BAB VI

IMPLEMENTASI MAP GENERATION MENGGUNAKAN GA

Menggunakan GeneticSharp oleh Giacomelli, bab ini akan menjelaskan dan menyelam menuju aspek Map Generationnya. Dalam bab ini, akan dijelaskan seluruh aspek Map Generation dari game Splatted, dimulai dari kromosom hingga seluruh fitness yang dipakai.

1. Chromosome / Representasi

Dimulai dengan dasar dari sebuah Genetic Algorithm, yaitu sebuah kromosom, dan di subbab ini akan dijelaskan implementasi Kromosom yang akan dipakai sebagai perantara solusi dalam Genetic Algorithm kita.

1. Tile Chromosome

Berikut adalah potongan kode untuk membuat kromosom tile generation :

Segmen Program 6.X Class TileChromosome

1. public class TileChromosome : ChromosomeBase{
2. private readonly int m\_ukuranMap;
3. private readonly int m\_powerup;
4. public GameChromosome(int ukuranMap,int powerup) : base(ukuranMap){
5. int temp;
6. m\_ukuranMap = ukuranMap;
7. m\_powerup = powerup;
8. var mapValues = RandomizationProvider.Current.GetInts (ukuranMap, 0, 2);
9. for (int i = 0; i < ukuranMap; i++)
10. ReplaceGene(i, new Gene(mapValues[i]));
11. for (int i = 0; i < 5; i++){
12. temp = Mathf.FloorToInt(Random.Range(0, ukuranMap));
13. while (mapValues[temp] == 3)
14. temp = Mathf.FloorToInt(Random.Range(0, ukuranMap));
15. ReplaceGene(temp, new Gene(3));
16. }
17. for (int i = 0; i < powerup; i++){

Segmen Program 6.X (Lanjutan)

1. temp = Mathf.FloorToInt(Random.Range(0, ukuranMap));
2. while (mapValues[temp] == 3 || mapValues[temp] == 2)
3. temp = Mathf.FloorToInt(Random.Range(0, ukuranMap));
4. ReplaceGene(temp, new Gene(2));
5. }
6. }
7. public override Gene GenerateGene(int geneIndex){
8. return new Gene(RandomizationProvider.Current.GetInt(0, 3));
9. }
10. public override IChromosome CreateNew(){
11. return new GameChromosome(m\_ukuranMap,m\_powerup);
12. }
13. public override IChromosome Clone(){
14. var clone = base.Clone() as GameChromosome;
15. return clone;
16. }
17. }

Penjelasan untuk masing masing baris kode :

* 1 : TileChromosome menggunakan class ChromosomeBase yang telah disediakan oleh GeneticSharp untuk membuat kromosom yang bisa dipakai untuk Genetic Algorithm nanti.
* 3–4&8-9 : Jujur ini ada karena wiki memasukkan ini di dalam tutorial, jadi kode ini tidak pasti memiliki kontribusi, tetapi dimasukkan aja agar aman.
* 6 : Constructor untuk membuat kromosom baru dengan parameter ukuran map yang berisi luas dari level dan powerup yang berisi jumlah powerup yang berada dalam map ini.
* 11 : Variabel mapValues akan diisi dengan sebuah array integer yang berukuran luas level dan akan diisi oleh angka dengan range 0 – 2.
* 12 – 13 : Fungsi ini adalah untuk mengisi Kromosom dari class ini (yang berada di class ChromosomeBase) dengan MapValues.
* 14 – 19 : Mengubah 5 gene dalam kromosom menjadi integer 3, dimana 3 adalah representasi atas seorang karakter, dan ini dilakukan untuk mempermudah proses Generasi.
* 20 – 25 : Sama seperti proses diatas, hanya ketimbang mengganti gene menjadi 3, gene diganti menjadi 2 yang merepresentasikan bola spesial di level.
* 28-30 : Override function yang dipakai untuk Membuat gene baru, sepertinya dipakai dalam beberapa Mutation yang disediakan.
* 32 – 33 : Fungsi yang dipakai untuk membuat kromosom baru? Sulit melihat kegunaan fungsi ini karena terdapat Clone beberapa baris dibawah.
* 36 – 38 : Fungsi ini dipakai untuk menduplikat kromosom, dimana fungsi ini tampaknya krusial dalam fase seleksi untuk menduplikat kromosom ke generasi baru.

