BAB 2

LANDASAN TEORI

2.1 Game Desktop

Game Berbasis Desktop merupakan permainan video yang bisa berjalan secara independen atau berjalan sendiri dalam sistem operasi tertentu dengan cara menginstallasi (memasamg aplikasi) dalam sistem operasi komputer (PC) sehingga bisa berjalan tanpa membutuhkan jaringan internet. Game desktop ini mulai dikenal seiring munculnya sistem operasi windows, karena setiap periode sistem operasi windows yang diluncurkan di dalamnya terdapat permainan yang melengkapi hiburan berbentuk multimedia untuk para penggunanya. Game sendiri diambil dari bahasa Inggris yang diterjemahkan yang artinya permainan. Di dalam era perkembangan era perkembangan teknologi yang pesat ini, seperti halnya permainan kartu, catur dan lain-lainnya dapat ditemui melalui dunia virtual atau yang biasa kita mainkan di dalam komputer. Dengan perkembangan teknologi sekarang ini munculah berbagai sarana permainan, misalnya Playstation 3, Xbox 360, Nintendo Wii, PSP, Nintendo DS, maupun PC dari yang berbasis individu ataupun multiplayer [3].

2.2 Definisi game

Berdasarkan data penelitian dari Longitudinal Study of Australian Children, para ilmuwan menemukan kalau anak yang menghabiskan waktu dengan bermain video game akan mengalami peningkatan daya kognitif yang lebih baik daripada anak yang menghabiskan waktu dengan menonton televisi [4].

Sebuah video game adalah permainan yang biasanya melibatkan player berinteraksi dengan alat pengendali untuk menghasilkan umpan balik secara visual dalam sebuah layar video. Perangkat input yang digunakan untuk memanipulasi permainan video disebut pengendali permainan, dan bervariasi di seluruh platform. Sebagai contoh, controller mungkin hanya terdiri dari sebuah tombol dan joystick, sementara yang lain mungkin mempunyai selusin

tombol dan satu atau lebih joystick. Video game biasanya menggunakan sarana tambahan menyediakan interaktivitas dan informasi untuk pemain. Audio hampir universal, menggunakan perangkat reproduksi suara, seperti speaker dan headphone. Umpan balik lain mungkin datang melalui peripheral haptic, seperti getaran atau kekuatan, dengan getaran kadang-kadang digunakan untuk mensimulasikan umpan balik kekuatan.

2.3 Genre Game

Saat ini sangat banyak kita temukan game, baik itu game PC, game console (seperti PS, XBOX, PSP, Nintendo DS), game HP (mobile game) dan game flash. Dari banyaknya game yang ada, berdasarkan gameplaynya (cara memainkannya), game dapat diklasifikasikan menjadi beberapa jenis. Dalam hal ini klasifikasi game flash bisa dilihat sebagai berikut [4]:

1) *Board game*

Board game atau dapat kita istilahkan sebagai "permainan papan". Pada jenis game ini, pemain diberikan sebuah tampilan yang berisi tentang masalah untuk diselesaikan. Contoh dari game ini antara lain: Pipe dream, Hangaroo, Rotation, Catur, permainan kartu dan sebagainya.

2) *Arcade*

Game bertipe arcade merupakan game yang menguci kecepatan tangan dari pemainnya. Pada permainan bertipe arcade, semakin tinggi level permainan, permainan akan berjalan semain cepat. Contoh dari game bertipe arcade adalah : zuma, feeding frenzy, Pacman, Arcanoid, Pong, Baba ball dan sebagainya.

3) *Action*

Berbeda dengan game bertipe aarcade, game bertipe action menjadikan pemain mengendalikan karakter utama dalam game tersebut untuk melakukan beberapa kegiatan (action) seperti melompat, menembak dan sebagainya. Contoh dari game bertipe action antara lain : super mario, mocil, petualangan paddle pop, alloy tease exile dan sebagainya.

4) Shooting

Game bertipe shooting sebagian besar menggunakan mouse sebagai alat pengendalinya. Pada game ini pemain seolah-olah berperan sebagai penembak (first person) atau pemain mengendalikan seorang penembak (third person). 4. ShootingGame bertipe shooting sebagian besar menggunakan mouse sebagai alat pengendalinya. Pada game ini pemain seolah-olah berperan sebagai penembak (first person) atau pemain mengendalikan seorang penembak (third person).Contoh game bertipe shooting antara lain : Duck hunt, Counter Strike, Onimusha dan sebagainya.

5) Fighting

Game bertipe fighting pada dasarnya sama dengan game bertipe action, hanya saja game bertipe fighting pemain mengendalikan sebuah karakter untuk berkelahi dengan karakter lain sampai salah satu karakter kalah. Contoh dari game bertipe fighting antara lain : street fighter, tekken, duel, pencak silat dan sebagainya.

6) Racing

Game betipe racing pada dasarnya adalah sebuah permainan menggerakkan kamera. Pemain diberikan sebuah kendaraan atau sejenisnya untuk menempuh rute tertentu. Contoh dari game bertipe racing antara lain : NFS, Auto bahn, MotoGP, Formula 1 dan sebagainya.

7) *Simulation*

Game bertipe simulasi adalah sebuah game yang mensimulasikan suatu kegiatan yang sesungguhnya. Contoh dari game bertipe simulasi antara lain : tycoon, simulator pesawat, burger empire dan sebagainya.

8) *Real Time Strategi*

Game bertipe RTS memposisikan pemain sebagai seorang pemimpin yang mengatur sesuatu (bisa berupa pasukan, koloni, kerajaan dan sebagainya). Contoh dari game bertipe RTS antara lain: war commander, empire earth, romance of the three kingdom dan sebagainya.

