BAB 2
LANDASAN TEORI

2.1. Budidaya Tanaman Holtikultura

Kata hoktikultura (*horticulture*) berasal dari bahasa latin, yakni hortus yang berarti kebun dan colere yang berarti menumbuhkan (terutama sekali mikroorganisme) pada suatu medium buatan. Secara harfiah, hortikultura berarti ilmu yang mempelajari pembudidayaan tanaman kebun. Akan tetapi, pada umumnya para pakar mendefinisikan hortikultura sebagai ilmu yang memperlajari budidaya tanaman sayura, buah-buahan, bunga-bungaan, atau tanaman hias [12].

Pada umumnya, isi kebun di Indonesia adalah berupa tanaman buah-buahan, tanaman sayuran, tanaman hias dan wangi-wangian, tanaman bumbu masak, tanaman obat-obatan, dan tanaman penghasil rempah-rempah. Sementara itu, di Negara-negara maju budidaya tanaman hortikultura sudah merupakan susatu usaha tani berpola komersial, yakni diusahakan secara monokultur di ladang produksi yang luas, misalanya perkebunan apel, anggur, tomat, dan pear(Pyrus Communis) di Amerika; perkebunan mangga dan kelengkeng di Queensland, Australia; serta perkebunan tomat hidroponik di New Zeland.

Seiring dengan semakin pentingnya kedudukan hortikultura dalam kehidupan sehari-hari sebagai sumber berbagai vitamin dan mineral di samping sebagai bahan baku berbagai produk olahan, pengusahaan hortikultura, khususnya buah-buahan, di Indonesia kini mulai dikelola dengan pola agribisnis.

2.1.1. Jenis – Jenis Tanaman Holtikultura

Hortikultura dapat dikelompokkan atas 3 kategori yaitu [2]

1. Tanaman Buah-buahan, kelompok tanaman ini memiliki keanekaragaman morfologi, seperti ada yang berbentuk pohon (misalnya rambutan, mangga, durian, jeruk, dan sebagainya), bentuk semak (markisa).
2. Tanaman sayuran, tanaman ini merupakan tanaman hortikultura yang utama. Beberapa jenis sayuran ada yang berasal dari buah (tomat), daun (bayam), akar (wortel), biji (buncis), bunga (kembang kol) dan sebagainya. Berbeda dengan tanaman buah-buahan, sayuran memiliki umur yang relatif singkat. Tanaman ini umumnya dikonsumsi dalam bentuk segar, oleh karenanya proses penanganannya lebih spesifik dibandingkan dengan hortikultura lainnya.
3. Tanaman Hias, manfaat dari tanaman hias ini adalah meningkatkan aestetika lingkungan. Budidaya tanaman ini dapat dilakukan pada ruang terbuka maupun didalam ruangan.

2.1.2. Ekologi Tanaman Hortikultura

Budidaya tanaman hortikultura menghendaki perhatian yang serius, khusunya dalam penentuan persyaratan ekologinya. Hal ini dikarenakan pertumbuhan dan perkembangan tanaman sangat tergantung ekologi atau lingkungan di mana tanaman tersebut tumbuh. Apabila tanaman diusahakan pada lingkungan yanga memenuhi kebutuhan syarat tumbuhnya, dapat dipastikan tanaman tersebut akan tumbuh dan berproduksi secara maksimal. Hal sebaliknya akan terjadi apabila tanaman tersebut tumbuh ada lingkungan yang tidak atau kurang mendukung.

Faktor-faktor lingkungan yang mempengaruhi pertumbuhan dan produksi tanaman hortikultura dapat dikelompokan atas faktor iklim (cahaya, suhu, dan keadaan udara) dan faktor medium tumbuh (tanah dan air).

2.1.3. Panen

Pemanenan merupakan kegiatan yang sangat menetukan dalam kegiatan operasional hortikultura, dan seringkali merupakan bagian termahal dari kegiatan produksi. Secara sederhana, istilah pemanenan diartikan sebagai

upaya memisahkan bagian tanaman yang memiliki nilai ekonomi dati tanaman induknya. Pada tanaman buah-buahan, bagian yang dipisahkan tersebut sudah tentu adalah buah, sedangkan pada tanaman sayuran, bagian-bagian tersebut dapat berupa daun, batang, akar, maupun buah. Sementara itu pada tanaman hias, bagian tanaman yang dipanen dapat berupa bunga berikut tangkainya (tanaman hias berbunga) atau daun berikut tangkainya (tanaman hias tidak berbunga) [12].

2.1.4. Perhitungan Waktu Panen Hortikultura

Waktu panen sangat ditentukan jenis (varietas) tanaman, hari tanam atau hari berbunga, dan kondisi lingkungan selama musim tanam. Selain itu, beberapa kriteria fisikologis juga telah digunakan untuk menentukan saat panen, seperti kekerasan, warna, dan kandungan karbohidrat. Pada sejumlah tanaman adakalanya digunakan kriteria lain, misalnya terbentuknya daerah absisi pada tangkai buah (misalnya pada labu dan blewah), tingkat perkembangan kuncup bunga (misalnya pada tanaman hiasa seperti mawar), warna buah (misalnya pada tomat, cabai, rambutan, dan nanas), rasio gula : asam (misalnya pada jeruk), bunyi kalau dipukul atau ditepuk (misalnya pada semangka dan nangka). Sedangkan pada tanaman durian, jatuhnya buah merupakan petunjuk yang paling tepat untuk pemanenan.

Penentuan tingkat kematangan buah yang tepat akan sangat nyata mengurangi kemerosotan kualitas buah yang dipanen. Apabila buah dipanen sebelum memasuki fase matang fisiologis, maka kualitasnya akan cepat sekali turun di dalam penyimpanan dan pengankutan karena tingginya laju tranpirasi yang mengakibatkan buah menjadi keriput akibat penuruan turgiditas.

Menurut [12] Penentuan waktu panen dilakukan dengan beberapa cara yaitu cara visual atau penampakan. Cara fisik: dengan perabaan. Cara komputasi, yaitu menghitung umur tanaman sejak tanam atau umur buah dari mulai bunga mekar. Cara kimia, yaitu dengan melakukan pengukuran atau analisis kandungan zat atau senyawa yang ada dalam komoditas.

Tabel 2.1 Tingkat Kematangan Buah dah Sayuran [1][2][3]

Index Kematangan	Tanaman
Hari setelah berbunga	Apple, Pir
Rata-rata panas selama pertumbuhan	Apple, jagung manis
Ukuran	Kebanyakan buah dan sayuran
warna	Kebanyakan buah dan sayuran
Bentuk	Pisang, brocoli

Sedangkan kematangan pada hortikultura [1] adalah ketika tanaman atau bagian tanaman memiliki prasyarat pemanfaatan oleh konsumen untuk tujuan tertentu.

1. Tahap buah dan sayuran di mana konsumen menginginkan buah dan sayuran atau buah-buahan dan sayuran yang dibutuhkan oleh pasar. tingkat kematangan hortikultura tergantung pada tujuan yang dipanen. Contoh: kacang polong Perancis okra polong matang bila lembut dengan ukuran maksimum, sesuai dengan kematangan hortikultura.
2. Tahap muncul setiap saat antara pembangunan dan pertumbuhan setiap tanaman pada buah.
3. Hal ini dapat diprediksi dengan menggunakan terminologi yang berbeda seperti belum matang, matang dan terlalu matang.
4. Tidak ada keharusan penuaan.

