan sesi permainannya karena frustrasi. Kita bermain game bukan untuk frustrasi, tapi untuk bersenang-senang. Di sinilah desain game menjadi serius.

Ada beberapa tipe aturan kemunculan musuh dalam berbagai game yang ada:

1. *Fixed spawn*

Fixed spawn adalah kemunculan musuh yang telah diatur secara langsung oleh pengembang game dengan tujuan mengatur skenario sesuai apa yang diinginkan oleh pengembang game. Kelebihan dari *fixed spawn* adalah kemampuannya memberikan tantangan yang sama pada semua pemain. Kekurangannya adalah tidak ada replay value dengan penggunaan *fixed spawn* karena setiap pemain menemui titik yang sama, pasti akan menghadapi musuh yang sama dan menggunakan strategi yang sama.

Gambar 1.1 Percobaan 1 kemunculan musuh di game *Bastion*

Gambar 1.2 Percobaan 2 kemunculan musuh di game *Bastion*

Dua screenshot game Bastion (dikembangkan oleh Supergiant Games dan dirilis oleh Warner Bros. Interactive Entertainment) di atas, adalah pertarungan

yang sama. Saya mengulangi proses pertarungan dengan sengaja menyerah dan mengulanginya kembali. Game ini menggunakan *fixed spawn* seperti yang telah dijelaskan, dan terlihat bahwa pada kedua pertarungan berbeda waktu itu memiliki komposisi pasukan musuh yang sama.

2. *Random spawn*

Random spawn adalah kemunculan musuh yang diatur secara tidak langsung oleh pengembang game yang melepaskan aturan kemunculan musuh ke tangan kanannya, *Random Number Generator* (RNG). Biasanya pengembang hanya memberikan berapa jumlah musuh yang harus dimunculkan dan berapa kemungkinan setiap tipe musuh muncul, yang dapat direpresentasikan sebagai permasalahan bola-bola dan kotak dalam kombinatorial. Kelebihan *random spawn* ini adalah setiap pemain akan mengalami hal yang berbeda dengan player lainnya, menciptakan pengalaman yang unik pada setiap pemain, dan memberikan replay value yang lebih karena setiap pertarungan akan berbeda. Kekurangannya adalah keseimbangan tingkat kesulitan dapat menjadi kacau karena memang tidak diatur secara langsung oleh pengembang game.

Gambar 1.3 Percobaan 1 kemunculan musuh di game *Borderlands 2*

Gambar 1.4 Percobaan 2 kemunculan musuh di game *Borderlands 2*

Dua screenshot game Borderlands 2 (dikembangkan oleh Gearbox dan dirilis oleh 2K) di atas, adalah pertarungan yang sama. Saya mengulangi proses pertarungan dengan keluar dari game dan mengulanginya kembali. Game ini secara garis besar menggunakan *random spawn* seperti yang telah dijelaskan, dan terlihat bahwa pada kedua pertarungan berbeda waktu itu memiliki komposisi pasukan musuh yang berbeda. Pada percobaan pertama, ada seorang tipe musuh mendekati pemain, lalu satu tipe membawa *shotgun*, dan satu lagi membawa *SMG*. Pada percobaan kedua ada dua tipe musuh pembawa *shotgun*, dan satu tipe pembawa *SMG*.

Pada makalah ini saya akan membahas bagaimana *random spawn* dapat dibuat lebih baik agar keseimbangan tingkat kesulitan tetap ada. Game yang akan saya gunakan sebagai game sampel dan ilustrasi dalam studi ini adalah Borderlands 2 yang dikembangkan oleh Gearbox dan dirilis oleh 2K.

