BAB II

GENETIC ALGORITHM

Dalam bab ini akan dijelaskan 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 banyak lagi. [[1]](#footnote-1)

1. Terminologi

Di dalam bab Genetic Algorithm, akan ada beberapa istilah yang berhubungan dengan biologi, seperti Kromosom dan Gene. Dikarenakan itu, akan ada beberapa kata yang jarang ditemukan di dalam kehidupan sehari – hari. Jadi sebelum mendalami Genetic Algorithm, perlu diketahui beberapa istilah asing yang akan sering dijumpai dalam bab ini, yaitu;

1. Gene & Chromosome (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 / Generasi

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. Generasi adalah nama lain dari sebuah populasi dimana Generasi menampilkan iterasi dari populasi tersebut. Seperti sebuah Generasi ke-7 menandakan bahwa Generasi tersebut adalah populasi ke-7 yang dibuat oleh Genetic Algorithm.



Gambar 2.1

Visualisasi Gene, Kromosom dan Populasi

1. Fitness

Fitness adalah sebuah angka yang menunjukkan kualitas jawaban dari sebuah kromosom, cenderung semakin tinggi Fitness-nya semakin baik kualitasnya. Semisal ada Kromosom pertama dengan Fitness 50 dan Kromosom ke-3 dengan Fitness 300, maka bisa kelihatan jelas bila Kromosom ke-3 lebih baik daripada kromosom pertama, tetapi bisa diatua apabila bila fitness semakin rendah menandakan bila solusi dari Chromosome tersebut semakin bagus tergantung dari permasalahannya. Nilai Fitness dapat diperoleh menggunakan Fitness Function buatan kita sendiri yang akan dijelaskan di dalam subbab selanjutnya.

1. Parent & Child Chromosome

Parent Chromosome adalah kromosom yang dipilih berdasarkan Fitnes kromosom itu sendiri, dimana parent tersebut akan dipasangkan dengan parent lain untuk membuat sebuah anak kromosom baru. Anak tersebut dipanggil Child Chromosome dan akan menjadi parent dalam Generasi selanjutnya.

1. Metodologi

Genetic Algorithm didasarkan pada seleksi alam, dimana didalam sebuah popuulasi spesies hewan, hanya hewan dengan fitur tertentu yang akan bertahan hidup untuk waktu yang lama. Dari konsep tersebut Genetic Algorithm akan menciptakan sejumlah kandidat solusi yang pada algoritma ini akan berevolusi dengan harapan setiap generasi solusi yang dibuat akan selalu berkembang dan lebih bagus daripada solusi generasi sebelumnya.[[2]](#footnote-2)



Gambar 2.2

Visualisasi Garis besar cara kerja Algoritma Genetik

Di gambar 2.2 diatas, ditunjukkan secara garis besar apa yang terjadi di dalam sebuah Algoritma Genetik. Di saat program dimulai dalam “Initial Population” akan dibuat Generasi pertama, biasanya dengan Gene yang isinya acak. Setelah dibuat maka akan diambil nilai Fitness dari setiap kromosom di Generasi pertama tersebut yang diwakilkan dengan tag “Calculate Fitness” diatas.

Setelah itu dilakukan “Selection” yang memilih parent – parent dengan dasar nilai fitness setiap kromosom dimana cenderung semakin tinggi fitness, semakin mungkin dipilih. Lalu terdapat “Crossover”, dimana setiap pasang parent akan membuat anak kromosom baru yang berisi Gene parent-nya. Pemilihan parent lalu pembuatan anak diulangi hingga dapat mengisi sebuah populasi penuh dengan Kromosom – kromosom baru, setelah itu dilakukan “Mutation” kepada seluruh kromosom-kromosom yang dibuat untuk manambah variasi dari generasi baru. Tahap terakhir yang tidak terdapat di gambar adalah Elitism, dimana beberapa dari kromosom-kromosom dengan fitness terbaik dari generasi sebelumnya diletakkan pada generasi baru untuk mempreservasi fitness terbaik.

Setelah generasi ke-2 baru telah dibuat, proses yang tadi dilakukan akan diulangi mulai dari menghitung Fitness setiap kromosom. Pengulangan ini akan terjadi terus sampai sebuah syarat tertentu tercapai, seperti jumlah Generasi maksimal, Fitness terbaik sudah mencapai nilai tertentu, Fitness tidak membaik setelah banyak generasi, dan banyak lagi. Tapi sebelum syarat berhenti tercapai, algoritma Genetik akan berjalan terus.