9) *Role Playing Game*

Pada game bertipe RPG pemain memerankan sebuah karakter dalam game. Berbeda dengan game bertipe action, pada game RPG hal yang diutamakan

adalah cerita dalam game. Selain itu didalam game bertipe RPG biasanya terdapat sub game dengan tipe lain (seperti bertipe fighting, action atau RTS). Contoh game bertipe RPG antara lain : BOBO games, Zelda, Megaman, Final Fantasy dan sebagainya.

10) MMO

MMO (Masive Multiplayer Online), merupakan game yang dapat dimainkan secara bersama-sama pada internet browser. Hal yang diutamakan dalam sebuah game bertipe MMO adalah kebersamaan dengan pemain lain. Contoh game bertipe MMO antara lain : ragnarok, pangya, gun bound, seal dan sebagainya. Game flash bertipe MMO masih sangat sedikit-hal tersebut dikarenakan untuk membuat game flash MMO diperlukan software lain (server side software) seperti PHP, CGI, Fire Fox, dan sebagainya. Contoh dari game flash bertipe MMO antara lain : Dofus dan battle-on.

2.4 *Scrabble*

Scrabble adalah permainan papan dan permainan menyusun kata yang dimainkan 2 atau 4 orang yang mengumpulkan poin berdasarkan nilai kata yang dibentuk dari keping huruf di atas papan permainan berkotak-kotak (15 kolom dan 15 baris). Biji permainan berupa keping berbentuk bujur sangkar yang bertuliskan huruf pada salah satu sisi. Pemain mengambil hingga sebanyak tujuh buah keping huruf dari kantong, dan berusaha menyusun kata secara mendatar atau menurun seperti teka-teki silang. Kata-kata yang dibuat harus merupakan kata yang diizinkan untuk dimainkan berdasarkan kamus standar sesuai dengan bahasa yang dimainkan. Pemain yang mengumpulkan total poin tertinggi dinyatakan sebagai pemenang. Scrabble dijual di 121 negara dalam 29 versi bahasa-bahasa di dunia. Seratus juta set permainan scrabble terjual di seluruh dunia, dan setiap tiga keluarga di Amerika memiliki satu set permainan scrabble [5].

2.4.1 Definisi *Scrabble*

Permainan *Scrabble* mulanya diciptakan pada tahun 1938 dengan nama "Criss-Crosswords" oleh seorang arsitek bernama Alfred Mosher Butts. Permainan ini merupakan penyempurnaan dari permainan Lexiko yang lebih dulu diciptakannya, tapi dilengkapi papan permainan dan cara bermain seperti teka-teki silang. Permainan tetap menggunakan keping huruf seperti Lexiko yang dibuat berdasarkan hasil perhitungan distribusi frekuensi penggunaan huruf berbagai tulisan berbahasa Inggris, termasuk artikel surat kabar The New York Times. Alfred Butts memproduksi sendiri permainan ini dan menawarkannya ke berbagai perusahaan mainan besar tapi tidak laku. Gambar 2.1 adalah sebagian contoh soal dari scrabble serta solusi penyelesaiannya.



Gambar 2.1 contoh soal dan solusi penyelesaiannya

2.4.2 Aturan permainan *scrabble*

Permainan dimainkan 2 hingga 4 orang dengan papan permainan berkotak-kotak yang terdiri dari 15 kolom dan 15 baris. Sewaktu menyusun kata, setiap kotak diisi dengan satu keping huruf. Di klub resmi atau turnamen, pertandingan hanya dimainkan dua orang pemain (atau, kadang-kadang antara 2 tim yang masing-masing anggotanya bekerja sama menyusun huruf dari rak yang

sama). Setiap huruf memiliki nilai tertentu (antara 1 sampai 10) yang bergantung pada frekuensi kemunculan huruf tersebut pada tulisan yang standar. Pada edisi Scrabble bahasa Inggris, huruf-huruf yang biasa muncul dalam kosakata bahasa Inggris, seperti huruf "E" atau "O" hanya bernilai 1 poin. Huruf-huruf yang jarang muncul, seperti "Q" dan "Z" masing-masing bernilai 10 poin. Selain itu, terdapat dua keping kosong (tidak ditulisi huruf) yang bernilai poin nol, tapi bisa dipakai untuk melambangkan semua huruf mulai dari "A" sampai "Z". Papan permainan ditandai dengan kotak-kotak "bonus" yang melipatgandakan jumlah poin. Kotak berwarna merah bertuliskan "triple-word" mengalikan 3 total poin yang didapat dari sebuah kata; kotak berwarna merah jambu "double-word" mengalikan 2 total poin yang didapat dari sebuah kata; kotak biru tua "triple-letter" mengalikan 3 nilai huruf yang diletakkan di atasnya, dan kotak biru muda "double-letter" mengalikan 2 nilai huruf yang diletakkan di atasnya. Kotak di bagian tengah papan permainan (H8) bernilai "double-word" dan diberi tanda bintang atau logo.

2.5 Kcerdasan Buatan

Kecerdasan buatan merupakan sub-bidang ilmu komputer (computer science) yang khusus ditujukan untuk membuat perangkat lunak dan perangkat keras yang sepenuhnya bisa menirukan beberapa fungsi otak manusia, atau cabang ilmu komputer yang mempelajari otomatisasi tingkah laku cerdas (intelligent). Kecerdasan buatan didasarkan pada teori suara (sound theoretical) dan prinsip-prinsip aplikasi dari bidangnya. Prinsip-prinsip ini meliputi struktur data yang digunakan dalam representasi pengetahuan algoritma yang diperlukan untuk mengaplikasikan pengetahuan serta bahasa dan teknik pemograman yang digunakan dalam mengimplementasikannya.