2.2. Kecerdasan Buatan

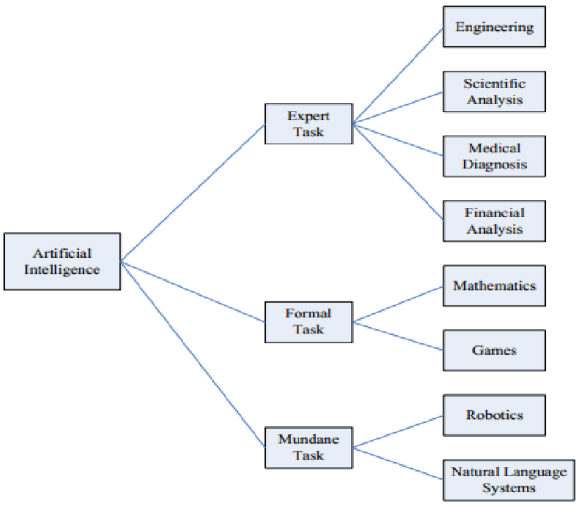
Kecerdasan buatan atau Artificial Intelligence (AI) merupakan cabang dari ilmu komputer yang berhubungan dengan pengautomatisan tingkah laku cerdas. Kecerdasan buatan didasarkan pada teori suara (sound theoretical) dan prinsip-prinsip aplikasi dari bidangnya [5]. Prinsip-prinsip ini meliputi struktur data yang digunakan dalam representasi pengetahuan, algoritma yang diperlukan untuk mengaplikasikan pengetahuan tersebut serta bahasa

dan teknik pemrograman yang digunakan dalam mengimplementasikannya. Berdasarkan sudut pandang, AI dapat dipandang sebagai berikut:

1. Sudut pandang kecerdasan, AI adalah bagaimana membuat mesin yang cerdas dan dapat melakukan hal -hal yang sebelumnya hanya dapat dilakukan manusia.
2. Sudut pandang bisnis, AI adalah sekelompok alat bantu yang berdaya guna dan metodologi yang menggunakan alat -alat bantu tersebut untuk menyelesaikan masalah-masalah bisnis.
3. Sudut pandang pemrograman, AI meliputi studi tentang pemrograman simbolik, pemecahan masalah dan proses pencarian.
4. Sudut pandang penelitian:

   a. Riset tentang AI dimulai pada awal tahun 1960-an, percobaan Pertama adalah membuat program permainan catur, membuktikan teori dan general problem solving.

   b. AI adalah nama pada akar dari studi area. Kecerdasan buatan memiliki sejumlah sub disiplin ilmu yang sering digunakan untuk pendekatan yang esensial bagi penyelesaian suatu masalah dan dengan aplikasi bidang AI yang berbeda. Berikut gambar bidang-bidang tugas dari AI yaitu [5]:



Gambar 2.1 Pembagian tugas AI

Aplikasi penggunaan AI dapat dibagi ke dalam tiga kelompok yaitu:

1. Expert Task

AI dibentuk berdasarkan pengalaman dan pengetahuan yang dimiliki oleh para ahli. Penggunaan ini dapat membantu para ahli untuk menyampaikan ilmu -ilmu yang dimiliki. Contohnya adalah:

a. Analisis finansial.
b. Analisis medikal.
c. Analisis ilmu pengetahuan.
d. Rekayasa (desain, pencarian, kegagalan, perencanaan, manufaktur).

2. Formal Task

AI digunakan untuk melakukan tugas-tugas formal yang selama ini manusia biasa lakukan dengan lebih baik. Contohnya adalah:

a. Game.
b. Matematika (geometri, logika, kalkulus, integral).

3. Mundane Task

Secara harfiah mundane adalah keduniaan. AI digunakan untuk melalukan hal-hal yang sifatnya duniawi atau melakukan kegiatan yang dapat membantu manusia. Contohnya adalah:

a. Persepsi.
b. Bahasa alami.
c. Robot control.

Aplikasi kecerdasan buatan memiliki dua bagian utama yaitu:

1. Basis Pengetahuan (Knowledge Base): berisi fakta-fakta, teori, pemikiran dan hubungan antara satu dengan lainnya.
2. Motor Inferensi (Inference Engine): kemampuan menarik kesimpulan berdasarkan pengalaman.

2.2.1. Kecerdasan Buatan yang digunakan

Dalam penelitian ini akan menggunakan dua bidang kedersaan buatan yaitu Algoritma Genetika dan Logika *Fuzzy*.

2.2.1.1. Algoritma Genetika

Algoritma genetika adalah algoritma pencarian heuristik yang didasarkan atas mekanisme evolusi biologis. Keberagaman pada evolusi biologis adalah variasi dari kromosom antar individu organisme. Variasi kromosom ini akan mempengaruhi laju reproduksi dan tingkat kemampuan organisme untuk tetap hidup. Pada dasarnya ada empat kondisi yang sangat mempengaruhi proses evaluasi yaitu:

1. Kemampuan organisme untuk melakukan reproduksi
2. Kebeeradaan populasi organisme yang bisa melakukan reproduksi
3. Keberagaman organisasi dalam suatu populasi
4. Perbedaan kemampuan untuk survive

Individu yang lebih kuat atau (fit) akan memiliki tinggkat survival dan tingkat reproduksi yang lebih tinggi jika dibandingkan dengan individu yang kurang fit. Pada kurun waktu tertentu (sering dikenal dengan istilah generasi), poulasi secara keseluruhan akan lebih banyak memuat organisme yang fit.

Algoritma genetika pertama kali dikembangkan oleh John Holland pada tahun 1975. John Holland mengatakan [5] bahwa setiap masalah yang berbentuk adaptasi (alami maupun buatan) dapat diformulasikan dalam terminologi genetika. Algoritma genetika adalah simulasi dari proses evolusi Darwin dan operasi genetika atas kromosom.

2.2.1.1.1. Stuktur Algoritma Genetika

Pada algoritma ini, teknik pencarian dilakukan sekaligus atas sejumlah solusi yang mungkin yang dikenal dengan populasi. Individu yang terdapat dalam satu populasi disebut dengan istilah kromosom. Kromosom ini merupakan suatu solusi yang masih berbentuk simbol. Populasi awal dibangun dengan secara acak, sedangkan populasi berikutnya merupakan hasil evolusi kromosom-kromosom melalui iterasi yang disebut dengan istilah generasi. Pada setiap generasi, kromosom akan melalui proses evaluasi dengan menggunakan alat ukur yang disebut dengan fungsi fitness. Nilai fitness dari suatu kromosom akan menunjukan kualitas kromosom dalam populasi tersebut.

Generasi berikut dikenal dengan istilah anak (*offspring*) terbentuk dari gabungan dua kromosom generasi sekarang yang bertindak sebagai induk (*parent*) dengan menggunakan operator penyilangan (*crossover*). Selain operator penyilangan, suatu kromosom juga dapat dimodifikasi dengan menggunakan operator mutasi. Generasi baru dibentuk dengan menyeleksi nilai fitness sehingga ukuran populasi konstan. Setelah beberapa generasi, maka algoritma ini akan konvergen ke kromosom terbaik.

2.2.1.1.2. Teknik Penyandian

Teknik penyandian disini meliputi penyandian gen dari kromosom. Gen merupakan bagian dari kromosom. Satu gen biasanya akan mewakili satu variabel.

Gen dapat dipresentasikan dalam bentuk: string bit, pohon, array bilangan real, daftar aturan, elemen permutasi, elemen program, atau representasi lainnya yang dapat diimplementasikan untuk operator genetika.

Beberapa skema pengkodean yang paling umum digunakan:

1. *Binary Encoding*, setiap kromosom adalah suatu string dari bit 0 atau 1
   Koromosom A 101 100 011 100
   Koromosom B 110 111 111 011
2. *Permutation Encoding*, setiap kromosom merupakan string dari sejumlah angka/nomor yang merepresentasikan suatu posisi dalam suatu urutan.
   Koromosom A 1 2 3 4 6 7 5 9 8
   Koromosom B 2 4 6 7 8 9 1 5 3
3. *Value* Encoding, setiap kromosom adalah suatu urutan dari beberapa nilai. Nilai tersebut bergantung pada permasalahan, seperti bilangan real, karakter dan objek lainnya.
   Koromosom A 1.2324 5.3243 0.4556
   Koromosom B ABDFDFAFFDJHAD
   Koromosom C (right), (left), (left), (right).