1. Langkah-langkah Algoritma Genetik

Setelah dijelaskan Metodologi dan seluruh Terminologi yang diperlukan, maka akan dilanjutkan dalam subbab ini. 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.

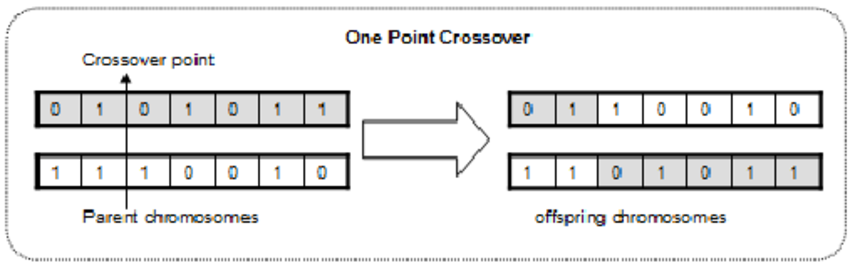
1. Seleksi

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 2.3

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.



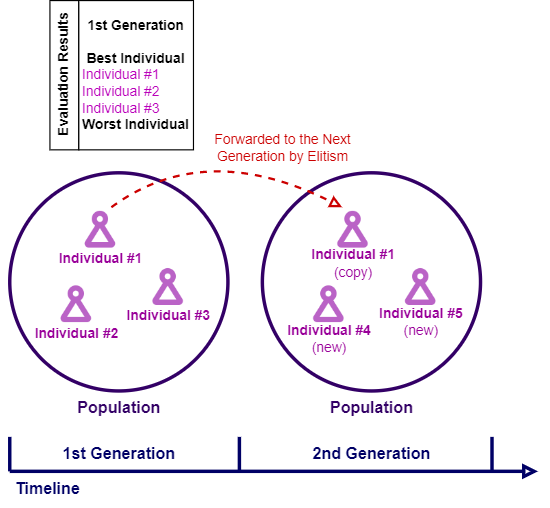
Gambar 2.4

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 2.5

Visualisasi konsep Elitism

1. Repetisi dan Pemberhentian

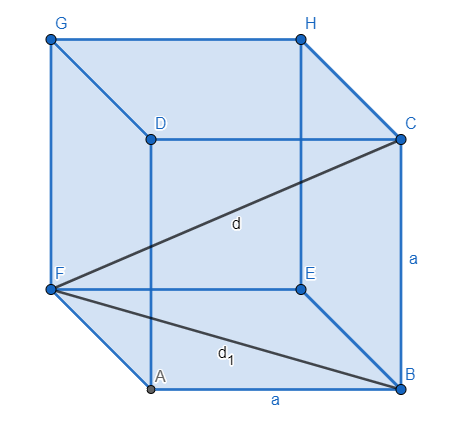
Setalah tahap – tahap di atas telah diselesaikan, algoritma genetik telah membuat sebuah generasi baru yang memiliki harapan nilai Fitness lebih baik daripada generasi sebelumnya. Setelah generasi baru dibuat, maka algoritma genetik akan kembali ke tahap ke-2 yaitu seleksi, maju ke tahap ke-3 reproduksi 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. Library GeneticSharp

Dalam tugas akhir ini, akan dipakai 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 diberi contoh penggunaan GeneticSharp beserta sebuah contoh sederhana.

1. Permasalahan

Permasalahan yang akan kita pakai sebagai contoh adalah mencari Panjang garis terpanjang dalam sebuah kubus tiga dimensi dengan sisi 10 cm. Bagi kita yang sudah mendalami matematika sudah mengetahui bila garis terpanjang dalam sebuah kubus 3D adalah sebuah garis diagonal yang melalui titik tengah kubus. Tetapi sang AI tidak mengetahui itu, AI hanya mengetahui panjang sisi kubus dan harus mencari nilai XYZ dari kedua titik yang akan membuat garis kita.