Beberapa para ahli, mendefinisikan kecerdasan buatan (artificial intelligence) secara berbeda-beda tergantung pada sudut pandangnya masing-masing. Para ahli ada yang mendefinisikan kecerdasan buatan (artificial intelligence) fokus kepada logika berpikir manusia saja, ada juga yang mendefinisikan kecerdasan buatan (artificial intelligence) secara lebih luas terhadap tingkah laku manusia. Stuart Russel dan Peter Norvig mengelompokan

definisi kecerdasan buatan (artificial intelligence) yang diperoleh dari beberapa textbook berbeda kedalam empat kategori, yaitu [6]:

1. Thinking humanly : The cognitive modeling approach

   Pendekatan thinking humanly dilakukan dengan dua cara sebagai berikut :

   a. Melalui intropeksi dimana proses ini adalah menangkap pemikiran pemikiran yang didapat pada saat berfikir.

   b. Melalui eksperimen-eksperimen psikologi.

2. Acting humanly : The turing test approach

   Pada tahun 1950, Alan Turing membuat sebuah eksperimen program penginterogasian untuk menguji kemampuan komputer pada manusia melalui teletype (komunikasi berbasis teks jarak jauh). Dalam hal ini, apabila integrator tidak dapat membedakan yang diinterogasi adalah manusia atau komputer maka program tersebut lolos dari turing test. Turing test sengaja menghindari interaksi fisik antara integrator dan komputer, karena simulasi fisik manusia tidak memerlukan intelejensia.

3. Thinking rationally : The laws of thought approach

   Terdapat dau masalah pada pendekatan ini, yaitu :

   a. Tidak mudahnya untuk membuat pengetahuan informal dan menyatakan pengetahuan tersebut kedalam formal term yang diperlukan oleh notasi logika, khususnya ketika pengetahuan tersebut memiliki kepastian kurang dari 100%.

   b. Terdapat perbedaan besar antara dapat memecahkan masalah “dalam prinsip” dan memecahkannya “dalam dunia nyata”.

4. Acting rationally : The rationally agent approach

   Membuat inferensi yang logis merupakan bagian dari suatu rational agent. Hal ini disebabkan satu-satunya cara untuk melakukan aksi secara rasional adalah dengan menalar secara logis. Dengan menalar secara logis, maka bisa didapatkan kesimpulan bahwa aksi yang diberikan akan mencapai tujuan atau tidak. Apabila mencapai tujuan, maka agent dapat melakukan aksi berdasarkan kesimpulan tersebut.

Thinking humanly dan acting humanly adalah dua definisi dalam arti yang sangat luas. Hingga saat ini, pemikiran manusia yang diluar rasio, yakni refleks dan intuitif (perasaan), belum dapat ditirukan oleh komputer. Sedangkan definisi thinking rationally jauh lebih sempit dari acting rationally dengan pendekatan rational agent. Hal ini berdasarkan pemikiran bahwa komputer dapat melakukan penlaran secara logis dan juga dapat melakukan aksi secara rasional berdasarkan hasil penalaran.

2.6 Algoritma

Algoritma adalah salah satu cabang ilmu komputer yang membahas prosedur penyelesaian suatu permasalahan. Algoritma adalah urutan langkah-langkah dalam menentukan suatu masalah. Algoritma juga dapat didefinisikan dengan deretan langkah komputasi yang mentransformasikan masukan (input) menjadi keluaran (output). Dapat disimpulkan algoritma adalah serangkaian urutan langkah-langkah atau prosedur untuk menyelesaikan suatu masalah dengan memproses nilai keluaran.

Algoritma dapat ditulis dalam bentuk pseudocode ataupun flowchart. Pseudocode berasal dari kata pseudo yang artinya mirip atau menyerupai dan code berarti kode, sehingga pseudocode adalah kode yang menyerupai kode program sebenarnya. Sedangkan flowchart merupakan bentuk gambar atau diagram yang mempunyai aliran satu atau dua arah secara skuensial [7].

2.6.1 Definisi Algoritma

Algoritma adalah urutan langkah-langkah logis penyelesaian yang disusun secara matematis dan logis. Langkah-langkah dalam algoritma harus logis dan harus dapat ditentukan bernilai salah atau benar. Dalam beberapa konteks, algoritma adalah spesifikasi urutan langkah untuk melakukan pekerjaan tertentu. Pertimbangan dalam pemilihan algoritma adalah [7].

1. Algoritma haruslah benar, artinya algoritma akan memberikan keluaran yang dikehendaki dari sejumlah masukan yang diberikan. Tidak peduli

sebagus apapun algoritma, apabila menghasilkan keluaran yang bernilai salah maka algoritma tersebut bukanlah algoritma yang baik.

2. Seberapa baik algoritma yang diterapkan, artinya algoritma yang baik harus mampu memberikan hasil yang sedekat mungkin dengan nilai sebenarnya. Dalam hal ini, hal terpenting pada algoritma untuk menyelesaikan masalah yang memerlukan aproksimasi hasil (hasil yang berupa pendekatan).
3. Efisiensi algoritma, artinya efisiensi algoritma dapat ditinjau dari dua hal yaitu efisiensi waktu dan memori.

2.6.2 Sejarah Algoritma

Kata algoritma berasal dari latinisasi nama seorang ahli matematika dari Uzbekistan Al Khawārizmi. Dahulu, orang hanya mengenal kata algorism yang berarti proses menghitung dengan angka arab. Para ahli bahasa berusaha menemukan asal kata algorism, dan akhirnya para ahli sejarah matematika berhasil menemukan asal kata yang berasal dari penulis buku arab yang terkenal yaitu Abu Ja'far Muhammad Ibnu Musa Al Khawārizmi. Al Khawārizmi dibaca orang barat menjadi algorism. Al Khawārizmi menulis kitab Al Jabar Wal-Muqabala yang artinya "Buku Pemograman Dan Pengurangan" (The Book of Restoration And Reducation). Perubahan kata dari algorism menjadi kata algoritm muncul dikarenakan kata algorism sering dikelirukan dengan arithmetic, sehingga akhiran -sm berubah menjadi –thm. Seiring perkembangan, lambat laun kata algorithm berangsur-angsur dipakai sebagai metode perhitungan (komputasi) secara umum sehingga kehilangan makna kata aslinya. Dalam bahasa Indonesia, kata algorithm diserap menjadi algoritma [7].