II. Dasar Teori

A. Pohon

Pohon adalah graf tak-berarah terhubung yang tidak mengandung sirkuit [5]. Misalkan G = (V, E) adalah graf tak-berarah sederhana dan jumlah simpulnya n. Maka, semua pernyataan di bawah ini adalah ekivalen:

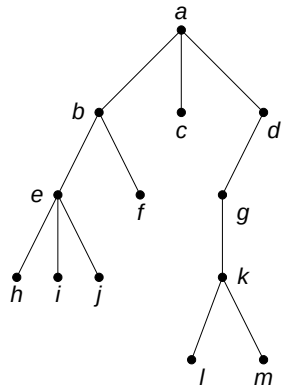
1. G adalah pohon.
2. Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
3. G terhubung dan memiliki m = n – 1 buah sisi.
4. G tidak mengandung sirkuit dan memiliki m=n–1 buah sisi.
5. G tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membuat hanya satu sirkuit.
6. G terhubung dan semua sisinya adalah jembatan.

B. Pohon berakar

Pohon yang satu buah simpulnya diperlakukan sebagai akar dan sisi-sisinya diberi arah sehingga menjadi graf berarah dinamakan pohon berakar (rooted tree).

Ada beberapa terminologi pohon berakar:

- Anak dan Orangtua
  b, *c*, dan *d* adalah anak dari simpul *a*.
  a adalah orangtua dari *b*, *c*, dan *d*.



Gambar 2.1 Pohon berakar

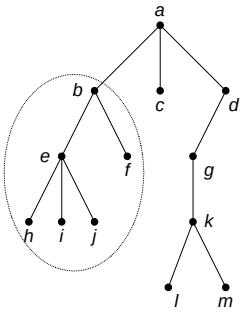
- Lintasan

Berdasarkan gambar 2.1, lintasan dari *a* ke *j* adalah *a*, *b*, *e*, *j*, dan panjang lintasan dari *a* ke *j* adalah 3.

- *Sibling*

Berdasarkan gambar 2.1, *f* adalah saudara kandung *e*, tetapi *g* bukan saudara kandung *e*, karena orangtua mereka berbeda.

- Upapohon



Gambar 2.2 Pohon berakar dengan upapohon yang dilingkari

- Derajat

Derajat sebuah simpul adalah jumlah upapohon (atau jumlah anak) pada simpul tersebut.

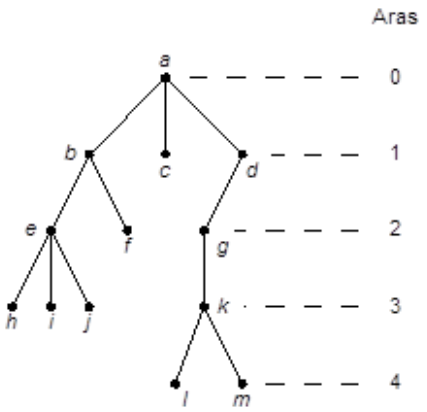
- Daun

Simpul yang berderajat nol (atau tidak mempunyai anak) disebut daun. Berdasarkan gambar 2.1, simpul *h*, *i*, *j*, *f*, *c*, *l*, dan *m* adalah daun.

- Simpul dalam

Simpul yang mempunyai anak disebut simpul dalam. Berdasarkan gambar 2.1, simpul *b*, *d*, *e*, *g*, dan *k* adalah simpul dalam.

- Aras/level



Gambar 2.3 Pohon berakar dengan aras yang ditunjukkan.

- Tinggi

Aras maksimum dari suatu pohon disebut tinggi atau kedalaman pohon tersebut. Pohon di atas, gambar 2.3, mempunyai tinggi 4.

C. Pohon *n-ary*

Pohon berakar yang setiap simpul cabangnya mempunyai paling banyak *n* buah anak disebut pohon *n-ary*.

Pohon *n-ary* dikatakan teratur atau penuh (*full*) jika setiap simpul cabangnya mempunyai tepat *n* anak.

III. KEMUNCULAN MUSUH DI *BORDERLANDS 2*

Borderlands 2 adalah game bergenre *First Person Shooter* dengan elemen *Role Playing Game* (RPG) yang mengutamakan *loot* dalam permainannya. Berbasis *Player versus Enemies*, game ini cocok digunakan sebagai sampel bagaimana kemunculan musuh berpengaruh pada keseimbangan pertarungan yang dihadapi pemain. Game ini menggunakan kombinasi dari *fixed spawn* dan *random spawn* untuk mencapai keseimbangan pertarungan dan variasi pertarungan yang lebih baik dengan tingkat kesulitan yang diatur secara langsung dan tidak langsung oleh pengembang game.