1. Templated Map Chromosome

Sekarang setelah melihat kode Tile Chromosome, akan ditunjukkan kode yang digunakan untuk membuat Template Map Chromosome, tetapi sebelum itu, kromosom ini menggunakan class lain yang dipakai untuk mengganti angka – angka dalam gene menjadi template 5X5, berikut adalah class tersebut :

Segmen Program 6.X Class PossibleTemplates

1. public static class PossibleTemplates{
3. public static int[][,] oneWayTemplates = new int[][,]{…}
4. public static int[][,] twoWayTemplates = new int[][,]{…}
5. public static int[][,] fourWayTemplates = new int[][,]{…}
7. public static int getTemplateAmount(){
8. return oneWayTemplates.Length + twoWayTemplates.Length \* 2 + fourWayTemplates.Length \* 4;
9. }
10. public static int[,] getTemplate(int id){
11. int tempID = id;
12. if (tempID < 0)
13. tempID = -tempID - 1;
14. int rotation;
15. int[,] chosenTemplate;
16. if (tempID < oneWayTemplates.Length){
17. rotation = 0;
18. chosenTemplate = oneWayTemplates[tempID];
19. }

Segmen Program 6.X (Lanjutan)

1. else if (tempID < oneWayTemplates.Length + twoWayTemplates.Length \* 2){
2. tempID -= oneWayTemplates.Length;
3. rotation = tempID % 2;
4. chosenTemplate = twoWayTemplates[Mathf.FloorToInt(tempID / 2)];
5. }
6. else{
7. tempID = tempID - oneWayTemplates.Length - (twoWayTemplates.Length \* 2);
8. rotation = tempID % 4;
9. chosenTemplate = fourWayTemplates[Mathf.FloorToInt(tempID / 4)];
10. }
11. int[,] resultTemplate = new int[5, 5];
12. if (rotation == 0)
13. resultTemplate = (int[,])chosenTemplate.Clone();
14. else if (rotation == 1)
15. //rotasi ke kanan
16. for (int i = 0; i < 5; i++)
17. for (int j = 0; j < 5; j++)
18. resultTemplate[j, 4 - i] = chosenTemplate[i, j];
19. else if (rotation == 2)
20. //rotasi ke 180 derajat
21. for (int i = 0; i < 5; i++)
22. for (int j = 0; j < 5; j++)
23. resultTemplate[4 - i, 4 - j] = chosenTemplate[i, j];
24. else
25. //rotasi ke kiri
26. for (int i = 0; i < 5; i++)
27. for (int j = 0; j < 5; j++)
28. resultTemplate[4 - j, i] = chosenTemplate[i, j];
29. // kalau nomor yang diberi negatif beri power
30. if (id < 0 && resultTemplate[2, 2] == 0)
31. resultTemplate[2, 2] = 2;
32. return resultTemplate;
33. }

Sekarang untuk penjelasan mengenai kode diatas :

* 3 – 5 : Variabel one,two dan four way template akan diisi secara manual apa template – template yang bisa dipakai oleh sistem berdasarkan jumlah orientasi berbeda yang dapat dimiliki sebuah template. Alternatif lain adalah menempatkan template sebagai file external, tetapi dimasukkan dalam script secara hard code juga bekerja karena game tidak memerlukan daftar template yang bisa diganti.
* 7 – 9 : Fungsi ini dipakai untuk mengambil jumlah potensial dari seluruh template, dimana four way dikali 4, two way dikali 2 dan one way tidak diubah, ketiga angka tersebut dijumlah dan di return sebagai jumlah potensial template yang tersedia.
* 11 – 12 : Ini adalah fungsi untuk mengambil template berdasarkan dari id yang diberikan. Setelah memberi id tersebut, id akan disimpan ke dalam variabel tempID agar bisa diolah.
* 13 – 14 : Bila tempID bernilai negatif, maka tempID dimutlakkan lalu dikurangi satu. Bile tempID bernilai negatif, itu menandakan bila template tersebut bisa memiliki sebuah bola spesial di tengah template tersebut, jadi karena nilai negatif sudah disimpan di parameter id, maka tempID bisa diubah positif, lalu tempID dikurangi 1 karena ada 0 tetapi tidak ada -0, sehingga ini bekerja sebagai alternatif -0 tersebut.
* 19 – 32 : Di sini akan diambil berdasarkan tempID sebuah template yang akan direturn setelah diolah. Akan ada 2 variabel yang diisi, yaitu rotation yang mengatakan seberapa banyak rotasi template yang akan dilakukan nanti dan chosen template yang berisi template tersebut.