2.7 Algoritma Genetika

Algoritma genetika adalah teknik pencarian yang digunakan dalam penghitungan untuk mencari solusi perkiraan maupun benar untuk optimasi dan masalah pencarian. Algoritma ini dikategorikan sebagai algoritma global search

heuristics. Algoritma ini terinspirasi dari prinsip dari genetika dan seleksi alam (teori evolusi Darwin). Algoritma ini ditemukan di Universitas Michigan, Amerika Serikat, melalui sebuah penelitian dan dipopulerkan oleh salah satu muridnya, David Goldberg [8].

Algoritma genetika diimplementasikan sebagai simulasi komputer di mana sebuah populasi dari representasi abstrak (disebut kromosom, genotipe, atau genom) kandidat solusi (disebut individu, makhluk, atau fenotipe). Biasanya, solusi direpresentasikan dalam biner sebagai string yang terdiri dari 0 dan 1, namun penggunaan metode encoding lain juga mungkin dipakai. Berbeda dengan teknik pencarian konvensional, algoritma genetika bermula dari himpunan solusi yang dihasilkan secara acak.

Kromosom-kromosom berevolusi dalam suatu proses iterasi yang berkelanjutan yang disebut generasi. Pada setiap generasi, kromosom dievaluasi berdasarkan suatu fungsi evaluasi. Setelah beberapa generasi maka algoritma genetika akan konvergen pada kromosom terbaik, yang diharapkan merupakan solusi optimal. Pertama kali, sebelum algoritma genetika dijalankan, maka perlu didefinisikan fungsi fitness sebagai masalah yang ingin dioptimalkan. Jika nilai fitness semakin besar, maka sistem yang dihasilkan semakin baik. Fungsi fitness ditentukan dengan metode heuristik.

Algoritma genetika sangat tepat digunakan untuk penyelesaian masalah optimasi yang kompleks dan sukar diselesaikan dengan menggunakan metode konvensional. Sebagaimana halnya proses evolusi di alam, suatu algoritma genetika yang sederhana umumnya terdiri dari tiga operasi yaitu: operasi reproduksi, operasi persilangan (crossover), dan operasi mutasi.

Struktur umum dari suatu algoritma genetika dapat didefinisikan dengan langkah-langkah sebagai berikut:

1. Membangkitkan populasi awal secara random.
2. Membentuk generasi baru dengan menggunakan tiga operasi diatas secara berulang-ulang sehingga diperoleh kromosom yang cukup untuk membentuk generasi baru sebagai representasi dari solusi baru.

3. Evolusi solusi yang akan mengevaluasi setiap populasi dengan menghitung nilai fitness setiap kromosom hingga kriteria berhenti terpenuhi.

Bila kriteria berhenti belum terpenuhi maka akan dibentuk lagi generasi baru dengan mengulangi langkah 2.

beberapa kriteria berhenti yang sering digunakan

2.7.1 Prinsip Algoritma Genetika

Algoritma genetika terinspirasi dari prinsip genetika dan seleksi alam (teori evolusi Darwin) yang ditemukan di Universitas Michigan, AS, oleh John Holland (1975) melalui sebuah penelitian dan dipopulerkan oleh salah satu muridnya, David Goldberg.

Konsep dasar algoritma genetika sebenarnya dirancang utk menyimulasikan proses-proses dlm sistem alam yang diperlukan utk evolusi, khususnya teori evolusi alam yg dicetuskan oleh Charles Darwin, yaitu survival of the fittest [9].

Permasalahan yang membutuhkan algoritma genetika :

1. Ruang pencarian sangat besar, kompleks atau kurang dipahami.
2. Tidak ada pengetahuan yang memadai untuk menyederhanakan ruang pencarian yang sangat besar menjadi ruang pencarian yang lebih sempit.
3. Tidak ada analisis matematis yang bisa menangani ketika metode konvensional gagal menyelesaikan masalah yang dihadapi.
4. Solusi yang dihasilkan tidak harus optimal, asal sudah memenuhi kriteria sudah bisa diterima.
5. Mempunyai kemungkinan solusi yang jumlahnya tak hingga.
6. Membutuhkan solusi “real-time”, yaitu solusi yang bisa didapatkan dengan cepat sehingga dapat diimplementasikan untuk permasalahan yang mempunyai perubahan yang cepat.

2.7.2 Struktur Algoritma Genetika

Algoritma genetika melakukan teknik pencarian secara sekaligus atas sejumlah solusi yang dikenal dengan istilah populasi sedangkan individu yang

terdapat pada sebuah populasi disebut dengan kromosom. Penentuan populasi awal didapat secara acak yang kemudian populasi berikutnya akan didapat dari hasil proses genetika.