2.2.1.1.3. Prosedur Inisialisasi

Ukuran populasi tergantung pada masalah yang akan dipecahkan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian harus dilakukan inisialisasi terhadap kromosom yang terdapat pada populasi tersebut. Inisialisasi kromosom dilakukan secara acak, namun demikian harus tetap memperhatikan domain solusi dan kendala permasalahan yang ada.

2.2.1.1.4. Fungsi Evaluasi

Suatu individu dievaluasi berdasarkan fungsi tertentu sebagai ukuran performansinya. Di dalam evolusi alam, individu yang bernilai fitness tinggi yang akan bertahan hidup. Sedangkan individu yang bernilai fitness rendah akan mati. Pada masalah optimasi, jika solusi yang dicari adalah memaksimalkan sebuah fungsi *h* (dikenal dengan masalah maksimasi), maka nilai fitness yang digunakan adalah nilai dari fungsi *h* tersebut, yakni *f=h* (dimana *f* adalah nilai fitness). Tetapi jika masalahnya adalah minimalkan fungsi *h* (masalah minimasi), maka fungsi *h* tidak bisa digunakan secara langsung. Hal ini disebakan adanya aturan bahwa individu bernilai fitness tinggi lebih mampu bertahan pada generasi berikutnya. oleh karena itu nilai fitness yang digunakan adalah *f=1/h,* yang artinya semakin kecil nilai *h,* semakin besar nilai *f*. Tetapi hal ini akan menjadi masalah jika *h* bernilai 0, yang mengakibatkan *f* bisa bernilai tak hingga. Untuk mengatasinya, *h* perlu ditambah sebuah bilangan yang dianggap sangat kecil sehingga nilai fitnessnya menjadi *f=1/(h=a)*, di mana *a* adalah bilangan yang dianggap sangat kecil dan bervariasi sesuai dengan masalah yang akan diselesaikan.

2.2.1.1.5. Seleksi Orang Tua

Pemilihan dua buah kromosom sebagai orang tua, yang akan dipindahsilangkan, biasanya dilakukan secara proporsional sesuai dengan nilai fitnessnya.

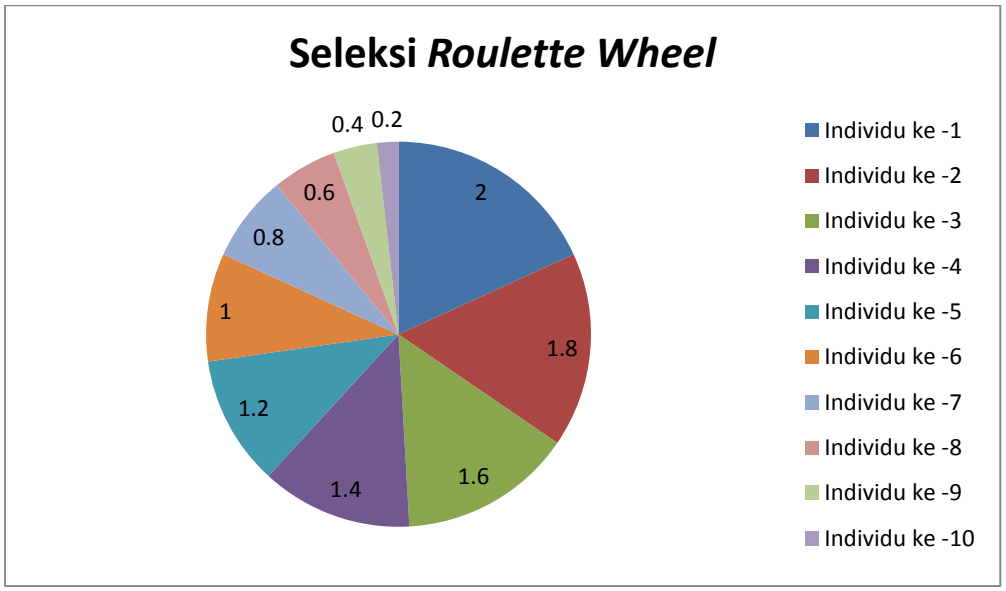
Beberapa metode yang biasa digunakan :

1. Seleksi *Roulette-wheel*

Sesuai dengan namanya, metode ini menirukan permainan *roulette-wheel* di mana masing-masing kromosom menempati potongan lingkaran pada roda roulette secara proporsional sesuai dengan nilai fitnessnya. Kromosom yang memiliki nilai fitness paling tinggi menempati potongan lingkaran yang lebih besar dibandingkan dengan kromosom yang bernilai fitness rendah.

Tabel 2.2 Probabilitas seleksi & nilai fitness [5]

Individu ke-	1	2	3	4	5	6	7	8	9	10	11
Nilai Fitness	2	1.8	1.6	1.4	1.2	1.0	0.8	0.6	0.4	0.2	0.0
Probabilitas Seleski	0.18	0.16	0.15	0.13	0.11	0.09	0.07	0.06	0.03	0.02	0.0



Gambar 2.2 Seleksi *Roulette Wheel*

Metode *roullette-wheel* sangat mudah diimplementasikan dalam pemrograman. Pertama, dibuat interval nilai kumulatif (dalam interval [0,1]) dari nial fitness masing-masing kromosom dibagi total nilai fitness dari semua kromosom. Sebuah kromosom terpilih jika bilangan random yang dibangkitkan berada pada interval nilai akumulatifnya.

2. *Tournament*

Pada seleksi alam yang terjadi di dunia nyata, beberapa individu (biasanya individu jantan) berkompetisi dalam sebuah kelompok kecil sampai tersisa hanya satu individu pemenang. Individu pemenang inilah yang bisa kawin (pindah silang), metode tournamen mengadopsi karakteristik alami ini. Dalam bentuk paling sederhana, metode ini mengambil dua kromosom secara random dan kemudian menyeleksi salah satu yang bernilai fitness paling tinggi untuk menjadi orang tua pertama. Cara yang sama dilakukan lagi untuk mendapatkan orang tua yang kedua.

Metode tournament yang lebih rumit adalah dengan mengambil *m* kromosom secara random. Kemudian kromosom bernilai fitness tertinggi dipilih menjadi orang tua jika nilai random yang dibangkitkan kurang dari *p*. Apbila lebih maka dipilih kembali kromosom selain kromosom yang bernilai fitness tertinggi. *P* adalah probabilitas tournament yang biasanya diset sekitar 0,75. sedangkan *m* atau ukuran grup biasanya diset sebagai nilai yang sangat kecil, misal 4 atau 5.

2.2.1.1.6. Penentuan Parameter

Parameter disini adalah parameter kontrol algoritma genetika, yaitu: ukuran populasi (*posize*), peluan *crossover* (Pc), dan peluang mutasi (Pm). Nilai parameter ini ditentukan juga berdasarnya permasalahna yang akan dipercahkan. Ada beberapa rekomendasi yang bisa digunakan, antara lain:

1. Untuk permasalahan yang mewakili kawasan solusi cukup besar, De Jong merekomendasikan untuk nilai parameter kontrol:

   (Poposize; Pc; Pm) = (50; 0.6, 0.001)

2. Bila rata-rata fitness setiap generasi digunakan sebagai indikator, maka Grefensette merekomendasikan:

   (Poposize; Pc; Pm) = (30; 0.95, 0.01)

3. Bila fitness dari individu terbaik dipantau pada setiap generasi, maka usulannya adalah:

   (Poposize; Pc; Pm) = (80; 0.45, 0.01)

4. Ukuran populasi sebaiknya tidak lebih kecil dari 30, utnuk sembarang jenis permasalahan.

2.2.1.1.7. Pindah Silang

Salah satu komponen paling penting dalam algoritma genetika adalah *crossover* atau pindah silang. Sebuah kromosom yang mengarah pada solusi yang bagus bisa diperoleh dari proses memindah-silangkan dua buah kromosom. Beberapa metode yang terdapat pada pindah silah yaitu [5]:

1. Penyilangan satu titiik (*Single-point Crossover*)

   Pada penyilangan satu titik, posisi penyilangan k (k=1, 2… N-1) dengan N= panjang kromosom diseleksi secara random. Variabel-variabel ditukar antar kromosom pada titik tersbut untuk menghasilka-kan anak. Contoh ada 2 kromosom dengan panjang 12:

   Induk 1 0 1 1 1 0 | 0 1 0 1 1 1 0
   Induk 2 1 1 0 1 0 | 0 0 0 1 1 0 1

   Setelah penyilangan, diperoleh kromosom- kromosom baru:

   Anak 1 0 1 1 1 0 | 0 0 0 1 1 0 1
   Anak 2 1 1 0 1 0 | 0 1 0 1 1 1 0

2. Penyilangan banyak titik (*Multi-point crossover*)

   Pada penyilangan banyak titik, m posisi penyilngan ki (k = 1, 2… N-1, i=1,2,….m) dengan N= panjang kromosom diseleksi secara random dan tidak diperbolehkan ada posisi yang sama, serta diurutkan naik. Variabel-variabel ditukar antar kromosom pada titik tersebut untuk menghasilkan anak. Contoh ada 2 kromosom dengan panjang 12:

   Induk 1 0 1 1 1 0 0 1 0 1 1 1 0
   Induk 2 1 1 0 1 0 0 0 0 1 1 0 1

   Posisi penyilangan yang terpilih:

   Misalkan (m=3): 2 6 10

   Setelah penyilngan, diperoleh kromosom-kromsom baru:

   Anak 1 0 1 | 0 1 0 0 | 1 0 1 1 | 0 1
   Anak 2 1 1 | 1 1 0 0 | 0 0 1 1 | 1 0

Pindah silang bisa juga berakibat buruk jika ukuran populasinya sangat kecil. Dalam suatu populasi yang sangat kecil, suatu kromosom dengan gen-gen yang mengarah ke solusi akan sangat cepat menyebar ke kromosom-kromosom lainnya. Untuk mengatasi masalah ini digunakan suatu aturan bahwa pindah silang hanya bisa dilakukan dengan suatu probabilitas tertentu Pc. Artinya, pindah silang bisa dilakukan hanya jika suatu bilangan random [0,1] yang dibangkitkan kurang dari Pc yang ditentukan. Pada umumnya Pc diset mendekati 1, misalnya 0, 8.

2.2.1.1.8. Mutasi

Setelah mengalami proses penyilangan, pada anak (*offspriing*) dapat dilakukan mutasi. Variabel anak (*offspring*) dimutasi dengan menambahkan nilai random yang sangat kecil (ukuran langkah mutasi), dengan probabilitas mutasi yang rendah. Peluang mutasi(Pm) didefinisikan sebagai persentasi dari jumlah total gen pada populasi yang mengalami mutasi. Peluang mutasi mengendalikan banyaknya gen baru yang akan dimunculkan untuk dievaluasi. Jika peluang mutasi terlalu kecil, banyak gen yang mungkin berguna tidak pernah dievaluasi. Tetapi bila peluang mutasi ini terlalu besar, maka akan terlalulu banyak gangguan acak, sehingga anak akan kehilangan kemiripan dari berlajar dari histori pencarian. Ada beberapa pendapat mengenai laju mutasi ini. Ada yang berpendapat bahawa, laju mutasi sebesar 1/n akan memberikan hasil yang cukup baik. Ada juga yang beranggapan bahwa laju mutasi tidak tergantung pada ukuran populasinya. Kromosom hasil mutasi harus diperiksa, apakah masih berada pada domain solusi, dan bila perlu bisa dilakukan perbaikan.

Mutasi ini berperan untuk menggantikan gen yang hilang dari populasi akibat proses seleksi yang memungkinkan munculnya kembali gen yang tidak muncul pada inisialisasi populasi. Ada 2 tipe mutasi menurut tipe bilangannya [5]:

1. Mutasi Bilangan Real

   Pada mutasi bilangan real, ukuran langkah mutasi biasanya sangat sulit ditentukan. Ukuran yang kecil bisanya sering mengalami kesuksesan, namun adakalanya ukuran yang lebih besar akan berjalan lebih cepat.

   Operator mutasi untuk bilangan real ini dapat ditetapkan sebagai:

a. Variabel yang dimutasi = variabel ± range*delta; (+ atau – memiliki probalitas yang sama).
b. Range = 0.5 * domain variabel; (interval pencarian).
c. Delta = ∑(ai * 2); ai= 1 dengan probabilitas 1/m, selain itu ai=0, dengan m=20;

   Mutasi bilangan real dapat juga di lakukan dengan menggunakan metode non-uniform mutation [2][8]. Operasi mutasi non uniform dilakukan sebagai berikut:

   Jika$C_v^t = (v1 \dots vm)$

   Adalah sebuah kromosom dimana t=generasi, dan elemen vk dipilih untuk dimutasi maka akan didapat sebuah vector$C_v^{t+1} = (v1..vk..vm)$ dimana

$$v_k^t = \begin{cases} vk + \Delta(t, UB - vk)(pesamaan\ ke\ 1) \\ vk - \Delta(t, vk - LB)(persamaan\ ke2) \end{cases} \qquad \text{(persamaan 2.1)}$$

   jika a adalah sebuah bilang random 0 dan 1, maka jika a=0 pilih persamaan ke 1 ke 1, a=1 pilih persamaan ke 2. Fungsi $\Delta(t, y)$ akan mengembalikan nilai pada rentang [0,y], sehingg peluang Fungsi $\Delta(t, y)$ mendekati nol akan meningkat seiring dengan bertambahnya nilai t.

$$\Delta(t, y) = y * (1 - r^{(1-\frac{t}{T})^b} \qquad \text{(persamaaan 2.2)}$$

   Dimana r adalah sebuah bilangan acak [0...1], T adalah generasi maximum, b adalah sebuah parameter yang ditentukan untuk menyatakan derajat ketergantungan terhadap iterasi.

2. Mutasi Biner

Cara sederhana untuk mendapatkan mutasi biner adalah dengan mengganti satu beberapa nilai gen dari kromosom. Langkah- langkah mutasi ini adalah:

a. Hitung jumlah gen pada populasi (panjang kromosom dikalikan dengan ukuran populasi).
b. Pilih secara acak gen yang akan dimutasi.
c. Tentukan kromosom dari gen yang terpilih untuk dimutasi.
d. Ganti nilai gen (0 ke 1, atau 1 ke 0) dari kromosom yang akan dimutasi tersebut.

2.2.1.1.9. Penggantian Populasi

Dalam algoritma genetika dikenal skema penggantian populasi yang disebut *generational peplacement*, yang berarti semua individu (misal N individu dalam suatu populasi) dari suatu generasi digantikan sekaligus oleh N individu baru hasil pindah silang dan mutasi. Skema penggantian ini tidak realistis dari sudut pandang biologi. Di dunia nyata, individu-individu dari generasi berbeda bisa berada pada waktu yang bersamaan. Fakta lainnya adalah individu-individu muncul dan hilang secara konstan, tidak pada generasi tertentu. Secara umum skema pergantian populasi dapat dirumuskan berdasarkan suatu ukuran yang disebut *generational gap* G. Ukuran ini meunjukan persentase populasi yang digantikan dalam setiap generasi. Pada skema *generational replacement*, $G = 1$.