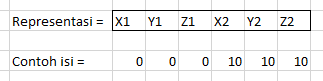


Gambar 2.6

Sebuah Kubus dan 2 garis diagonal

1. Representasi

Representasi dari permasalahan di atas akan menggunakan array integer 1 dimensi dengan panjang 6. Setiap isi array akan diisi dengan sebuah integer yang akan merepresentasikan entah X,Y atau Z dari satu titik. Berikut adalah representasi yang akan dipakai untuk permasalahan ini :



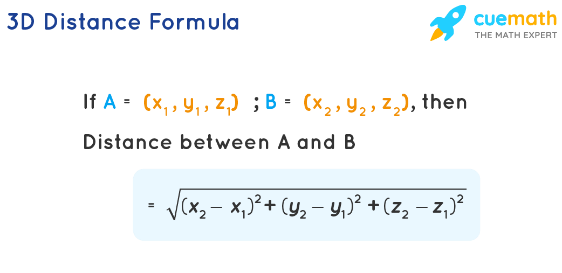
Gambar 2.7

Contoh Representasi permasalahan kubus

Seperti yang bisa dilihat dari gambar 2.7, Representasi yang akan kita pakai dalam permasalahan ini adalah XYZ titik pertama dalam 3 nilai array pertama, dan XYZ titik 2 di 3 nilai array terakhir. Berdasarkan contoh isi di atas tersebut, dari representasi dibawah akan didapat titik 1 dengan posisi 0,0,0 sementara titik ke-2 akan memiliki posisi 10,10,10. Perlu ditegaskan bila ini bukanlah cara yang “paling benar” untuk masalah ini, urutannya bisa diatur sebagai XXYYZZ, atau bisa juga XYZZYX, dan itu juga merupakan sebuah representasi yang valid.

1. Fitness Function

Fitness Function atau fungsi Fitness adalah sebuah fungsi yang dapat digunakan untuk mendapatkan nilai Fitness dari sebuah Chromosome. Menggunakan contoh permasalahangaris terpanjang dari 2 titik dalam sebuah kubus , titik kita bisa berada di mana mana dalam sebuah kubus, jadi tidak cocok untuk menggunakan rumus standar seperti sisi kubus sebagai fungsi fitness kita. Sebagai pengganti yang lebih layak, kita akan menggunakan Euclidian distance dari 2 titik XYZ sebagai fungsi Fitness kita. Rumus tersebut dapat dilihat di dalam gambar 2.8.



Gambar 2.8

Rumus menghitung jarak antara 2 titik di dalam kubus

Dari rumus ini, dapat diambil panjang dari garis yang dibuat, dan panjang tersebut akan kita gunakan sebagai fitness value kita. Dikarenakan Genetic Algorithm kita mengambil kromosom dengan fitness value terbanyak di sebuah populasi, maka semakin panjang garis yang dibuat dari kedua titik tersebut, maka akan semakin besar kemungkinan kromosom tersebut dipilih untuk beranak.

1. Metode – metode pemrosesan data

Bila dilihat dari penjelasan di bab 2.3, ada 3 proses yang memiliki lebih dari 1 metode, yaitu Seleksi,Crossover dan Mutasi. Ini adalah metode yang dipilih untuk ketiga proses dalam permasalahan ini:

* 1. Seleksi

Roulette Wheel Selection, dimana akan dilakukan pemilihan random seorang parent seperti memutar roda Roulette, tapi semakin besar fitness kromosom tersebut, maka akan semakin besar kemungkinan kromosom tersebut dipilih, bisa dianggap seperti membesarnya bagian kromosom tersebut di dalam roda Roulette.

* 1. Crossover

Uniform Crossover, dimana setiap gene dalam sebuah kromosom dapat ditukar dengan kemungkinan dasar 50%. Kemungkinan ini dapat diganti oleh user bila diinginkan.

* 1. Mutasi

Uniform Mutation, mirip dengan Uniform Crossover dimana setiap gene memiliki kemungkinan untuk berubah. Bila ada gene yang berubah, maka akan diambil nilai baru secara acak dari range yang telah ditentukan. Kemungkinan dari sebuah gene melakukan mutasi di Uniform Mutation secara bawaan adalah 1/Jumlah Gene dalam kromosom.