A. Tipe musuh

Pada Borderlands 2, ada sangat banyak jenis musuh dan varian-variannya. Di game ini tipe musuh dibagi menjadi 6 jenis berdasarkan tingkat kesulitannya dan *loot* yang dihasilkan [3][4]:

1. *Chump*
2. *Badass*
3. *Super Badass*
4. *Ultimate Badass*
5. *Chubby*
6. *Raid*

Raid tidak pernah muncul pada random spawn, kecuali satu musuh yang persentase kemungkinan kemunculannya sangat kecil.

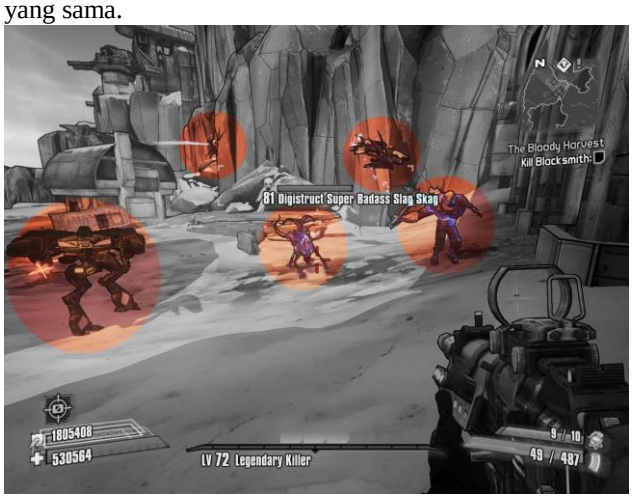
Untuk setiap kenaikan tingkat kesulitan, akan ada perbedaan yang cukup banyak, atau mencolok, atau keduanya. Secara garis besar, pertumbuhan tingkat kesulitan di game ini adalah eksponensial.

B. Bagaimana musuh muncul

Sebagian besar game ini diatur oleh RNG, termasuk bagaimana musuh muncul. Tidak ada bukti pasti bagaimana musuh muncul dan musuh apa yang dimunculkan di game ini, tetapi karena saya hanya menggunakan Borderlands 2 sebagai ilustrasi, aturan kemunculan musuh yang ingin saya bahas di sini adalah yang diatur oleh tipe RNG yang sama dengan pengaturan penyebaran loot di Borderlands 2 seperti pada artikel *Inside The Box* [3][4]. Masalah ini dapat direpresentasikan sebagai masalah bola-bola dan kotak pada kombinatorial. Misalkan ada 5 bola chumps, 1 bola Badass, 3 kotak spawn, dan setiap pengambilan bola, bola yang ada tidak berkurang, maka ada (1/6) * 3 atau 50% kemungkinan sebuah Badass akan muncul di area pertarungan. Dengan persoalan yang sama, akan ada (1/6) ^ 3 atau (1/48) kemungkinan 3 Badass akan muncul di pertarungan.

C. RNG dan keacakannya

(1/48) memang terdengar kemungkinan yang kecil atau bahkan sangat kecil, tetapi RNG tidaklah selalu *random*, karena sifat RNG adalah *pseudo-random* [4][6]. Ketika RNG melakukan pengacakan, ada kemungkinan untuk hasil pengacakan berikutnya menjadi mirip dengan hasil sebelumnya, dan menciptakan ketidakseimbangan persebaran angka acak yang dihasilkan. Pemain Borderlands 2 ataupun game lain yang menggunakan RNG pasti pernah mengalami kejadian yang sama berulang-ulang, misal, mendapatkan *legendary loot* berkali-kali dari kesempatan yang sangat kecil (1/30) ataupun tidak mendapatkannya untuk waktu yang sangat lama hingga ribuan kali percobaan dengan kesempatan yang sama.