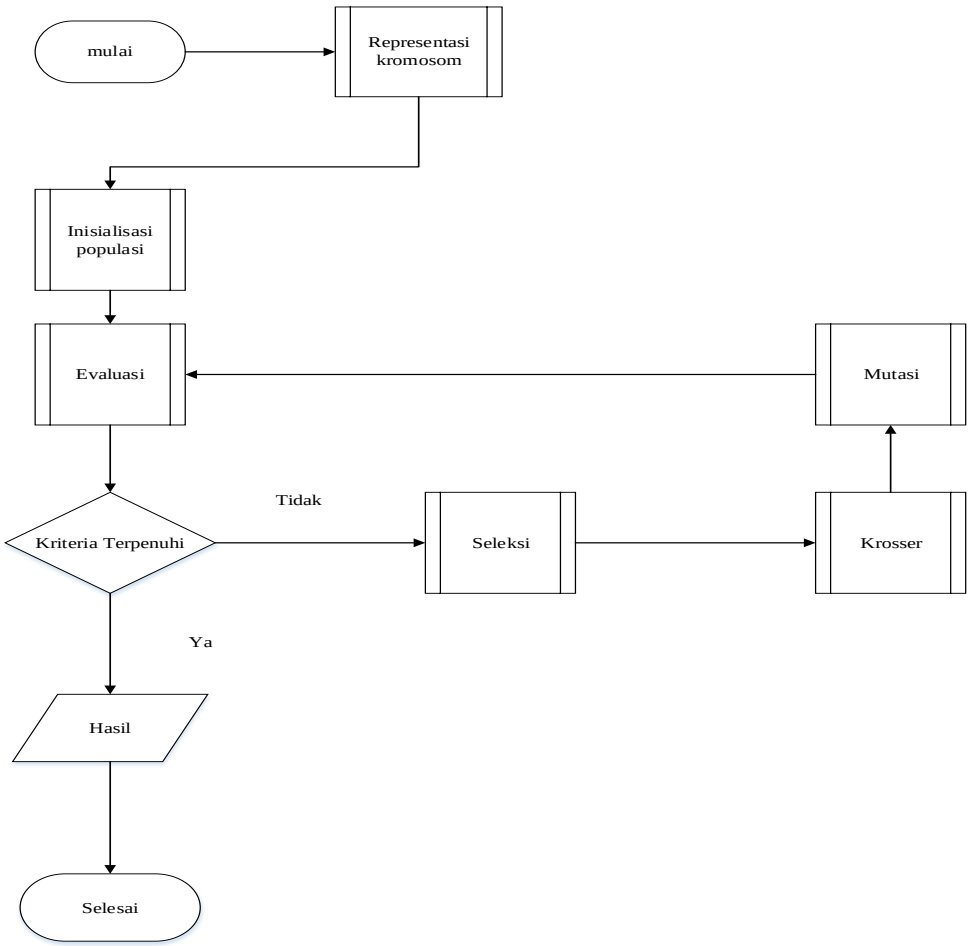
Ada beberapa hal yang harus dilakukan dalam algoritma genetika adalah :

1. Mendefinisikan individu, merupakan pernyataan solusi awal dari kasus yang diambil
2. Mendefinisikan nilai fitness, merupakan suatu ukuran seberapa baik individu dalam mencapai solusi.
3. Menentukan proses pembangkitan populasi awal yang dilakukan secara acak
4. Menentukan proses seleksi, crossover atau perkawinan silang.

Beberapa definisi penting yang digunakan dalam Algoritma Genetika, antara lain :

a. Genotype (Gen), sebuah nilai yang menyatakan satuan dasar yang membentuk arti tertentu dalam satu kesatuan yang disebut kromoson. Gen ini dapat berupa nilai biner, float, integer maupun karakter.
b. Kromoson, gabungan gen yang membentuk nilai tertentu.
c. Individu, menyatakan suatu nilai atau keadaan yang menyatakan salah satu solusi yang mungkin dari permasalahan yang diangkat.
d. Populasi, sekumpulan individu yang akan diproses bersama dalam satu siklus proses evolusi.
e. Nilai Fitness, menyatakan seberapa baik nilai dari suatu individu atau solusi yang didapatkan.

Algoritma genetika mempunyai beberapa komponen di dalam prosesnya yaitu representasi kromosom, prosedur inisialisasi, fungsi evaluasi, seleksi, operator genetika dan penentuan parameter. Berikut ini pada Gambar 2.2 adalah flowchart alur proses dari algoritma genetika :



Gambar 2.2 flowchart algoritma genetika

2.7.3 Kelebihan algoritma genetika

Beberapa hal yang termasuk kelebihan dari algoritma genetika adalah sebagai berikut:

1. Mengoptimalkan dengan variabel kontinu atau diskrit.
2. Tidak memerlukan informasi derivatif.
3. Bersamaan pencarian dari sebuah sampling yang luas pada permukaan biaya.
4. Berkaitan dengan sejumlah besar *variable*.
5. Baik untuk komputer paralel.
6. Mengoptimalkan permukaan variable dengan biaya yang sangat kompleks (GA bisa melompat dari minimum lokal).

7. Memberikan daftar *variable* yang optimal, bukan hanya solusi tunggal.
8. Dapat menyandikan *variable* sehingga optimasi dilakukan dengan mengkodekan *variable.*
9. Bekerja dengan data numerik yang dihasilkan, data eksperimen, atau analitis fungsi.

Algoritma genetika berangkat dari himpunan solusi yang dihasilkan secara acak yang disebut populasi. Sedangkan setiap individu dalam populasi disebut kromosom yang merupakan representasi dari solusi dan masing-masing dievaluasi tingkat ketanggguhannya (fitness) oleh fungsi yang telah ditentukan. Melalui proses seleksi alam atas operator genetik, gen-gen dari dua kromosom (disebut parent) diharapkan akan menghasi lkan kromosom baru dengan tingkat fitness yang lebih tinggi sebagai generasi baru atau keturunan (offspring) berikutnya. Kromosom-kromosom tersebut akan mengalami iterasi yang disebut generasi (generation). Pada setiap generasi, kromosom dievaluasi berdasarkan nilai fungsi fitness. Setelah beberapa generasi maka algoritma genetika akan konvergen dapat kromosom terbaik, yang merupakan solusi optima

2.7.4 Langkah-langkah algoritma genetika

Struktur umum dari suatu algoritma genetika dapat didefinisikan dengan langkah-langkah sebagai berikut:

1. Pengkodean

   Pengkodean disini meliputi pengkodean gen dan kromosom.
2. Inisisalisasi populasi awal

   Membangkitkan sejumlah kromosom (sesuai dengan ukuran populasi) untuk dijadikan anggota populasi awal. Populasi itu sendiri terdiri dari sejumlah kromosom yang merepresentasikan solusi yang diinginkan.
3. Evaluasi nilai fitness

   Setiap kromosom pada populasi dihitung nilai fitness-nya berdasarkan fungsi fitness. Nilai fitness suatu kromosom menggambarkan kualitas kromosom dalam populasi tersebut.