1. Pseudocode dan Penjelasan

Sekarang dengan seluruh persiapan yang telah dilakukan, kita dapat melihat pseudocode dibawah dan melihat kegunaan dari setiap baris kode :

Algoritma 2.1 Pseudocode Library GeneticSharp

1. Menggunakan library GeneticSharp.Domain.\*
2. varKromosom = new FloatingPointChromosome(
3. Array batas minimum gene => [0,0,0,0,0,0],
4. Array batas maximum gene => [10,10,10,10,10,10],
5. Array jumlah bit gene => [3,3,3,3,3,3],
6. Array jumlah bilangan desimal => [0,0,0,0,0,0]
7. )
8. varPopulasi = new Population(
9. minPopulasi,maxPopulasi,kromosom => varKromosom
10. )
11. varFungsiFitness = new FuncFitness((c)=>
12. kromosomSekarang = c as FloatingPointChromosome
13. arrKromosom = kromosomSekarang.ToFloatingPoints()
14. X1 = arrKromosom[0]
15. Y1 = arrKromosom[1]
16. Z1 = arrKromosom[2]
17. X2 = arrKromosom[3]
18. Y2 = arrKromosom[4]
19. Y2 = arrKromosom[5]
20. euclDist = Akar2(Pangkat2(X1-X2)+ Pangkat2(Y1-Y2) + Pangkat2(Z1-Z2))
21. return euclDist
22. )
23. varMetodeSeleksi = new RouletteWheelSelection()
24. varCrossover = new UniformCrossover(kemungkinan crossover terjadi {0-1})
25. varMutasi = new UniformMutation()
26. varStopCon = new GenerationNumberTermination(100);
27. varGA = new GeneticAlgorithm(
28. Populasi => varPopulasi,
29. Fitness => varFungsiFitness,
30. Metode Seleksi => varMetodeSeleksi,
31. Crossover => varCrossover,
32. Mutasi => varMutasi
33. )

Algoritma 2.1 (Lanjutan)

1. varGA.Termination = varStopCon
2. varGA.Start();
3. hasilJadi = varGA.BestChromosome

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

1. Line 2-7

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, dalam kasus ini akan diisi array dengan panjang 6 berisi 0 dan 10 untuk batas min dan max secara berurutan.

Parameter selanjutnya adalah jumlah bit dikarenakan GeneticSharp menggunakan mewarisi Class BinaryChromosomeBase yang hanya menggunakan angka biner, dikarenakan 10 hanya memerlukan 3 bit maka parameter ini diisi 3. Parameter terakhir adalah jumlah angka desimal apabila Gene yang diinginkan ingin memiliki angka koma, kita tidak menggunakan angka desimal jadi diisi 0.

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 8-10 :

Di Line ini ditentukan jumlah minimum dan maximum jumlah kromosom dari sebuah populasi. Minimum kromosom akan dipakai dalam inisialisasi pembuatabn generasi pertama, sementara maximum kromosom adalah batas maksimal kromosom dalam setiap generasi. Lalu terakhir di parameter ketiga adalah kromosom yang telah dibuat di line 3.

1. Line 11–22

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 12 dan 13, lalu dengan array kromosom tersebut, kita bisa mengambil nilai XYZ dari kedua titik berdasarkan representasi yang telah ditentukan sebelumnya. Dengan XYZ dari kedua titik diambil, kita dapat mencari Euclidian Distance dari kedua titik tersebut. dan setelah mendapat Euclidian distance, data tersebut akan di return untuk dipakai dalam selection.

1. Line 23

Line ini dipakai untuk menentukan metode Seleksi kromosom, dalam kasus kita Roulette Wheel Selection. 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.

1. Line 24

Di sini dipilih Crossover yang akan dipakai oleh Algoritma Genetik, dalam kasus kita Uniform Crossover,dimana 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.

1. Line 25

Sekarang baris kode ini dipakai untuk memilih mutasi yang akan digunakan dalam algoritma genetik, kita akan memakai Uniform Mutation. Beberapa pilihan yang lain diantaranya adalah ; Flip-Bit, Reverse Sequence dan Twors. Mutasi buatan sendiri harus mewarisi Class MutationBase.

1. Line 26

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. Tapi dalam kasus ini kita akan memilih Generation Number Terminantion dimana setelah 100 iterasi Algoritma Genetik akan berhenti. Seperti biasa dapat menggunakan Class sendiri yang mewarisi Class TerminationBase.

1. Line 27 – 35

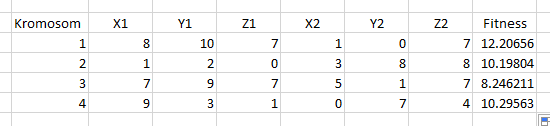
Di sini seluruh pilihan dan fungsi yang telah dibuat sebelumnya dimasukkan dalam Constructor Genetic Algorithm yang baru ini, mulai dari populasi hingga mutasi yang ada. Lalu di line 35 algoritma genetik sudah mulai berjalan dan mulai mencari solusi

1. Line 36

Setelah Genetic Algorithm selesai dijalankan, maka hasilnya akan disimpan dalam sebuah variabel yang ditentukan. Hasil ini dapat diakses untuk menunjukkan jawaban terbaik yang telah ditemukan oleh Algoritma Genetik dan dapat diaplikasikan ke masalah yang diperlukan.