Skema pergantian yang paling ekstrem adalah hanya mengganti satu individu dalam setiap generasi, yaitu $G = 1/N$, dimana N adalah jumlah individu dalam populasi. Skema pergantian ini disebut sebagai *Steady-state reproduction.* Pada skema tersebut, G biasanya sama dengan $1/N$ atau $2/N$. Dalam setiap generasi, sejumlah NG individu harus dihapus untuk menjaga ukuran populasi tetap N. Terdapat beberapa prosedur penghapusan individu, yaitu penghapusan individu yang bernilai fitness paling rendah atau penghapusan individu yang paling tua. Penghapusan bisa berlaku hanya pada

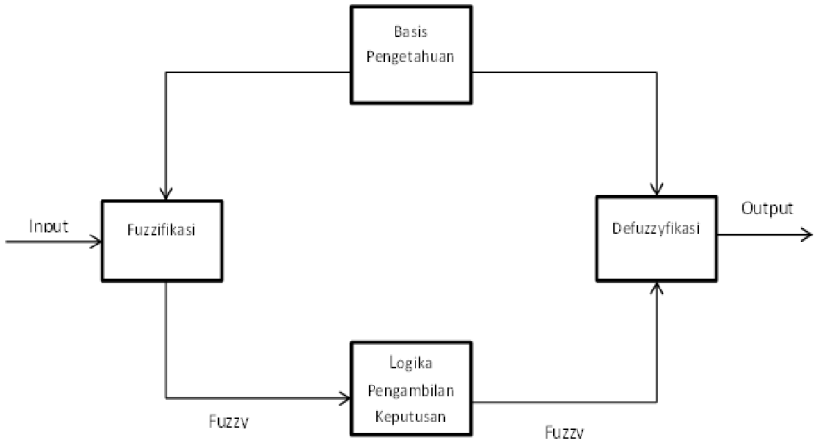
individu orang tua saja atau bisa juga berlaku pada semua individu dalam populasi.

2.2.1.2. Logika Fuzzy

Konsep tentang logika *fuzzy* diperkenalkan oleh Prof. Lotfi Astor Zadeh pada 1962. Logika *fuzzy* [5] adalah metodologi sistem kontrol pemecahan masalah, yang cocok diimplementasikan pada sistem, mulai dari sistem yang sederahana, sistem kecil, embedded system, jaringan PC, multi-channel atau workstation berbasis akurasi data, dan sistem kontrol[5]. Metodologi ini dapat diterapkan pada perangkat keras, perangkat lunak, atau kombinasi keduanya. Dalam logika klasik dinyatakan bahwa segala sesuatu bersifat biner, yang artinya adalah hanya mempunyai dua kemungkinan, "Ya atau Tidak", "Benar atau Salah", "Baik atau Buruk" dan lain-lain. Oleh karena itu, semua ini dapat mempunyai nilai keanggotaan 0 dan 1. Akan tetapi, dalam logika *fuzzy* memungkinakan nilai keanggotaan berada diantara 0 dan 1. Artinya, bisa saja suatu keadaan mempunyai dua nilai "Ya dan Tidak", "Benar dan Salah", "Baik dan Buruk" secara bersamaan, namun besar nilainya tergantung pada bobot keanggotaan yang dimilikinya. Logika *fuzzy* dapat digunakan diberbagai bidang, seperti pada sistem diagnosis penyakit (dalam bidang kedokteran), pemodelan sistem pemasaran, riset operasi (dalam bidang ekonomi), kendali kualitas air, prediksi adanya gempa bumi, klasifikasi dan pencocokan pola (dalam bidang teknik) [5].

Konsep himpunan *fuzzy* memiliki 2 atribut, yaitu [5]:

1. Linguistik, yaitu nama suatu kelompok yang me wakili suatu keadaan tertentu dengan menggunakan bahasa alami, misalnya dingin, sejuk, panas mewakili variable temperatur.
2. Numeris, yaitu suatu nilai yang menunjukan ukuran dari suatu variabel, misalnya 10, 35, 40 dan sebagainya. Cara kerja logika fuzzy dapat digambarkan sebagai struktur elemen dasar sistem inferensi *fuzzy* seperti pada gambar 2.3:



Gambar 2.3 Logika *fuzzy* [5]

2.2.1.2.1. Basis Pengetahuan Fuzzy

Fakta adalah bagian dari pengetahuan yang memuat informasi mengenai objek, peristiwa, atau situasi. Sedangkan kaidah (Rule base) berisi informasi tentang cara membangkitkan fakta baru atau hipotesa yang suda ada. Dalam pendefinisian tersebut biasanya dilakukan secara subjektif dan didasrkan pada pengalaman dan petimbangan yang menyangkut kerekayasaan, sehingga bergantung penuh pada perancang [5]. Kaidah atau rule base dalam fuzzy biasanya tersusun dengan pernyataan:

IF (antecedent) Then (consequent) atau dapat juga

IF x is A Then y is B

Keterangan:

Antecedent: berisi himpunan fakta masukan (sebab)

Consequent: berisi himpunan fakta keluaran (akibat).

2.2.1.2.2. Fuzzyfikasi

Proses untuk mengubah input sistem yang mempunyai nilai tegas (nilai *crisp*) kedalam himpunan fuzzy yang sesuai dengan variasi semesta pembicaraan masukan. Data yang telah dipetakan selanjutnya dirubah kedalam

variabel linguistic menggunakan fungsi keanggotaan yang disimpan dalam basis pengetahuan *fuzzy*.

2.2.1.2.3. Logika Pengambilan Keputusan

Sering pula disebut sebagai Fuzzy Inference System (FIS) merupakan bagian terpenting dalam logika fuzzy. Logika pengambil keputusan merupakan kerangka komputasi yang didasarkan pada teori hmpunan fuzzy, aturan berbentuk IF-THEN, dan penalaran fuzzy. Fuzzy Infrerence system(FIS) menerima masukan crisp. Masukan ini kemudian dikirim ke basis pengetahuan yang berisi n aturan fuzzy berbentuk IF-THEN. Apabila jumlah aturan ebih dari satu, maka akan dilakukan agregasi dari semua aturan. Suatu aturan juga dapat mempunyai beberapa kondisi masukan, yang satu sama lainnya dihubungkan dengan AND atau OR untuk mendapatkan rule keluaran. Selanjutnya pada hasil agregasi akan dilakukan untuk mendapatkan nilai crisp sebagai keluaran sistem. Pada Fuzzy Inference system(FIS) dikenal adanya metode Sugeno dan metode Mamdani[5].

Metode sugeno diperkenalkan oleh Takagi-Sugeno Kang pada tahun 1985. Keluaran (output) yang diproses dengan metode ini adalah berupa konstanta atau persamaan linear.

Metode Mamdani sering juga dikenal dengan Metode Max-Min. Metode ini diperkenalkan oeleh Ebrahim Mamdani pada tahun 1975. Pada metode mamdani keluaran(output) yang dihasilkan berupa himpunan fuzzy.

2.2.1.2.4. Defuzzyfikasi

Merupakan proses mengubah output *fuzzy* yang diperoleh dari mesin inferensi menjadi nilai tegas menggunakan fungsi keanggotaan yang sesuai dengan saat dilakukan *fuzzy*fikasi.