Proses ini akan mengevaluasi setiap populasi dengan menghitung nilai fitness setiap kromosom dan mengevaluasinya sampai terpenuhi kriteria berhenti.

4. Pembentukan kromosom baru

a. Seleksi

Memilih sejumlah kromosom yang akan menjadi kromosom calon parent.

b. *Crossover*

Mengkombinasikan dua kromosom parent (induk) berdasar nilai probabilitas crossover-nya untuk menghasilkan offspring.

c. Mutasi

Mengubah sejumlah gen berdasar nilai probabilitas mutasinya untuk menghasilkan kromosom baru.

d. Update Generasi

Membaharui kromosom yang terdapat dalam populasi.

e. Pengecekan faktor pemberhenti

Jika memenuhi dari salah satu dari kondisi untuk berhenti, maka siklus algoritma genetika berhenti. Proses evolusi bisa dihentikan berdasarkan beberapa kondisi, misalnya ketika evolusi telah mencapai 19 jumlah generasi maksimum yang diizinkan, terdapat suatu individu yang telah memiliki fitness tertentu yang diharapkan, keberagaman populasi telah mencapai tingkat minimum yang diizinkan, dalam beberapa generasi tertentu, tidak ada peningkatan nilai fitness yang diharapkan.

Sebelum algoritma genetika dilakukan, ada dua hal penting yang harus dilakukan yaitu pendefinisian kromosom yang merupakan suatu solusi yang masih berbentuk simbol dan fungsi fitness atau fungsi obyektif. Dua hal ini berperan penting dalam algoritma genetika untuk menyelesaikan suatu masalah [10].

2.8 Unified Modeling Language (UML)

Unified Modeling Language merupakan salah satu alat bantu yang dapat digunakan dalam bahasa pemograman yang berorientasi objek, saat ini UML akan mulai menjadi standar masa depan bagi industri pengembangan sistem/perangkat lunak yang berorientasi objek sebab pada dasarnya UML digunakan oleh banyak perusahaan raksasa seperti IBM, Microsoft, dan sebagainya. Berikut ini beberapa definisi UML [11]:

Unified Modeling Language merupakan metode pengembangan perangkat lunak (sistem informasi) dengan menggunakan metode grafis serta merupakan bahasa untuk visualisasi, spesifikasi, konstruksi serta dokumentasi.

10. Unified Modeling Language (UML) adalah bahasa yang telah menjadi standar untuk visualisasi, menetapkan, membangun dan mendokumentasikan arti suatu sistem perangkat lunak.
11. Unified Modeling Language (UML) dapat didefinisikan sebagai sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem perangkat lunak.
12. Unified Modeling Language (UML) merupakan standard modeling language yang terdiri dari kumpulan-kumpulan diagram, dikembangkan untuk membantu para pengembang system dan software agar bisa menyelesaikan tugas-tugas seperti spesifikasi, visualisasi, desain arsitektur, konstruksi, simulasi dan testing, serta dokumentasi.

Berdasarkan beberapa pendapat yang dikemukakan diatas dapat ditarik kesimpulan bahwa Unified Modeling Language (UML) adalah sebuah bahasa yang berdasarkan grafik atau gambar untuk memvisualisasikan, menspesifikasikan, membangun, dan pendokumentasian dari sebuah sistem pengembangan perangkat lunak berbasis Object Oriented Programming (OOP).



Gambar 2.3 Logo Unified Modeling Language (UML)

2.8.1 Diagram Unified Modeling Language (UML)

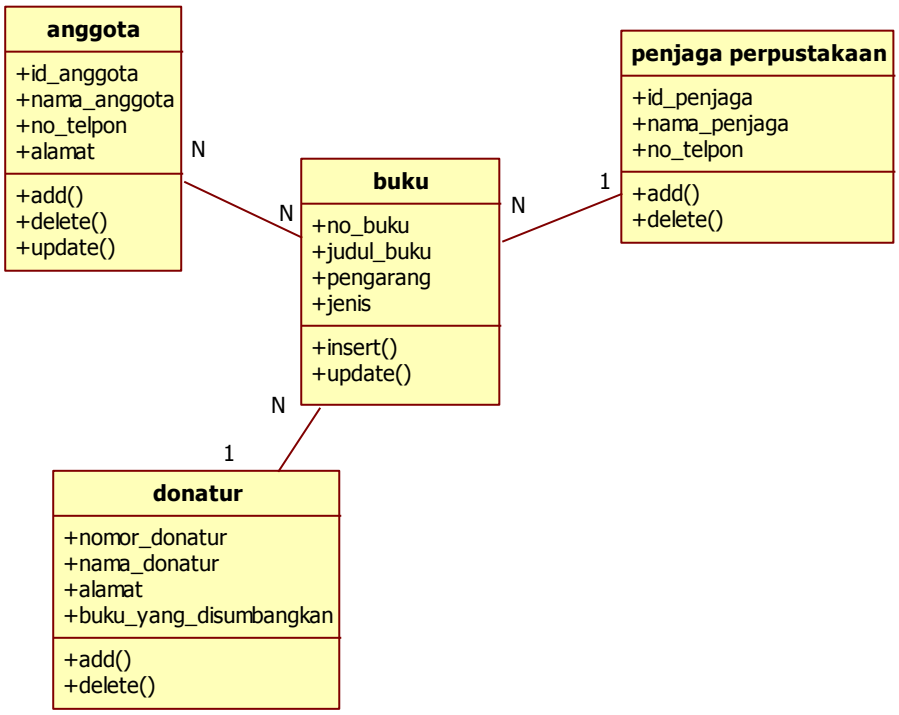
Beberapa literatur menyebutkan bahwa UML menyediakan sembilan jenis diagram, yang lain menyebutkan delapan karena ada beberapa diagram yang digabung, misalnya diagram komunikasi, diagram urutan dan diagram pewaktuan digabung menjadi diagram interaksi. Namun demikian model-model itu dapat dikelompokkan berdasarkan sifatnya yaitu statis atau dinamis. Jenis diagram itu antara lain :

13. Diagram Kelas (Class Diagram)

Diagram kelas menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

a. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.
b. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Diagram kelas dibuat agar pembuat program atau programmer membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak menjadi sinkron. Berikut adalah contoh dari diagram kelas.