1. Tracing

Dengan pseudocode yang sudah dijelaskan, kita akan melihat apa yang terjadi di dalam Algoritma genetik dengan melakukan tracing (melaksanakan kode menggunakan tangan kita sendiri) terhadap algoritma Genetik selama 1 generasi dengan 5 kromosom dalam tiap populasi (karena kalau kita menggunakan 100 kromosom per populasi pasti akan makan waktu yang sangat lama). Untuk inialisasi, akan diambil angka acak untuk setiap angka dalam gene. Hasilnya adalah sebagai berikut :



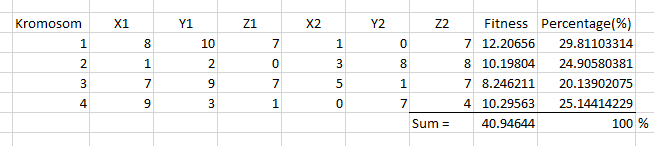
Gambar 2.9

Generasi Pertama

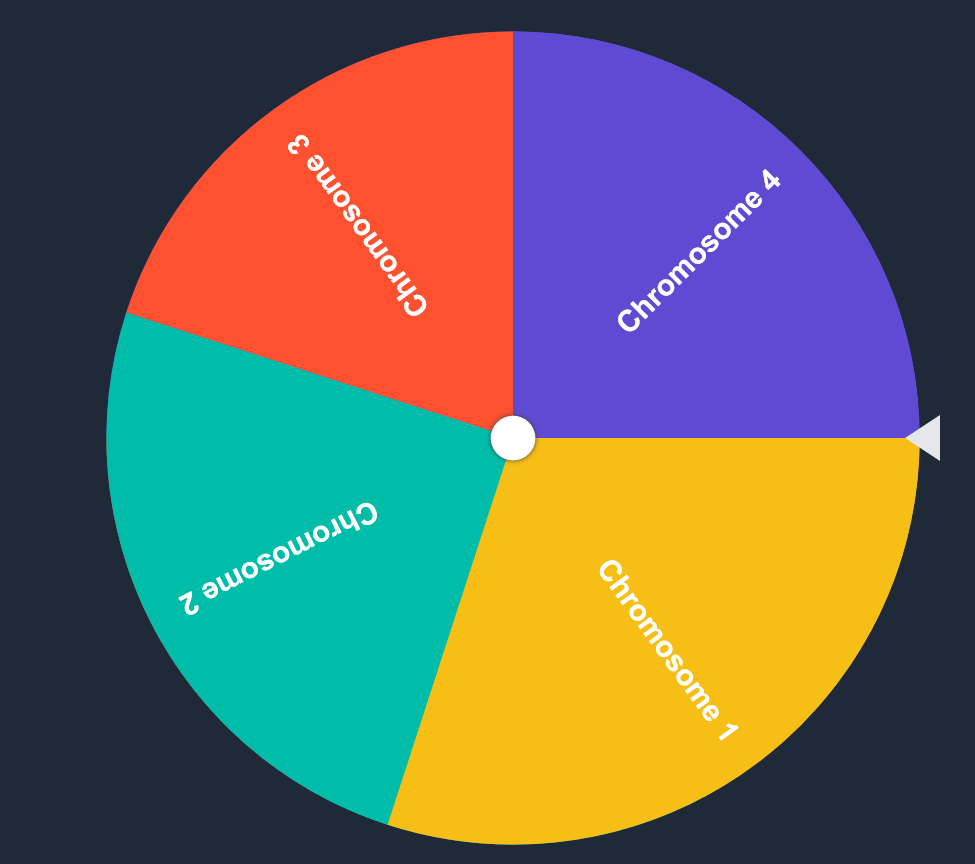
Dengan angka yang didapat secara acak, kita dapat menghitung Fitness dari ke-empat kromosom tersebut. Kita akan coba menghitung Kromosom yang pertama dengan metode Euclidian Distance, seperti berikut;

(2.1)

Setelah menghitung fitness dari kromosom pertama, kita dapat melakukan hal yang sama terhadap 3 kromosom lainnya. Dimana hasilnya telah tertera di gambar 2.10 sebelah kanan. Dengan Fitness yang didapat, kita akan memasukkan ke-4 Fitness tersebut ke dalam sebuah roda Roulette secara metafor. Kita akan mencari kemungkinan persentase dipilihnya setiap kromosom berdasarkan total fitness dari keseluruhan kromosom.

