BAB II

GENETIC ALGORITHM

Dalam bab ini kita akan menjelaskan mengenai Genetic Algorithm dan kegunaan algoritma tersebut dalam game “Splatted”. Genetic Algorithm (atau disebut Algoritma Genetik) adalah sebuah Algoritma yang terinspirasi oleh proses seleksi alam, diciptakan oleh John Holland dan tim-nya pada tahun 1960. Genetic Algorithm dapat dipakai untuk menemukan pilihan terbaik dari sebuah permasalahan, seperti catur, optimisasi Decision Tree dan masih banyak lagi.

1. Metodologi

Genetic Algorithm yang didasarkan pada seleksi alam, akan menciptakan sejumlah kandidat solusi yang pada algoritma ini akan berevolusi dengan harapan setiap generasi solusi yang dibuat akan selalu berevolusi dan lebih bagus daripada solusi generasi sebelumnya.

1. Terminologi

Sebelum memasuki tahap – tahap dalam sebuah Genetic Algorithm, perlu diperjelas beberapa istilah yang akan sering diulang dalam bab ini, yaitu;

1. Gene, Chromosom (Kromosom)

Kromosom adalah sebuah representasi calon solusi yang dibuat oleh algoritma genetik, umumnya direpresentasikan menggunakan sebuah array 1 dimensi. sementara bagian – bagian yang membentuk Kromosom adalah sebuah Gene, yang merepresentasikan sebuah variabel dalam kromosom, direpresentasikan menggunakan angka, huruf ataupun angka biner.

1. Populasi

Populasi adalah kumpulan dari kromosom – kromosom yang ada. Dalam sebuah populasi dapat berisi satuan hingga ratusan jumlah kromosom yang nantinya akan mempengaruhi kecepatan dari Algoritma tetapi akan meningkatkan kemungkinan mendapat solusi yang lebih baik.



Gambar 2.1

Visualisasi Gene, Kromosom dan Populasi

1. Langkah - langkah Algoritma Genetik

Pada subbab ini akan dijelaskan tahap – tahap dalam Algoritma Genetik dan apa yang terjadi. Tahap – tahap yang dilakukan adalah sebagai berikut;

1. Inisialisasi Populasi

Pada tahap ini, dibuatlah populasi pertama atau juga bisa disebut generasi pertama dengan kromosom – kromosom yang isinya acak. Di sini perlu ditetapkan beberapa hal penting, seperti : cara merepresentasikan solusi, jumlah kromosom dalam 1 populasi dan jumlah iterasi maksimal agar Algoritma bisa berhenti.

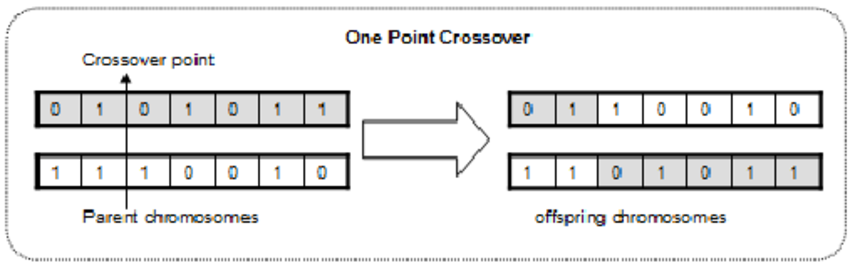
1. Seleksi

Setelah populasi pertama dibuat, maka akan dipilih kromosom – kromosom terbaik berdasarkan fitness mereka. Fitness adalah sebuah nilai yang menunjukkan kecocokan antara kromosom dengan solusi yang diinginkan, cenderung bila nilai fitness semakin tinggi, maka itu merupakan tanda bila kromosom semakin bagus. Fitness dihitung menggunakan fungsi, dimana nilai dari seluruh Gene dalam sebuah kromosom akan dimasukkan dalam sebuah rumus atau perhitungan yang mengeluarkan nilai Fitness dari kromosom tersebut.

Setelah mendapatkan fitness dari seluruh kromosom dalam sebuah populasi, kromosom – kromosom dengan nilai Fitness tertinggi akan dipilih untuk “reproduksi”. Metode untuk ber-reproduksi ini ada berbagai macam, tetapi terdapat 2 unsur utama yang harus ada dalam sebuah metode pemilihan kromosom; pemilihan berdasarkan fitness tertinggi dan memiliku unsur. Contoh pemilihan kromosom yang populer adalah metode Roulette Wheel, dimana setiap kromosom yang ada memiliki kesempatan untuk dipilih dan semakin tinggi fitness-nya, semakin tinggi kemungkinan agar kromosom tersebut dipilih.

1. Reproduksi

Tahap ini akan terjadi setelah dipilih 2 kromosom yang akan dirujuk sebagai parent atau orang tua, lalu akan diciptakan anak baru menggunakan 2 parent yang telah dipilih menggunakan metode yang dinamakan crossover, dimana kedua kromosom parent yang dipilih akan digabung untuk membuat anak kromosom baru. Terdapat banyak metode untuk melakukan crossover, salah satu contoh yang umum adalah One Point Crossover.