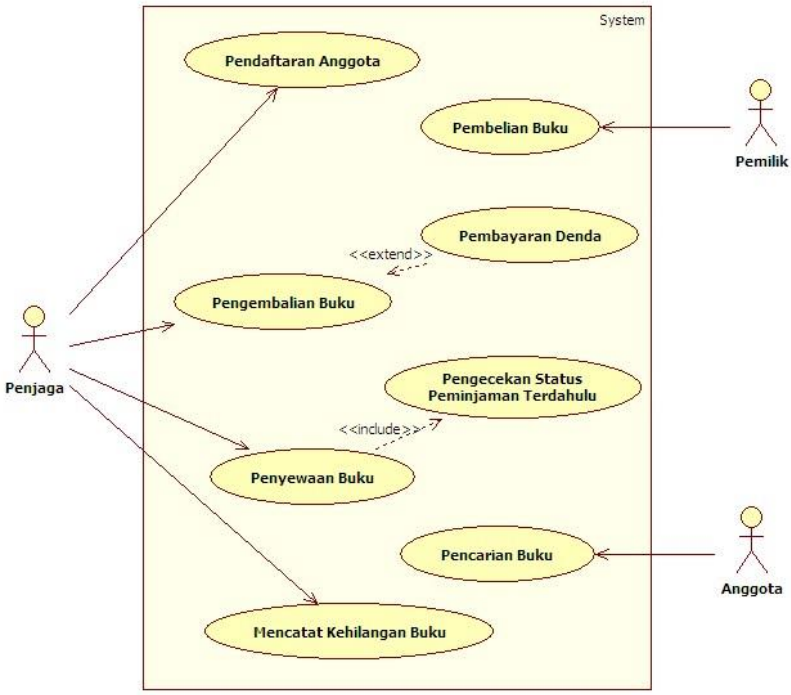
Gambar 2.4 Contoh class diagram

14. Diagram Use Case

Diagram use case merupakan pemodelan untuk kelakuan (behaviour) sistem informasi yang akan dibuat. Use case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, use case digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Syarat penamaan pada use case adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada use case yaitu pendefinisian apa yang disebut aktor dan use case.

a. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.

b. Use case merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.



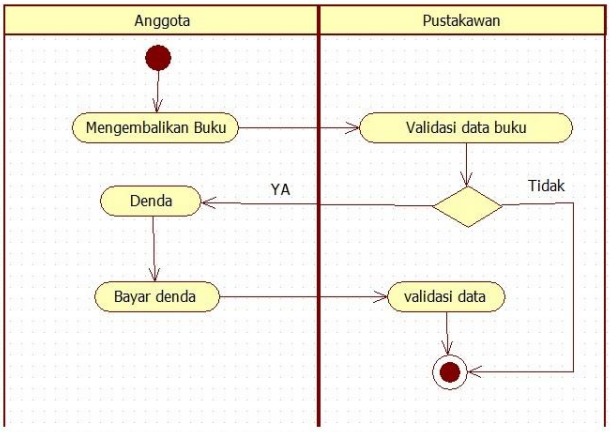
Gambar 2.5 Contoh use case diagram

15. Diagram Aktivitas (Activity Diagram)

Diagram aktivitas menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut :

a. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
b. Urutan atau pengelompokkan tampilan dari system / user interface dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.

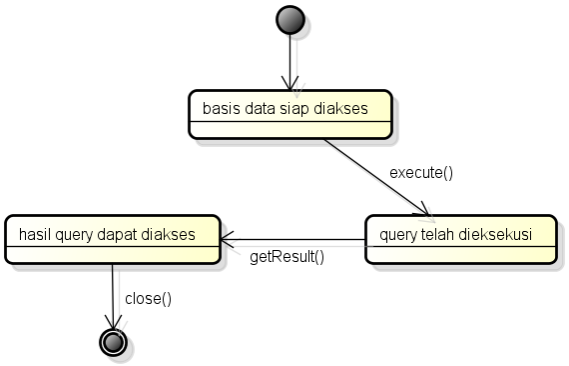
c. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
d. Rancangan menu yang ditampilkan pada perangkat lunak.



Gambar 2.6 Contoh activity diagram

16. Diagram State Machine (Statechart Diagram)

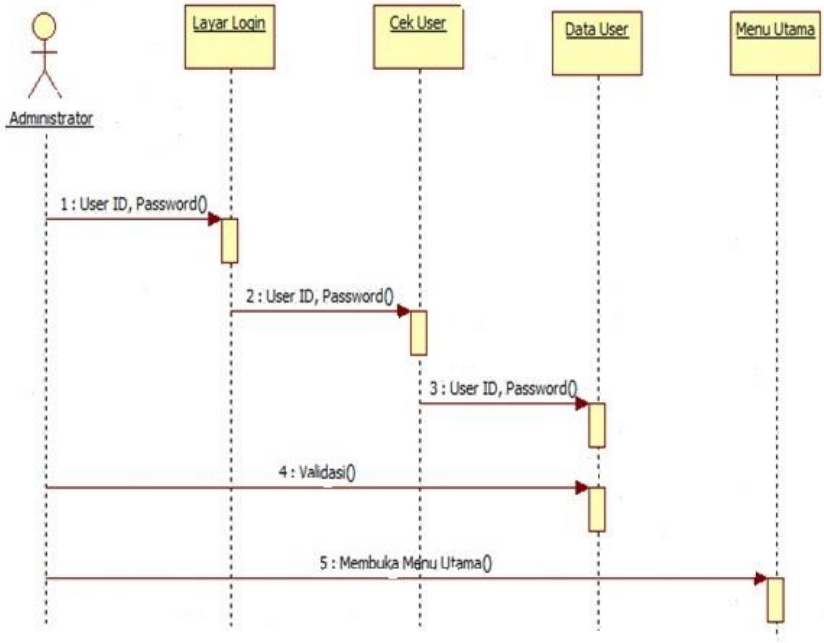
State machine diagram atau statechart diagram atau dalam bahasa Indonesia disebut diagram mesin status atau sering juga disebut diagram status digunakan untuk menggambarkan perubahan status atau transisi status dari sebuah mesin atau sistem atau objek. Jika diagram sekuen digunakan untuk interaksi antar objek maka diagram status digunakan untuk interaksi di dalam sebuah objek. Perubahan tersebut digambarkan dalam suatu graf berarah.