Gambar 3.1 Sampel kemunculan musuh yang "buruk", memunculkan 5 (lima!) *Super Badass* sekaligus

Rincian kemunculan musuh pada sampel di atas adalah sebagai berikut:

Gambar 3.2 *Digistruct Super Badass Slag Skag*

Digistruct Super Badass Slag Skag adalah versi *Super Badass* dari musuh biasa *Skag*. Berada dua level lebih tinggi dari varian normal, musuh yang satu ini memiliki beberapa kelebihan, yaitu, memiliki *health* yang jauh lebih banyak, elemen yang dapat men-*debuff* player, mengeluarkan cairan *debuff* dengan *damage* yang sangat banyak pada pemain di dekatnya, dapat melempar bola-bola *debuff* yang meledak dari jarak jauh pada pemain, dapat menyembuhkan luka (menambah *health*) musuh di sekitarnya, dan memberikan elemen yang dibawanya kepada teman-temannya untuk dibawa oleh teman-temannya juga. Di sampel kemunculan musuh di atas, musuh ini muncul sebanyak 1 kali.

Gambar 3.3 *Super Badass Loader*

Super Badass Loader adalah versi *Super Badass* dari musuh level *Chump* bernama *Loader*. Dua level yang berbeda ini membawa banyak perbedaan, yaitu, musuh ini memiliki *health* yang sangat banyak, menggunakan 2 (dua) senjata api seperti yang dibawa pemain (masing-masing satu di tangannya, *Loader* biasa hanya membawa satu senjata api), memiliki *rocket pod* di punggungnya, menenteng dua *autocannon* api di pundaknya, dan dapat menggetarkan tanah di sekitarnya untuk menjauhkan pemain yang terlalu dekat. Musuh ini muncul sebanyak satu kali di sampel kemunculan musuh di atas.

Gambar 3.4 *Digistruct Badass Maniac*

Digistruct Badass Maniac adalah versi *Super Badass* dari musuh level *Chump* bernama *Maniac*. Jangan tertipu oleh namanya yang hanya membawa "gelar" *Badass*, musuh ini dua level lebih tinggi dari sobat-sobat kecilnya yang memiliki kedua tangan dengan ukuran yang sama (musuh ini memiliki tangan kanan sangat besar dan

tangan kiri sangat kecil karena mutasi). Varian *Badass* dari musuh *Maniac* adalah *Armored Maniac* dan *Gravedigger*, yang tidak muncul pada sampel di atas. *Digistruct Badass Maniac* ini memiliki *health* yang jauh lebih tinggi dari *Maniac* biasa secara signifikan, serangan yang lebih mematikan, jangkauan serangan yang lebih jauh, dan kecenderungan lebih untuk melempar senjatanya dari jauh untuk menyerang pemain lebih awal. Pada sampel di atas, musuh ini muncul sebanyak 1 kali.

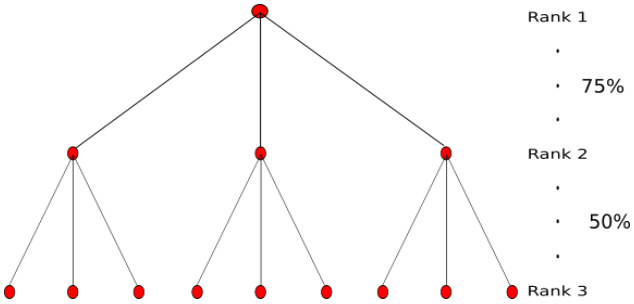
Gambar 3.5 *Super Badass Repair Surveyor v2.0*

Super Badass Repair Surveyor v2.0 adalah versi *Super Badass* dari musuh level *Chump* bernama *Repair Surveyor v2.0*. Perbedaan dua level ini tidak begitu memberi banyak perubahan dari varian biasanya, yaitu, memiliki serangan berbasis area yang ditembakkan berturut-turut sebanyak 3 kali dalam satu kali menyerang, *health* dan *shield* yang lebih tangguh secara signifikan, dan gerakan manuver terbang yang lebih sulit diprediksi oleh pemain. Musuh ini muncul sebanyak 2 (dua) kali pada sampel kemunculan musuh di atas.