****

(a)

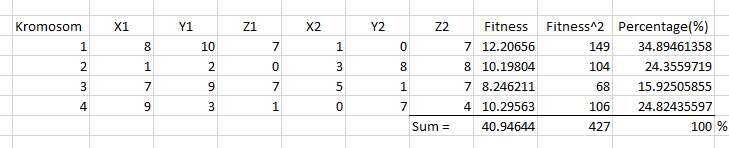
****

**(b)**

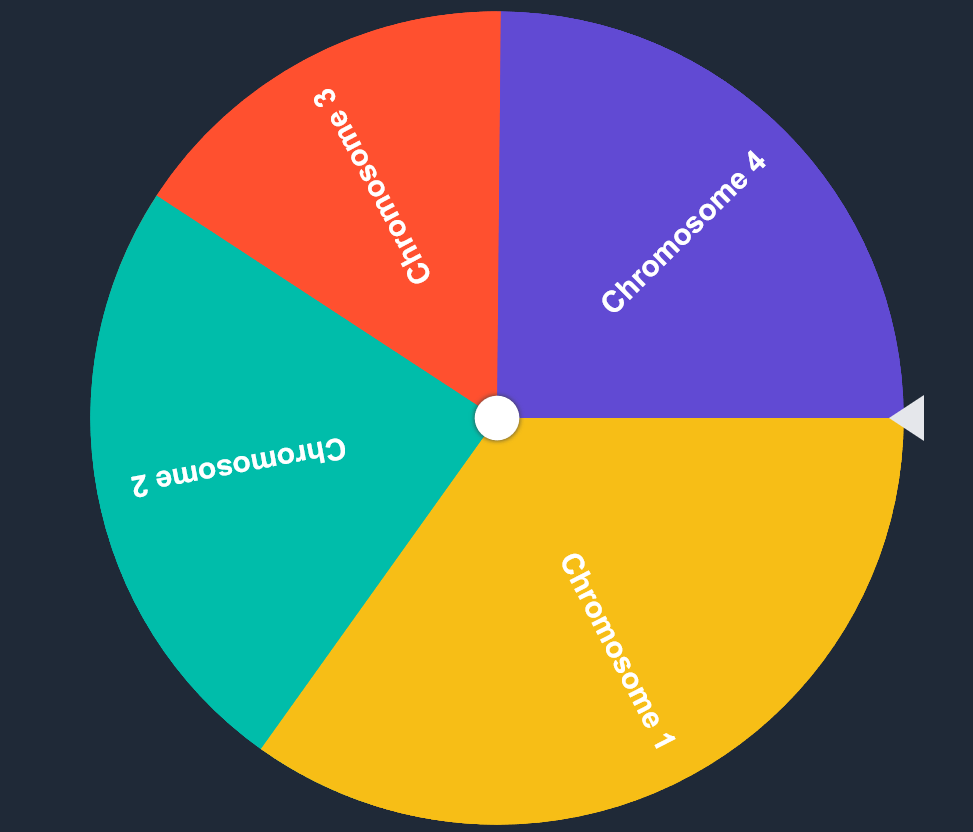
Gambar 2.10

**(a) Persentase seluruh Populasi dan (b) Visualisasi roda Roulette**

Persentase dari setiap kromosom dapat dilihat di gambar 2.10 beserta dengan visualisasi roda Roulette-nya. Bisa dilihat bila Kromosom 1 memikiki daerah yang lebih besar dibandingkan dengan kromosom yang lainnya, khususnya kromosom 3 yang hanya memiliki persentase 20%. Tapi meskipun begitu, bisa disadari perbedaan dari tiap kromosom tidak terlalu mencolok, sehingga kromosom 3 masih memiliki kemungkinan yang bagus untuk dipilih. Metode untuk menangani ini adalah dengan memberi pangkat terhadap fitness kita, sehingga perbedaan antar fitness dapat lebih mencolok lagi. Seperti berikut;



(a)