Tahap pertama dari metode ini adalah untuk melihat apabila tempID masuk dalam range one way, two way atau four way. Setelah ditentukan, tempID dikurangi dengan jumlah variabel dibawahnya, semisal bila tempID memasuki range four way dimana four way adalah yang paling terkhir diakses, maka tempID akan dikurangi panjang array one way dan two way dikali 2 (karena setiap isi array two way bisa diorientasikan 2 arah).

Dan setelah mendapat tempID, tempID tersebut akan di DIV oleh jumlah orientasi range array, bila kasus diatas maka akan didiv 4, lalu dari tempID div 4 maka akan diambil Template dari array template yang bersangkutan dan memasukkan template tersebut ke dalam variabel chosen template. Dan terakhir untuk menentukan rotasi dari template, rotation akan diisi dengan tempID di MOD dengan jumlah orientasi range array.

* 34 – 51 : Meski panjang, bagian ini sederhana, dimana template dirotasi sebanyak 90˚ \* variabel rotation ke arah kanan.
* 53 – 54 : Bila parameter ID bernilai negatif dan tile di paling tengah bernilai 0 (tidak ada apa apa), maka tile di tengah diberi bola spesial.

Dengan Class PossibleTemplates dijelaskan, maka kita bisa masuk ke class TemplatedChromosome, sebuah Chromosome berbasis template :

Segmen Program 6.X Templated Map Chromosomes

1. public class TemplatedMapChromosome : ChromosomeBase{
2. private readonly int m\_ukuranMap;
3. public TemplatedMapChromosome(int ukuranMap) : base (ukuranMap){
4. m\_ukuranMap = ukuranMap;
5. var mapValues = RandomizationProvider.Current.GetInts(ukuranMap, -(PossibleTemplates.getTemplateAmount()), (PossibleTemplates.getTemplateAmount()) - 1);
6. for (int i = 0; i < ukuranMap; i++)
7. ReplaceGene(i, new Gene(mapValues[i]));
8. }
9. public override Gene GenerateGene(int geneIndex){
10. return new Gene(RandomizationProvider.Current.GetInt(-(PossibleTemplates.getTemplateAmount()) , (PossibleTemplates.getTemplateAmount())-1));
11. }
12. public override IChromosome CreateNew(){
13. return new TemplatedMapChromosome(m\_ukuranMap);
14. }
15. public override IChromosome Clone(){
16. var clone = base.Clone() as TemplatedMapChromosome;
17. return clone;
18. }
19. }

Dibandingkan Segmen Program 6.X diatas kode ini kurang lebih sama dengan perbedaan utama terdapat di line 7, dimana MapValues ketimbang diisi 0-3 akan diisi dengan range -JumlahTemplate hingga JumlahTemplate-1. Selain itu, tidak terdapat perbedaan lain, mungkin salah satu perbedaan yang lain adalah ukuran map yang didapat oleh Kromosom ini adalah JumlahArea / 25, 25 merupakan luas area 5x5.

1. Main Code

Setelah Chromosome template dan tile, sekarang adalah kode yang digunakan untuk menjalankan Genetic Algorithm tersebut :

Segmen Program 6.X Genetic Algorithm

1. bool useMirrorFitness = true;
2. [SerializeField] MainMenuNavigation MMN;
3. [SerializeField] bool useTemplatedGeneration;
4. [SerializeField] int StagnationTerminationAmt;
5. InLoopFitnessBase[] fitnesses;
6. void algorithmStart(){
7. fitnesses = GetComponents<InLoopFitnessBase>();
8. useTemplatedGeneration = MainMenuNavigation.isTemplate;
9. for (int i = 0; i < fitnesses.Length; i++){
10. //Ngilangin Fitness yang ngga dipakai
11. if (!fitnesses[i].IsUsed || (useTemplatedGeneration && !fitnesses[i].ForTemplateGen) || (!useTemplatedGeneration && !fitnesses[i].ForTileGen))
12. fitnesses[i] = null;
13. }
14. fitnesses = fitnesses.Where(f => f != null).ToArray();
15. double fitness;
17. int generatedMapLength = SetObjects.getHeight() \* SetObjects.getWidth() / 2;
18. //Multithreading
19. var taskExecutor = new ParallelTaskExecutor();
20. taskExecutor.MinThreads = 12;
21. taskExecutor.MaxThreads = 12;
22. ChromosomeBase chromosome;
23. //Kromosom
24. if (useTemplatedGeneration)
25. chromosome = new TemplatedMapChromosome(Mathf.RoundToInt(generatedMapLength / 25));
26. else{
27. chromosome = new GameChromosome(generatedMapLength, Mathf.FloorToInt(GetComponent<PowerUpRatioFitness>().getRatio() \* generatedMapLength/2));
28. }
29. //Populasi
30. var population = new Population(50, 100, chromosome);
31. //Fitness
32. var fitnessfunc = new FuncFitness((c) =>{
33. var fc = c.GetGenes();
34. int[,] map;
35. //Menghasilkan map utuh yang siap diperiksa oleh fitness
36. if (useTemplatedGeneration)