Gambar 2.2

Visualisasi One Point Crossover

Dalam One Point Crossover, dipilih sebuah titik diantara 2 Gene yang akan memisah sebuah kromosom jadi 2. Lalu setelah posisi tersebut dipilih, potong kedua kromosom di posisi tersebut, tukar kromosom yang telah terpotong, entah yang kiri atau yang kanan, dan sambung kembali kedua kromosom tersebut. Dari situ telah diciptakan 2 anak kromosom baru, dan Algoritma Genetik kembali ke tahap seleksi untuk memilih parent baru. Proses ini akan diulang hingga jumlah kromosom anak yang dibuat mencapai jumlah kromosom seharusnya dalam sebuah populasi yang telah ditetapkan di tahap inisialisasi.

1. Mutation (Mutasi)

Saat seluruh anak kromosom telah dibuat dan sebelum memasuki iterasi selanjutnya, Mutasi dilakukan terhadap setiap anak kromosom yang telah dibuat. Dalam mutasi, sebuah kromosom akan mengalami perubahan ke dalam struktur Gene-nya yang dapat terjadi dalam berbagai cara, seperti salah satunya adalah perubahan isi Gene secara acak atau dinamakan Random Mutation.



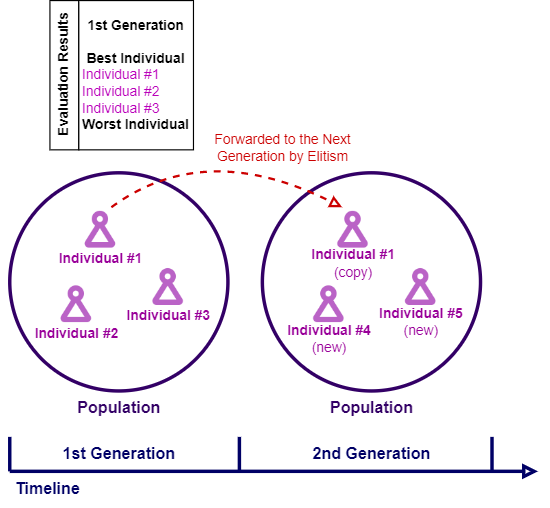
Gambar 2.3

Visualisasi Random Mutation

Ide dari Random Mutation itu sederhana, dimana setiap Gene yang ada di dalam kromosom memiliki kemungkinan yang bisa diatur oleh pengguna untuk mengganti isi sebuah Gene menjadi isi yang baru. Terdapat banyak lagi metode untuk melakukan mutasi, seperti Bit-String mutation dimana setiap Gene dilihat sebagai sebuah angka biner dan setiap bit bisa diganti. Perbedaan utama antara mutasi dan Crossover adalah meskipun mereka memiliki tujuan yang mirip, yaitu mengubah sebuah kromosom, Crossover mengubah kromosom berdasarkan Gene 2 kromosom, sementara Mutation mengubah sebuah kromosom berdasarkan isi kromosom itu sendiri atau mengubah isi Gene menjadi isi yang baru.

1. Elitism

Elitism adalah sebuah tahap yang bersifat opsional tapi dapat membantu dalam mencegah turunnya kualitas kromosom. Elitism adalah sebuah tahap dimana sejumlah kromosom terbaik dalam sebuah populasi dimasukkan dalam populasi generasi berikutnya, hal ini bertujuan agar bila kromosom – kromosom yang terdapat di generasi baru secara tidak beruntung memiliki tingkat Fitness yang secara keseluruhan lebih rendah daripada generasi sekarang ini, kromosom – kromosom yang dipindah ke generasi baru dapat menjadi jaminan yang menuntun kromosom lain untuk menjadi lebih baik.



Gambar 2.4

Visualisasi konsep Elitism

1. Repetisi dan Pemberhentian

Setalah tahap – tahap di atas telah diselesaikan, algoritma genetik telah membuat sebuah generasi baru yang memiliki harapan nilai Fitness lebih baik daripada generasi sebelumnya. Setelah generasi baru dibuat, maka algoritma genetik akan kembali ke tahap ke-2 yaitu seleksi, maju ke tahap ke-3 reproduksi dan siklus ini akan terus berlanjut hingga jumlah generasi telah mencapai iterasi maksimal yang telah ditetapkan atau tidak terjadi perkembangan terhadap Fitness maksimal dalam sebuah populasi sebanyak jumlah iterasi yang telah ditentukan.

1. Implementasi dalam Tugas Akhir

Dalam tugas akhir kami, kita akan memakai library yang telah disediakan oleh Diego Giacomelli dengan username Github giacomelli bernama GeneticSharp. GeneticSharp sesuai namanya adalah sebuah library yang telah menyediakan Genetic Algorithm siap dipakai. Dalam subbab ini akan dijelaskan kode – kode dalam pembuatan Algoritma genetik dasar.

