Computational Intelligence

Assoc Prof. Sansanee Auephanwiriyakul, Ph.D.

Topics

- Introduction to Computational Intelligence
- Introduction to artificial neural networks
  - Neural networks foundation
  - Multi-layer perceptrons
  - Self organizing feature map
- Introduction to fuzzy system
  - Fuzzy set foundation
  - Fuzzy inference system
  - Fuzzy measure & fuzzy integral

Topics

- Introduction to evolutionary computing
  - Introduction to genetic algorithm
  - Introduction to genetic programming
  - Introduction to evolutionary computation
  - Co-evolution
- Introduction to swarm intelligence

References

- [Bezdek94] J. C. Bezdek, "What is computational intelligence?" in *Computational Intelligence: Imitating Life*, J. M. Zurada, R. J. Marks, and C. J. Robinson (Eds.) Piscataway, NJ: IEEE Press, 1994, pp 1 – 11.
- [Bonabeau99] E. Bonabeau, M. Dorigo, and G. Theraulez, *Swarm Intelligence:From Natural to Artificial Systems*, Oxford University Press, Inc., New York, USA., 1999.
- [Clerc02] M. Clerc and J. Kennedy, "The particle swarm: explosion, stability, and convergence in a multi-dimensional complex space", *IEEE Transactions on Evolutionary Computation*, vol 6. 2002, pp. 58 – 73.
- [Dorigo96] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: optimization by a colony of cooperating agents", *IEEE Trans. Syst. Man Cybern. B*, vol 26, 1996, pp. 29 – 41.

References

- [Dorigo99] M. Dorigo, *Artificial Life: The swarm intelligence approach*, Tutorial TD1, Congress on Evolutionary Computing, Washington, DC., 1999.
- [Dubois82] Dubois, D. and Prade, H., "A Class of Fuzzy Measures based on Triangular norms", *Internat. J. General Systems*, 8, 1982, pp. 43-61.
- [Dumitrescu00] D. Dumitrescu, B. Lazzerini, L.C. Jain, and A. Dumitrescu, *Evolutionary Computation*, CRC Press LLC, Florida, USA., 2000.
- [Eberhart07] R. Eberhart and Y. Shi, *Computational Intelligence: Concepts to Implementations*, Morgan Kaufmann Publishers, USA., 2007.
- [Eng05] G. K. Eng, and A. M. Ahmad, "Malay Speech Recognition using Slef-Organizing Map and Multilayer Perceptron", *Proceedings of the Postgraduate Annual Research Seminar*, 2005.

References

- [Engelbrecht02] A. P. Engelbrecht, *Computational Intelligence: An Introduction*, John Wiley & Sons, Ltd., West Sussex, England, 2002.
- [Engelbrecht07] A. P. Engelbrecht, *Computational Intelligence: An Introduction*, John Wiley & Sons, Ltd., West Sussex, England, 2007.
- [Fogel95] D. B. Fogel, "Review of Computational Intelligence: Imitating Life (book Review)", *Proceedings of the IEEE*, Vol. 83, Issue 11, 1995, pp.1588 – 1592.
- [Fogel00a] D. B. Fogel, *Evolutionary Computation: Principles and Practice for Signal Processing*, SPIE-The international Society for Optical Engineering, Washington, USA., 2000.
- [Fogel00b] D.B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*, IEEE Press, USA., 2000.

References

- [Goldberg05] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley Longman, Inc., USA., 2005.
- [Grabisch94] Grabisch, M. and Nicolas, J-M., "Classification by fuzzy integral: Performance and tests", *Fuzzy Set and Systems*, 65, 1994, pp. 255-271.
- [Haykin94] S. Haykin, *Neural Networks: A comprehensive Foundation*, Macmillan Publishing Company, Inc. New Jersey, USA., 1994.
- [Haykin10] S. Haykin, *Neural Networks and Learning Machines*, Pearson Education, Inc., USA., 2010.
- [Karaboga05] D. Karaboga, "An Idea based on Honey Bee Swarm for Numerical Optimization", Techincal Report –TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