Segmen Program 6.X (Lanjutan)

1. map = putPlayerinTemplate(fc, SetObjects.getWidth() / 2, SetObjects.getHeight());
2. else
3. map = deflatten(fc, SetObjects.getWidth() / 2, SetObjects.getHeight());
4. fitness = fitnessFunction(map, fc);
5. return fitness;
6. });
7. //Metode milih ortu
8. var selection = new RouletteWheelSelection();
9. //Metode Crossover
10. var crossover = new UniformCrossover();
11. var mutation = new PartialShuffleMutation();
12. var termination = new FitnessStagnationTermination(StagnationTerminationAmt);
13. var ga = new GeneticAlgorithm(population, fitnessfunc, selection, crossover, mutation);
14. ga.Termination = termination;
15. ga.Start();
16. var a = ga.BestChromosome.GetGenes();
17. if (useTemplatedGeneration)
18. SetObjects.setMap(putPlayerinTemplate(a, SetObjects.getWidth() / 2, SetObjects.getHeight()), useMirrorFitness);
19. else
20. SetObjects.setMap(deflatten(a, SetObjects.getWidth() / 2, SetObjects.getHeight()), useMirrorFitness);
21. MMN.changeSceneIndex(-6);
22. }

Berikut penjelasannya :

* 1 : Mengatakan kepada GA bila fitness diukur berdasarkan hasil akhirnya, karena hasil generasi dari GA hanyalah sebelah kiri dari sebuah level, yang nantinya akan dicerminkan di sebelah kanan, membuat level yang jadi.
* 2 : Referensi Script MainMenuNavigation agar setelah Genetic Algorithm selesai bisa berganti level
* 3 : Menentukan apabila Genetic Algorithm menggunakan Template atau Tile Generation
* 4 : Menentukan berapa generasi tanpa perkembangan fitness sebelum Genetic Algorithm menyerah dalam menyari kromosom terbaik.
* 5 : Variabel yang akan menampung semua fitness
* 8 : Mengambil semua fitness yang tersedia untuk dipakai
* 9 : Mengambil data apabila player memilih Map Template atau tile
* 10 – 15 : Mengecek semua fitness yang ada di variabel fitnesses, lalu menghapus fitness yang ditentukan tidak cocok dengan permintaan player.
* 19 : Menentukan variabel MapLength yang berisi luas level / 2 dikarenakan itu adalah ketentuan jumlah Gene dalam sebuah kromosom.
* 21 - 24 : Digunakan untuk melakukan Multirhreading terhadap Genetic Algorithm, tetapi kurang diketahui bila potongan kode ini bekerja atau tidak.
* 28 – 32 : Mengecek apabila player meminta Template atau Tile level generation, dan membuat kromosom sesuai dengan permintaan player tersebut.
* 35 : Menentukan jumlah populasi dari kromosom yang ditentukan. Dalam game ini ditentukan populasi awal adalah 50 sementara populasi maksimum adalah 100.
* 37 – 47 : Menentukan fungsi fitness yang akan dipakai pada Genetic Algorithm.
* 38 – 44 : Variabel fc akan diisi dengan seluruh Gene dalam kromosom, lalu gene tersebut akan diubah menjadi sebuah array level yang berbentuk 2D sesuai dengan input player
* 45 – 46 : Variabel fitness akan diisi skor fitness menggunakan function fitnessFunction yang akan dijelaskan pada subbab berikutnya, lalu fitness itu akan direturn untuk dilakukan tahap seleksi.
* 49 : Menyuruh GA untuk memakai Roulette Wheel Selection.
* 52 : Menyuruh GA untuk memakai Uniform Crossover.
* 53 : Sama kayak diatas, GA disuruh memakai Partial Shuffle Mutation.
* 54 : Sama aja, GA memakai Fitness Stagnation Termination dengan toleransi sebanyak yang ditentukan StagnationTerminationAmt yang ada di baris 4.
* 56-57 : Menggabung semua aspek dari Genetic Algorithm (Populasi yang dibuat, Crossover yang dipilih dst) menjadi satu Genetic Algorithm yang siap dipakai.
* 59 : Genetic Algorithm mulai dijalankan.
* 61 : Variabel a akan diisi Kromosom terbaik yang dihasilkan oleh Genetic Algorithm
* 62 – 65 : Menggunakan script SetObjects, ubah kromosom yang didapat pada line 61 menjadi sebuah map jadi, sekali lagi menggunakan function berbeda sesuai dengan pilihan level generation player.
* 66 : Setelah semuanya selesai, maka akan di load scene level utama agar player bisa memainkan level yang degenerate.