2.2.1.2.5. Fungsi keanggotaan (Member Function)

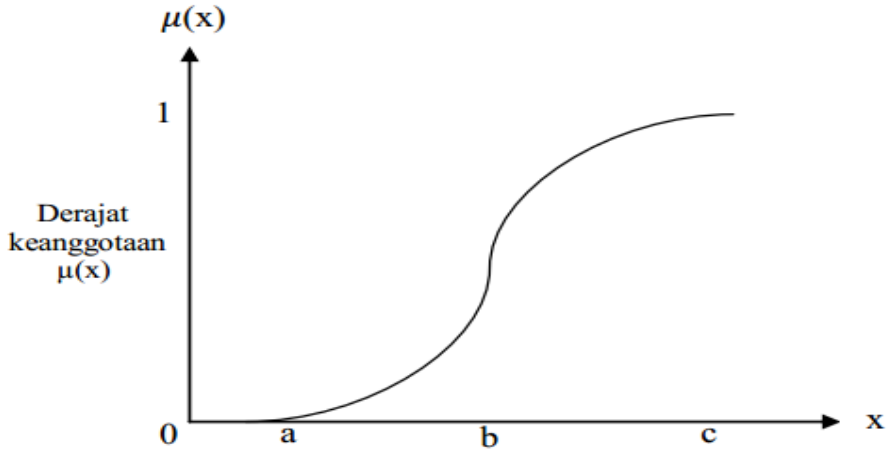
Pada fuzzy system, fungsi keanggotaan memainkan peran yang sangat penting untuk merepresentasikan masalah dan menghasilkan keputusan yang

akurat. Terdapat banyak sekali fungsi keanggotaan yang dapat digunakan, diantaranya adalah [5]:

1. Fungsi sigmoid

   Sesuai dengan namanya, fungsi ini berbentuk kurva sigmoidala seperti huruf

S. Setiap nilai x (anggota crips set) dipetakan ke dalam interval [0,1]. Grafik dan notasi matematika untuk fungsi ini adalah sebagai berikut [5] :

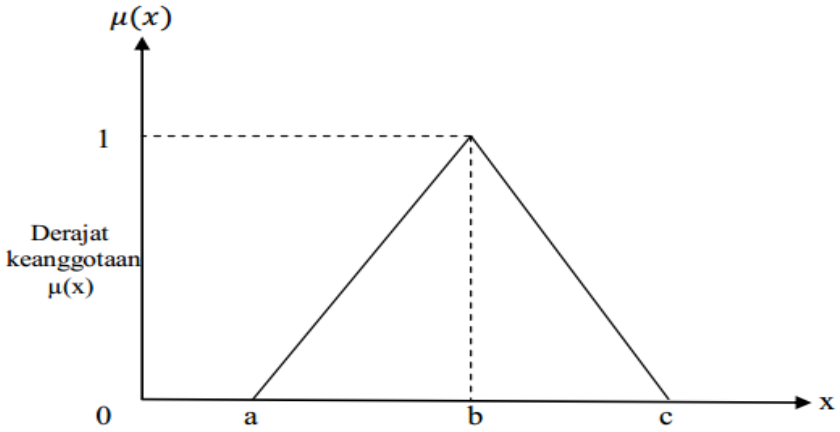


Gambar 2.4 Fungsi sigmoid

$$Sigmoid(x,a,b,c) = \begin{cases} 0; x \le a \\ 2(x-a)/(c-a)^2; a \le x \le b \\ 1-2((c-x)/(c-a))^2; b \le x \le c \\ 1; x \ge c \end{cases} \quad \text{(Persamaan 2.9)}$$

2. Fungsi Segitiga

Fungsi segitiga hanya terdapat satu nilai x yang memiliki derajat kenanggotaan sama dengan 1, yaitu ketika x = b. Tetapi, nilai-nilai disekitar b memiliki derajat keanggotaan yang turun cukup tajam (menjauhi 1). Grafik dan notasi matematika dari fungsi segitiga adalah sebagai berikut:

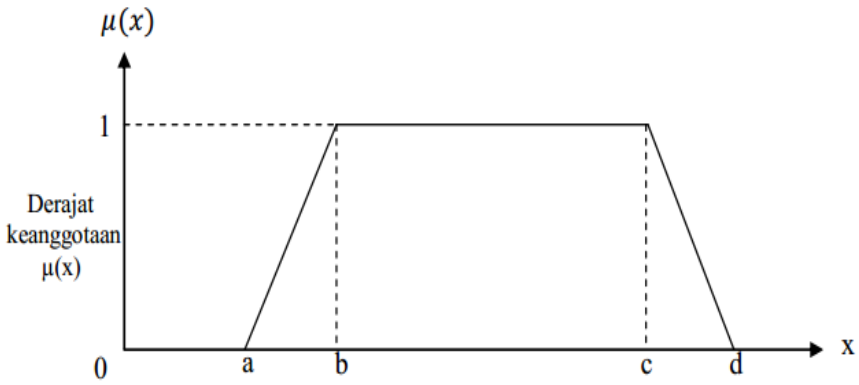


Gambar 2.5 Fungsi Segitiga

$$\mu(x) = \begin{cases} 0;\ x \leq a\ atau\ x\ \geq c \\ (x-a)/(b-a)\,;\, a \leq x \leq b \\ (b-x)/(c-b)\,;\, b \leq x \leq c \end{cases} \qquad \text{(Persamaan 2.10)}$$

3. Fungsi Trapesium

Berbeda dengan fungsi segitiga, pada fungsi ini terdapat beberapa nilai x yang memiliki derajat keanggotaan sama dengan 1, yaitu ketika $b \leq x \geq c$ Tetapi, derajat keanggotaan untuk $c \leq x \geq d$ memiliki karakteristik yang sama dengan fungsi segitiga. Grafik dan notasi matematika dari fungsi Trapesium adalah sebagai berikut [5]:



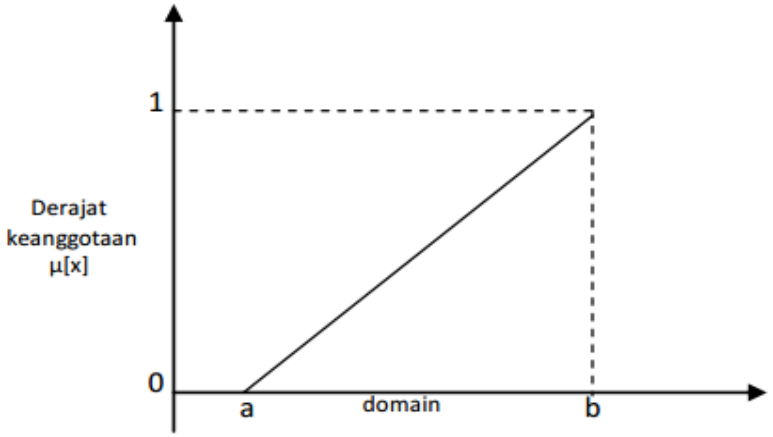
Gambar 2.6 Fungsi Trapesium

Fungsi Keanggotaan :

$$\mu[x] = \begin{cases} 0; & x \le a \text{ atau } x \ge d \\ (x-a)/(b-a); & a \le x \le b \\ 1; & b \le x \le c \\ (d-x)/(d-c); & c \le x \le d \end{cases} \qquad \text{(Persamaan 2.11)}$$

4. Fungsi Linear

Pada fungsi linear, pemetaan input ke derajat keanggotaannya digambarkan sebagai suatu garis lurus. Bentuk kurva ini paling sederhana dan menjadi pilihan yang baik untuk mendekati suatu konsep yang kurang jelas. Ada 2 keadaan himpunan fuzzy yang linear. Pertama, kenaikan himpunan dimulai pada nilai domain yang memiliki derajat keanggotaan nol [0] bergerak ke kanan menuju ke nilai domain yang memiliki derajat keanggotaan lebih tinggi

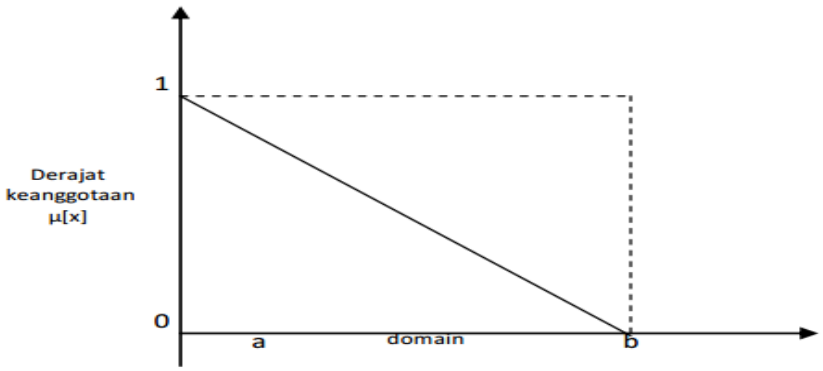


Gambar 2.7 Fungsi linear derajat 0 ke 1

Fungsi keanggotaan :

$$\mu[x] = \begin{cases} 0; & x < a \\ (x-a)/(b-a); & a < x \le b \\ 1; & x \ge b \end{cases} \qquad \text{(Persamaan 2.12)}$$