Gambar 2.7 Contoh state chart diagram

17. Diagram Sekuen (Squence Diagram)

Diagram sekuen menggambarkan kelakuan pada objek use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada use case. Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian use case yang memiliki proses sendiri atau yang penting semua use case yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak use case yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak.



Gambar 2.8 Contoh squence diagram

2.9 Java

Java adalah salah satu bahasa pemrograman komputer yang berorientasi objek, yang diciptakan oleh satu tim dari perusahaan Sun Microsystem, perusahaan workstation UNIX (Sparc) yang cukup terkenal. Java diciptakan

berdasarkan bahasa C++, dengan tujuan platform independent (dapat dijalankan pada berbagai jenis hardware tanpa kompilasi ulang), dengan slogan “Write Once, Run Anywhere (WORA), dan Java pada hakikatnya lebih sederhana dan memakai objek secara murni dibanding dengan bahasa pemrograman C++ [12].

Asal usul java dimulai pada tahun 1991 ketika Sun Microsystem melakukan penelitian terhadap berbagai produk elektronika. James Gosling (kepala proyek penelitian) beserta Patrick Naughton ditugaskan untuk merancang perangkat lunak aplikasi yang independen, tidak bergantung pada jenis perangkat keras, agar bisa dipakai pada berbagai peralatan elektronik itu. James Gosling kemudian memakai bahasa pemrograman C++ untuk menulis beberapa aplikasi untuk peralatan mikro, namun ternyata dia mengalami banyak kesulitan. Setiap kali sebuah peralatan mikro menggunakan jenis microchip yang berbeda, program yang digunakannya harus dikompilasi ulang. Berdasarkan hal itu kemudian munculah ide dari James Gosling untuk menciptakan sebuah bahasa baru, yang ditulis berdasarkan C++ namun dengan beberapa perbaikan. Bahasa ini kemudian disebut Oak (nama pohon yang terlihat dari jendela ruangan kerja James Gosling), yang digunakan untuk membuat perangkat lunak cerdas bagi peralatan elektronika buatan Sun Mirosystem.

Pada tahun 1994, ketika Web mulai populer nama bahasa Oak kemudian diganti menjadi Java (salah satu sebabnya ternyata nama Oak telah terlebih dahulu dipakai untuk hal lain). Menurut gosip, Gosling bingung mencari nama untuk bahasa baru tersebut karena setiap nama yang dipikirkannya untuk nama bahasa pemrogramannya ternyata sudah ada yang memakai, sampai suatu ketika dia mampir di sebuah kafe untuk minum kopi, dan ia meminta “Java”. Timbullah idenya untuk memakai nama Java. Menurut Gosling nama ini cukup cool.

Java kemudian dipakai untuk membuat sebuah browser yang bernama WebRunner. Setelah berhasil maka browser ini kemudian dinamakan HotJava dengan simbol secangkir kopi panas.



Gambar 2.9 Logo Java

2.10 Eclipse

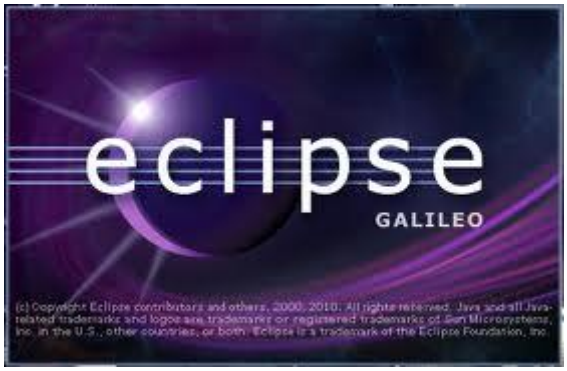
Eclipse adalah sebuah IDE (Integrated Development Environment) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua platform (platform-independent) [13].

18. Multi-platform: Target sistem operasi Eclipse adalah Microsoft Windows,Linux, Solaris, AIX, HP-UX dan Juga Mac OS X.
19. Mulit-language: Eclipse dikembangkan dengan bahasa pemrograman Java, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis bahasa pemrograman lainnya, seperti C/C++, Cobol, Python, Perl, PHP, dan lain sebagainya.
20. Multi-role: Selain sebagai IDE untuk pengembangan aplikasi, Eclipse pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, test perangkat lunak, pengembangan web, dan lain sebagainya.

Eclipse awalnya dikembangkan oleh IBM untuk menggantikan perangkat lunak IBM Visual Age for Java 4.0. Produk ini diluncurkan oleh IBM pada tanggal 5 November 2001, yang menginvestasikan sebanyak US$ 40 juta untuk pengembangannya. Semenjak itu konsursium Eclipse Foundation mengambil alih untuk pengembangan Eclipse lebih lanjut dan pengaturan organisasinya.

Pada saat ini merupakan salah satu IDE favorit dikarenakan gratis dan open source, yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari Eclipse yang membuatnya populer adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan

komponen yang dinamakan plug-in.Sampai saat sekarang ini Eclipse sudah mencapai versi 3.6 yang diberinama Helios.



Gambar 2.10 Logo eclips