Seperti pada sampel sebelumnya, kemunculan musuh dengan tingkat kesulitan yang tinggi dan jumlah yang banyak sekaligus yang dapat mengakibatkan ketidakteraturan tingkat kesulitan ini bisa mengubah pengalaman permainan pemain menjadi lebih buruk, terlebih di tingkat kesulitan yang memang sudah lebih sulit, dan berakhir kurang baik, baik dalam sisi reputasi game ataupun psikologis pemain. Tantangan tidak terduga seperti ini memang bisa jadi dibuat secara sengaja dengan mengimplementasikan random spawn.

IV. Penggunaan Pohon Berakar

Dengan pohon berakar, saya dapat merepresentasikan tingkat kesulitan musuh sebagai titik dan daun, dengan makin rendah level titik atau daun tersebut, makin mudah musuh yang muncul.



Gambar 4.1 Pohon berakar 3-*ary*

Pohon di atas adalah cara saya membuat keseimbangan pertarungan yang lebih baik. Setiap level merepresentasikan tingkat kesulitan musuh, dengan makin rendah level, makin mudah musuh yang muncul, lalu jumlah *node* pada level itu merepresentasikan jumlah musuh yang dimunculkan pada tingkat kesulitan tersebut, dan ada kemungkinan sekian persen, yang tidak ditentukan, untuk setiap *node* turun ke level di bawahnya. Cara kerja sistem pohon ini adalah pengembang game menentukan tingkat kesulitan suatu pertarungan di suatu area, dijadikan sebagai akar pohon berakar, lalu RNG menentukan apakah musuh yang dimunculkan lebih mudah atau tidak, jika iya, maka level turun, tetapi jumlah musuh bertambah sebanyak tiga kali lipat. Setelah itu, setiap *node* di level tadi diproses secara rekursif menjadi akar pohon lain, dan RNG menentukan kembali untuk turun atau tidak, dan seterusnya.

Misal, pengembang game ingin suatu pertarungan di suatu area menjadi sesulit melawan suatu *Super Badass*. Jika RNG menetapkan level *Super Badass*, maka hanya 1 musuh dengan kesulitan *Super Badass* yang muncul. Tetapi andaikan RNG ingin turunkan level, maka sekarang ada 3 pohon untuk diproses kembali. Pada setiap pohon, mungkin saja RNG ingin turun ke level bawahnya, atau tidak. Beberapa kemungkinan kemunculan musuh dengan tingkat kesulitan pertarungan setara *Super Badass* adalah:

1. 1 *Super Badass*
2. 3 *Badass*
3. 2 *Badass* ditambah 3 *Chumps*
4. 1 *Badass* ditambah 6 *Chumps*
5. 9 *Chumps*

Dengan sistem pohon seperti ini, pengembang game tidak menentukan berapa banyak musuh yang muncul, tetapi, pengembang game hanya memerlukan untuk mengatur pertarungan apa, di mana, dan sesulit apa. Tidak hanya tingkat kesulitan pertarungan yang diatur, tetapi dengan sistem ini jumlah musuh pun dinamis, sesuai tingkat kesulitan musuh yang dimunculkan.

V. Kesimpulan

Penggunaan RNG secara utuh dapat menghasilkan *output* yang tidak terduga atas ketidakmurnian hasil acaknya. Pohon berakar dapat menjadi alternatif dalam mengatur kemunculan musuh di game berbasis *Player versus Enemies*.

Referensi

[1] http://www.sciencedaily.com/releases/2013/03/130326121605.htm, diakses 8 Desember 2015, 21.40

[2] https://www.academia.edu/182611/Video_game_research_in_cognitive_and_educational_sciences, diakses 8 Desember 2015, 21.52

[3] http://www.gearboxsoftware.com/community/articles/1092/inside-the-box-evolution-of-loot, diakses 9 Desember 2015, 8.59

[4] http://www.gearboxsoftware.com/community/articles/1087/inside-the-box-the-borderlands-2-loot-system, diakses 9 Desember 2015, 8.59
[5] Munir, Rinaldi. *Matematika Diskrit*. Bandung : Penerbit Informatika, Palasari.
[6] Hellekakek, Peter. 1998. *Good random number generators are (not so) easy to find.*

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 2015

Muhammad Diaztanto Haryaputra, 13514002