Kedua, merupakan kebalikan dari yang pertama. Garis lurus dimulai dari nilai domain dengan derajat keanggotaan tertinggi pada sisi kiri, kemudian bergerak menurun ke nilai domain yang memiliki derajat keanggotaan lebih rendah



Gambar 2.8 Fungsi linear derajat 1 ke 0

Fungsi Keanggotaan :

$$\mu[x] = \begin{cases} (b-x)/(b-a); & a \le x < b \\ 0; & x \ge b \end{cases} \qquad \text{(Persamaan 2.13)}$$

2.2.1.3. *Genetic Fuzzy System*

. Sistem Fuzzy-Genetik (SFG) [6] [7] adalah gabungan dari sistem kendali fuzzy dengan algoritma genetik untuk melakukan pembangkitan atau modifikasi basis pengetahuan atau basis data dari sistem kendali *fuzzy*. Basis pengetahuan yang dimaksud mencakup basis aturan dan basis data dari suatu sistem kendali *fuzzy*. Basis data meliputi fungsi penskalaan dan fungsi keanggotaan yang didefinisikan untuk sistem kendali *fuzzy*. Basis aturan adalah kumpulan aturan yang digunakan untuk inferensi dalam sistem kendali fuzzy. Pembangkitan atau modifikasi basis pengetahuan atau basis data dilakukan melalui proses tuningdan atau learning. Proses tuning merupakan proses untuk mengatur kembali parameter dari sistem kendali fuzzy, yaitu modifikasi fungsi

penskalaan dan atau fungsi keanggotaan. Proses learning adalah proses untuk memodifikasi basis aturan dari sistem kendali *fuzzy* [6][7]. Proses tuningsecara genetik terhadap basis data sistem kendali *fuzzy* dapat dibedakan atas approximate genetic tuning dan descriptive genetic tuning. Pada approximate genetic tuning kromosom akan mengkodekan basis pengetahuan secara lengkap (baik premis maupun konklusi dari aturan). Sedangkan pada descriptive genetic tuning kromosom hanya mengkodekan definisi basis data (bagian premis aturan) berdasarkan partisi domain *fuzzy* dari variabel input. Pada proses *learning* secara genetik terhadap basis aturan dapat dilakukan tiga macam pendekatan belajar, yaitu pendekatan Michigan, pendekatan Pittsburgh dan pendekatan *Iterative Rule Learning* (IRL). Kromosom pada pendekatan Michigan akan merepresentasikan suatu aturan sebagai suatu individu, sehingga populasi dinyatakan sebagai kumpulan aturan. Pendekatan ini digunakan dalam proses yang bersifat on-linedan dikenal sebagai sistem klasifikasi (*classifier system*). Pada pendekatan Pittsburgh, kromosom akan merepresentasikan seluruh basis aturan sebagai suatu individu, sehingga populasi dinyatakan sebagai kumpulan basis aturan. Dalam hal ini operasi genetik dilakukan untuk menghasilkan kombinasi aturan yang baru. Pendekatan ini digunakan dalam proses yang bersifat *off-line*. Pada pendekatan IRL, kromosom akan merepresentasikan suatu aturan sebagai suatu individu, sehingga populasi dinyatakan sebagai kumpulan aturan. Pembentukan basis aturan dilakukan dengan memilih hanya individu terbaik pada setiap iterasi dan membuang kumpulan individu sisanya sehingga menghasilkan solusi parsial. Pendekatan ini digunakan dalam proses yang bersifat *off-line*. Tujuan dari proses tuning aturan kendali dalam sistem kendali fuzzy adalah untuk membuat sistem inferensi *fuzzy* memberikan hasil yang optimal dengan cara mengatur kembali parameter fungsi keanggotaan yang dipakai dalam sistem. Untuk itu diperlukan sekumpulan data training, yaitu pasangan data input dan output yang dihasilkan dari sistem kendali *fuzzy*. Dalam hal ini data input berhubungan dengan nilai variabel input dan data output dianggap sebagai nilai output yang diharapkan dapat dihasilkan oleh sistem. Proses tuning akan

memodifikasi parameter himpunan fuzzy dari variabel input sehingga sistem dapat memberikan output yang mendekati data training yang diberikan.

2.2.2. Pemrograman Berorientasi Objek (*Object Oriented Programming*)

Pemrograman berorientasi objek adalah metode implementasi dimana pemogram diorganisasaikan sebagai komunitas objek yang bekerja sama berinteraksi.Masing-masing merepresentasikan instan suatu kelas yang menyediakn sejumlah layanan yang relatif sederhana.Objek-objek saling berkomunikasi untuk mempertukarkan informasi.Prilaku kompleks diperolah dengan kerjasama antar objek-objek.Kelas-kelas dapat merupakan anggota hirarki yang disatukan lewat hubungan pewarisan.

Tiga aspek penting dalam pemrograman beriorientasi objek:

1. Menggunakan objek-objek bukan algoritma-algoritma sebagai blok-blok bangunnan logik dasar (hirarki "*part of*").
2. Masing-masing objek adalah instan suatu kelas.
3. Kelas-kelas saling berhubungan lewat keterhubungan perwarisan ("*is a*").

Stroustrup mendeskripsikan permograman beriorientasi objek sebagai penggabungan abstraksi data dengan pewarisan. Fitur-fitur penting untuk mendukung abstraksi data adalah *constructor* da*n destructor* untuk penciptaan dan pemusnahan objek*,* tipe berparamete*r, mekanisme exception, object, coercions* dan *iteraror*. Defenisi tersebut mengarah ke bahasa yang mendukung entitas-entitas independen yang disebut objek. Kelakukan objek harus didefenisikan tipe yang dapat didefenisiskan oleh program menggunakan bahasa.Tipe-tipe ini dapat dihubungkan melalui mekanisme pewarisan.

Tujuan nyata dari pemrogram adalah menciptakan kode yang benar (yaitu seluruhnya memenuhi kebutuhan), sementara kompilator hanya sanggup memeriksa sintaks dan membangkitkan kodek objek.

2.2.3. UML

Unified Modeling Language (UML) adalah bahasa grafis untuk mendokumentasi, menspesifikasikan, dan membangun sistem perangkat lunak. UML berorientasi objek, menerapkan banyak level abstraksi, tidak bergantung proses pengembangan, tidak tergantung bahasa dan teknologi, pemaduan beberapa notasi di beragam metodologi, usaha bersama dari banyak pihak, didukung oleh kakas-kakas yang diintegrasikan lewat XML.Standar UML dikelola oleh OMG (*Object Management Group*).

UML adalah bahasa pemodelan untuk menspesifikasikan, memvisualisasikan, membangun dan mendokumentasikan artifak-artifak dari sistem

1. Di dalam *system intensive process*, metode diterapkan sebagai proses untuk menurunkan atau mengevolusikan sistem.
2. Sebagai bahasa, UML digunakan untuk komunikasi yaitu alat untuk menangkap pengetahuan (semantiks) mengenai satu obyek dan mengekspresikan pengetahuan (sintaks) yang mempedulikan subyek untuk maksud berkomunikasi. Subyek adalah sistem yang dibahas.
3. Sebagai bahasa pemodelan, UML fokus pada pemahaman subyek melalui formulasi model dari subyek (dan konteks yang terhubung).Model memuat pengetahuan pada subyek, dan aplikasi dari pengetahuan ini berkaitan dengan intelejensia.
4. Berkaitan dengan unifikasi, UML memadukan praktek rekayasa terbaik sistem informasi dan industri, meliputi beragam tipe sistem (perangkat lunak dan non perangkat lunak), domain (bisnis, perangkat lunak) dan proses siklus hidup.
5. Begitu diterapkan untuk menspesifikasikan sistem. UML dapat digunakan untuk mengkomunikasi “apa” yang diperlukan dari sistem dan “bagaimana” sistem dapat direalisasikan.
6. Begitu diterapkan untuk memvisualisasikan sistem, UML dapat digunakan untuk menjelaskan sistem secara visual sebelum direalisasikan.
7. Begitu diterapkan untuk membangun sistem, UML dapat digunakan untuk memandu realisasi sistem serupa dengan “*blueprint*”.