1. Function FitnessFunction

Di potongan kode 6.X baris 45 diatas terdapat sebuah function yang bernama FitnessFunction, dimana fungsi ini bertanggung jawab dalam menentukan apabila sebuah kromosom tergolong bagus atau tidak. Sekarang untuk penjelasan dari potongan kode FitnessFunction tersebut :

Segmen Program 6.X Fungsi FitnessFunction

1. double fitnessFunction(int[,] map , Gene[] original){
2. int playerAmount = 0;
3. InLoopFitnessBase[] currentFitnesses = (InLoopFitnessBase[])fitnesses.Clone();
4. //Khusus Template Variety
5. for (int i = 0; i < currentFitnesses.Length; i++)
6. if (currentFitnesses[i].GetType() == GetComponent<TemplateVarietyFitness>().GetType())
7. ((TemplateVarietyFitness)currentFitnesses[i]). getTemplateMap(original);
9. double[] fitnessScores = new double[fitnesses.Length];
10. foreach (InLoopFitnessBase item in currentFitnesses)
11. item.resetVariables();
12. Coordinate tempCoor;
13. for (int i = 0; i < SetObjects.getHeight(); i++)
14. for (int j = 0; j < SetObjects.getWidth(); j++) {
15. tempCoor = new Coordinate(j, i);
16. if (map[tempCoor.yCoor, tempCoor.xCoor] == 3)
17. playerAmount++;

Segmen Program 6.X (Lanjutan)

1. foreach (InLoopFitnessBase item in currentFitnesses)
2. item.calculateFitness(map,tempCoor);
3. }
4. for (int i = 0; i < currentFitnesses.Length; i++)
5. fitnessScores[i] = currentFitnesses[i].getFitnessScore();
6. return Mathf.Pow((float)fitnessScores.Sum() / Mathf.Pow(MathF.Abs(playerAmount - 10) \* 5 + 1, 3), 3);
7. }

Berikut adalah penjelasannya setiap baris :

* 1 : Function ini memerlukan 2 parameter, yaitu map jadi level dan juga Kromosom originalnya.
* 3 : Dari variabel global fitnesses yang sebelumnya telah diisi di baris 10 – 15 kode utama Genetic Algorithm, diduplikat fitness tersebut ke dalam variabel currentFitnesses agar setiap kromosom memiliki perhitungan fitness yang tidak saling mengganggu.
* 6 – 8 : Bila terdapat fitness TemplateVariety dalam currentFitness, maka fitness tersebut akan diberikan kromosom original yang hanya berisi id – id template.
* 11 – 12 : Setiap fitnesses yang ada akan di reset variabel internalnya agar memulai dari nilai – nilai default yang ditentukan.
* 14 – 17 : Setiap tile dalam level akan diakses dan disimpan koordinatnya.
* 18 – 19 : Bila tile yang dipilih mengandung sebuah player maka jumlah player akan ditambah.
* 20 – 21 : Setiap fitness yang dimiliki akan melakukan calculateFitness, dimana calculataFitness adalah sebuah fungsi yang bisa berfungsi untuk menyimpan dan mengolah data untuk membantu dalam menghitung nilai fitness.
* 24 – 25 : Setiap fitness yang ada akan dikalkulasikan nilai fitness akhirnya, lalu setelah fitness tersebut dihitung, maka fitness tersebut akan dikembalikan ke array fitnessScores.
* 26 : Seluruh fitness akan dijumlah dan akan didapat nilai fitness total, tetapi sebelum direturn akan dilihat bila karakter berjumlah 10, bila jumlah karakter tidak berjumlah 10 maka fitness akan dibagi sehingga memiliki fitness yang jauh lebih kecil.

1. Fitness Individual

Segmen Program 3.1 Contoh Segmen Program Cek Prima

1. SREAD N
2. C = 0
3. FOR I = 2 TO N-1
4. IF N mod I = 0
5. C = C + 1
6. END IF

**Segmen Program 3.1 (Lanjutan)**

1. END FOR
2. IF C = 0
3. WRITE “PRIMA”
4. ELSE
5. WRITE “BUKAN PRIMA”
6. END IF