Algoritma 2.1 Pseudocode Library GeneticSharp

1. Menggunakan library GeneticSharp.Domain.\*
2. varKromosom = new FloatingPointChromosome(
3. Array batas minimum gene,
4. Array batas maximum gene,
5. Array jumlah bit gene,
6. Array jumlah bilangan desimal)
7. varPopulasi = new Population(
8. minPopulasi,maxPopulasi,kromosom => varKromosom)
9. varFungsiFitness = new FuncFitness((c)=>
10. kromosomSekarang = c as FloatingPointChromosome
11. arrKromosom = kromosomSekarang.ToFloatingPoints()
12. {fungsi untuk menghitung fitness anda}
13. return fitness
14. )
15. varMetodeSeleksi = new RouletteWheelSelection()
16. varCrossover = new UniformCrossover(kemungkinan crossover terjadi {0-1})
17. varMutasi = new UniformMutation()
18. varStopCon = new GenerationNumberTermination(1000);
19. varGA = new GeneticAlgorithm(
20. Populasi => varPopulasi,
21. Fitness => varFungsiFitness,
22. Metode Seleksi => varMetodeSeleksi,

**Algoritma 2.1 (Lanjutan)**

1. Crossover => varCrossover,
2. Mutasi => varMutasi)
3. varGA.Termination = varStopCon
4. varGA.Start();
5. hasilJadi = varGA.BestChromosome

Sekarang akan diperjelas kegunaan dari masing – masing Line of Code ;

1. Line 3 - 7 : Dalam membuat sebuah kromosom, GeneticSharp memerlukan 4 parameter, yaitu : batas max Gene, batas min Gene, jumlah bit Gene, dan jumlah bilangan desimal, semua dalam bentuk array. Batas max dan min Gene berguna untuk mengatur batas dari isi Gene dalam membuat populasi awal, ini berguna apabila setiap Gene memiliki batas maksimal yang berbeda. Parameter selanjutnya adalah jumlah bit dikarenakan GeneticSharp menggunakan mewarisi Class BinaryChromosomeBase yang hanya menggunakan angka biner. Parameter terakhir adalah jumlah angka desimal apabila Gene yang diinginkan ingin memiliki angka koma.

GeneticSharp juga menyediakan Class IntegerChromosome yang memiliki kromosom numerik dan Gene angka biner yang mendirikan angka numerik tersebut, lalu ada BinaryChromosomebase yang merupakan basis dari 2 kelas Kromosom yang lainnya. Bila user tidak puas dengan pilihan ini maka user dapat membuat sebuah Class kromosom baru yang harus mewarisi class ChromosomeBase, dan melakukan Override Function tertentu.

1. Line 9 - 10 : Di Line ini ditentukan jumlah minimum dan maximum jumlah kromosom dari sebuah populasi, dan di parameter ketiga adalah kromosom yang telah dibuat di line 3.~
2. Line 12 - 17 : Di sini, dapat dibuat sebuah fungsi fitness yang akan menghitung fitness dari semua kromosom yang ada. Array dari kromosom yang diperiksa dapat diambil menggunakan kode di line 13 dan 14, lalu dengan array kromosom tersebut, fitness dapat dihitung sesuai dengan keperluan user.~
3. Line 19 : Line ini dipakai untuk menentukan metode Seleksi kromosom. Metode yang disediakan oleh Diego Giacomelli diantaranya ada Elite, Stochastic Universal sampling, Uniform Crossover dan masih banyak lagi. Bila ingin membuat seleksi sendiri dapat membuat class baru yang mewarisi Class SelectionBase.`
4. Line 21 : Di sini dipilih Crossover yang akan dipakai oleh Algoritma Genetik, Giacomelli sekali lagi menawari kita dari banyak pilihan seperti One-Point, Two-Point, Cut and spliced, dan banyak lagi. Crossover buatan sendiri bisa dibuat juga tetapi mewarisi Class CrossoverBase.
5. Line 23 : Mirip dengan line 21, line ini dipakai untuk memilih mutasi yang akan digunakan dalam algoritma genetik. Beberapa pilihan yang ada diantaranya adalah ; Flip-Bit, Reverse Sequence dan Twors. Mutasi buatan sendiri harus mewarisi Class MutationBase.
6. Line 25 : Kita dipersilahkan memilih dari berbagai kondisi berhenti untuk algoritma genetik kita, beberapa factor yang tersedia adalah Jumlah Generasi, Lama Evolusi, Fitness yang tidak naik naik, dan Fitness yang telah mencapai nilai tertentu. Seperti biasa dapat menggunakan Class sendiri yang mewarisi Class TerminationBase.
7. Line 27 - 36: Di sini seluruh pilihan dan fungsi yang telah dibuat sebelumnya dimasukkan dalam Constructor Genetic Algorithm yang baru ini, mulai dari populasi hingga mutasi yang ada. Lalu di line 36 algoritma genetik sudah mulai berjalan dan mulai mencari solusi
8. Line 38 :Hasil kerja keras algoritma anda disimpan dalam sebuah variabel.