8. Begitu diterapkan untuk mendokumentasikan sistem, UML dapat digunakan untuk menangkap pengetahuan mengenai sistem pada selururh siklus hidup.

2.2.3.1. Tujuan UML

Tujuan utama perancanagn UML adalah:

1. Menyediakan bahasa pemodelan visual yang ekspresif dan siap pakai untuk mengembangkan dan pertukaran model-model yang berarti.
2. Menyediakan mekanisme perluasan dan spesialisasi untuk memperluas konsep-konsep inti.
3. Mendukung spesifikasi independen bahasa pemrograman dan proses pengembangan tertentu.
4. Menyediakan basis formal untuk pemahaman bahasa pemodelan.
5. Mendorong pertumbuhan pasar kakas berorientasi objek.
6. Mendukung konsep-konsep pengembangan level lebih tinggi seperti komponen, kolaborasi, *framework* dan *pattern*.

2.2.3.2. Diagram UML

Kegunaan diagram pada pemodelan adalah untuk formulisasi ekspresi model objek secara koheren, presisi dan mudah dirumuskan. Pemodelan berorientasi objek memerlukan kakas untuk mengekspresikan model. UML menyediakan sejumlah diagran untuk mengekspresikan pemodelan berorientasi objek yang dilakukan.

1. Use case diagram

Diagram *use-case* digunakan untuk mendeskripsikan apa yang seharusnya dilakukan oleh sistem. Diagram *use-case* menyediakan cara mendeskripsikan pandangan eksternal terhadap sistem dan interaksi –interaksinya dengan dunia luar. Dengan cara ini, diagram *use-case* menggantikan diagram konteks pada pendekatan konvensional.

Pemodelan ini biasa dilakukan lewat proses berulang interaksi antara pengembang dan pemakai untuk memperoleh spesifikasi kebutuhan yang sama-sama disepakati. Pemodelan *use-case* diciptakan oleh Ivar Jacobson.

Tujuan utama pemodelan *use-case* adalah:

a. Memutuskan dan mendeskripsikan kebutuhan-kebutuhan fungsional sistem.
b. Memberikan deskripsi jelas dan konsisten dari apa yang seharusnya dilakukan, sehingga model *use case* digunakan di seluruh proses pengembangan untuk komunikasi dan menyediakan basis untuk pemodelan berikutnya yang mengacu sistem harus memberikan fungsionalitas yang dimodelkan pada *use-case*.
c. Menyediakan basis untuk melakukan pengujian sistem yang memverifikasi sistem.Menguji apakah sistem telah memberikan fungsionalitas yang diminta.
d. Menyediakan kemampuan melacak kebutuhan fingsionalitas menjadi kelas-kelas dan operasi-operasi aktual di sistem.Untuk menyederhanakan perubahan dan ekstensi ke sistem dengan mengubah model *use-case* dan kemudian melacak *use-case* yang dipengaruhi ke perancangan dan implementasi sistem.

Elemen diagram *use-case* adalah:

a) Aktor

Aktor adalah pemakai sistem, dapat berupa manusia atau sistem terotomatisasi lain.Aktor adalah sesuatu atau seseorang yang berinteraksi dengan sistem, yaitu siapa atau apa yang menggunakan sistem.Yang dimaksud dengan berinteraksi adalah aktor mengirim atau menerima pesan ke atau dari sistem, atau mempertukarkan informasi dengan sistem. Dalam perspektif aktor, use-case melakukan sesuatu yang berharga bagi aktor.

Aktor adalah tipe (kelas), bukan instan. Aktor merepresentasikan peran bukan pemakai individu dari sistem.Aktor mempunyai nama. Nama yang dipilih seharusnya menyatakan peran aktor.

b) Use-Case

Use-case adalah cara spesifik penggunaan sistem oleh aktor.Ciri-ciri dari *use-case* adalah:

1. Pola perilaku yang harus dipenuhi oleh sistem.
2. Sekuen transaksi terhubung yang dilakukan aktor dan sistem.
3. Memberikan sesuatu yang berharga bagi aktor.

c) Hubungan antar *use-case*.

Keterhubungan antar *use-case* dengan *user-case* lain berupa generalisasi antar *use-case*, yaitu:

1. *Include*, perilaku *use-case* merupakan bagian dari *use-case* yang lain. Sehingga sebuah use-case yang di include oleh use-case lain akan ikut di eksekusi pada saat use-case tersevut dijalankan.
2. *Extend*, perilaku *use-case* memperluas perilaku *use-case* yang lain. *Use-case* yang diextend akan menjadi sebuah pilihan (optional), use-case yang diextend akan tetap ditampilkan oleh sistem akan tetapi prosesnya tidak akan dilakukan.

2. *Activity Diagram*

Activity diagram menggambarkan berbagai allur aktifitas dalam sistem yang sedang dirancang, bagaimana masing-masing alur berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Aktivity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

Activity diagram merupakan state diagram khusus, dimana sebagian besar state adalah action dan sebagian besar transisi di-triger oleh selesainya state sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan behavior internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur aktivitas dari level atas secara umum.

3. *Sequence diagram*

Sequence diagram mendeskripsikan komunikasi di antara objek-objek, meliputi pesan-pesan yang ada dan urutan pesan tersebut muncul.

Diagram sekuen digunakan untuk:

1. *Overview* perilaku sistem.
2. Menunjukkan objek-objek yang diperlukan.
3. Mendokumentasikan skenario dari suatu diagram use case
4. Memeriksa jalur-jalur pengaksesan

Diagram kelas merupakan diagram paling umum dipakai di semua pemodelan berorientasi objek. Pemodelan kelas merupakan pemodelan paling utama di pendekatan berorientasi objek.Pemodelan kelas menunjukkan kelas-kelas yang ada di sistem dan hubungan antar kelas-kelas itu, atribut-atribut dan operasi-operasi di kelas-kelas.

2.2.4. C#

C# merupakan sebuah bahasa pemrograman yang berorientasi objek yang dikembangkan Microsoft sebagai bagian dari inisiatif kerangka .NET *Framework*. C# adalah Java versi Microsoft, sebuah bahasa multi *flatform* yang didesain untuk bisa berjalan diberbagai mesin. C# memiliki kekuatan bahasa C++ dan portabilitas seperti Java. Fitur-fitur yang diambilnya dari bahasa C++ dan Java adalah desain berorientasi objek, seperti *garbage collection, reflection*, akar kelas (*root class*), dan juga penyerdehanaan terhadap pewarisan jamak (*multiple inheritances)*.

Bahasa pemrograman C# dibuat sebgai bahasa pemrograman yang bersifat *genaral-purpose* (untuk tujuan jamak), berorientasi objek, modern, dan sederhana. C# ditujukan agar cocok digunakan untuk menulis program aplikasi baik dalam sistem *client-server* (*hosted system*) maupun sistem *embedded* (*embbeded system*), mulai dari program aplikasi yang sangat besar yang menggunakan sistem operasi yang canggih hingga kepada program aplikasi yang sangat kecil.

Meskipun aplikasi C# ditujukan agar bersifat ‘ekonomis’ dalam hal kebutuhan pemrosesan dan memori komputer, bahasa C# tidak ditujukan untuk bersaing secara langsung dengan kinerja dan ukuran program aplikasi yang dibuat dengan bahasa pemrograman C.