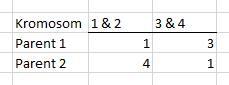


**(b)**

Gambar 2.11

**(a) Persentase dan (b) Visualisasi Fitness kuadrat**

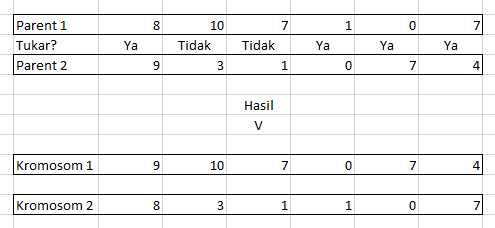
Dengan menghitung persentasi Fitness kuadrat, maka visualisasi yang didapat lebih menguntungkan Kromosom 1, dan Kromosom 3 lebih kecil kemungkinannya untuk dipilih. Fitness bisa diberi pangkat berapapun sesuai keperluan user, tapi bagi kita pangkat 2 saja sudah cukup. Tahap selanjutnya adalah menggunakan roda roulette ini, pilih parent kromosom untuk generasi selanjutnya. Jadi pilih sepasang kromosom sebanyak 2 kali dikarenakan sepasang parent membuat 2 anak. Ingat bila sepasang kromosom tersebut tidak boleh kromosom yang sama. Dan inilah hasil yang kita dapat :

****

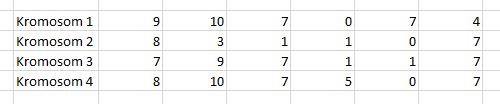
Gambar 2.12

**Hasil Memutar roda Roulette**

Dan dengan parent dipilih, kita dapat memasuki tahap selanjutnya yaitu Crossover, dimana sepasang parent akan menukar gene satu sama lain. Kita menggunakan metode Uniform Crossover,yang telah dijelaskan.. Dan dengan itu kita akan melakukan tracing terhadap pasangan kromosom yang pertama. Inilah hasil yang didapat dari Crossover tersebut :



**(a)**

****

**(b)**

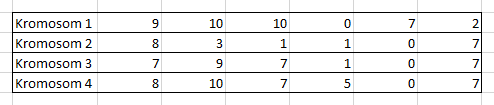
Gambar 2.13

**Crossover (a) pasangan kromosom pertama dan (b) hasil keseluruhan**

Tahap terakhir setelah membuat generasi baru adalah untuk melakukan mutasi terhadap setiap Kromosom yang ada. Mutasi kita adalah Uniform Mutation dimana setiap gene yang terpilih akan berubah menjadi gene baru berdasarkan batas yang kita tentukan di awal. Kita akan melakukan Mutation ini terhadap kromosom baru yang pertama. Perlu diingat karena satu kromosom memiliki 6 gene, maka setiap gene akan memiliki kemungkinan untuk melakukan mutasi.



**(a)**



**(b)**

Gambar 2.14

**Mutasi (a) kromosom pertama dan (b) hasil keseluruhan**

Setelah melakukan mutasi terhadap kromsom 1,dilakukan hal yang sama terhadap 3 kromosom lainnya. Dalam tracing ini, yang berubah hanya gene ke 5 di Kromosom 3. Setelah mutasi dilakukan, maka Generasi ke – 2 ini siap dipakai untuk menemukan generasi ke-3. Seharusnya di dalam tracing ada proses Elitism, tetapi oleh GeneticSharp Elitism dijadikan metode Seleksi dimana Kromosom terbaik akan dipilih duluan dan Sebagian kecil dari kromosom – kromosom terbaik akan dibawa ke generasi selanjutnya.

Dan dengan tahap mutasi selesai, Generasi ke – 2 siap memasuki tahap seleksi, dilanjutkan dengan Crossover lalu diakhiri dengan Mutasi. Tahap ini akan berulang – ulang terus hingga mencapai 100 generasi. Selama itu, kromosom terbaik akan selalu diperbarui dan disimpan, jadi setelah Algoritma Genetik selesai, kromosom terbaik dapat diambil dan dapat dipakai untuk masalah yang ada, dalam kasus ini menentukan garis terpanjang dalam sebuah kubus.

1. Xin-She Yang, Yang. *Genetic Algorithms. Nature-Inspired Optimization Algorithms.* X.-S. (2014) [↑](#footnote-ref-1)
2. Vijini Mallawaarachchi, *Introduction to Genetic Algorithms — Including Example Code* (https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3, diakses 27 Januari 2023) [↑](#footnote-ref-2)