References

- [Keller94] Keller, J. M., Gader P., Tahani, H., Chiang, J-H. and Mohaned, M., "Advances in fuzzy integration for pattern recognition", *Fuzzy Sets and Systems*, 65, 1994, pp.273-283.
- [Keller16] Keller, J.M., Liu, D. and Fogel, D.B., *Fundamentals of Computational Intelligence:Neural Networks, Fuzzy Systems, and Evolutionary Computation*, IEEE Press, Wiley, New Jersey, USA., 2016.
- [Kennedy98] J. Kennedy "The behavior of particles" *in VW Porto, N. Saravanan, D. Waagen(eds), Proceedings of the 7th International Conference on Evolutionary Programming*, 1998. pp. 581 – 589.
- [Klir95] G. J. Klir and B. Yuan, *Fuzzy Stes and Fuzzy Logic: Theory and Applications*, Prentice Hall Inc., New Jersey, USA., 1995.
- [Klir97] G. J. Klir, U. H. St. Clair, and B. Yuan, *Fuzzy Set Theory: Foundations and Applications*, Prentice Hall Inc., New Jersey, USA., 1997.

References

- [Kohavi95] R. Kohavi, "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection", *International Joint Conference on Artificial Intelligence (IJCAI)*, 1995
- [Kosko87] B. Kosko, "Fuzzy associative memories", *in Fuzzy Expert Systems*, A. Kandel, Ed. Reading, MA: Addison-Wesley, 1987.
- [Kosko88] B. Kosko, "Bidirectional Associative Memories", *IEEE Transactions on Systems, Man, and Cybernetics*, vol 18(1), 1988, pp. 49 – 60.
- [Koza92] J. R. Koza *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, The MIT Press, England, 1992.
- [Kreyszig05] E. Kreyszig, *Advanced Angineering Mathematics*, John Wiley & Sons, Inc., New Yorj USA., 2005.
- [Kruse95] R. Kruse, J. Gebhardt, and F. Klawonn, *Foundations of Fuzzy Systems*, John Wiley & Sons Ltd., West Sussex, England, 1995.

References

- [Mitchell98] M. Mitchell, *An Introducton to Genetic Algorithms*, MIT Press, Massachusetts Institute Technology, USA., 1998
- [Ohnishi90] N. Ohnishi, A. Okamoto, and N. Sugiem, "Selective Presentation of Learning Samples for Efficient Learning in Multi-Layer Perceptron", *Proceedings of the IEEE International Joint Conference on Neural Networks*, vol 1, 1990, pp. 688 – 691.
- [Rosenblatt58] F. Rosenblatt, "The Perceptron: A propabilistic model for information storage and organization in the brain", *Psychological Review*, vol 65, 1958, pp. 386 – 408.
- [Ross04] T. J. Ross, *Fuzzy Logic with Engineering Applications*, John Wiley & Sons Ltd., West Sussex, England, 2004.
- [Suganthan99] P. N. Suganthan, "Particle swarm optimizer with neighborhood operators", *Proceedings of the IEEE Congress on Evolutionary Computation*, 1999, pp. 1958 – 1961.

References

- [Sugeno77] Sugeno, M., " Fuzzy Measures and Fussy Integrals: A survey", *Fuzzy Automata and Decision Processes*, Amsterdam, North Holland, 1977, pp. 89-102.
- [Tahani90] Tahani, H. and Keller, J. M., "Information Fusion in Computer Vision using the Fuzzy Integral", *IEEE Trans. on Systems, Man, and Cybernetics*, 20(3), 1990, pp.773-741.
- [Theodoridis09] S. Theodoridis and K, Koutroumbas, *Pattern Recognition*, Academin Press, Elsevier Inc., London, UK., 2009.
- [Thodberg91] H. H. Thodberg, "Improving Generalization of Neural Networks through Prunning", *International Journal of Neural Systems*, 1(4), 1991, pp. 317 – 326.
- [Turing50] A. M. Turing, Computing Machinery and Intelligence, *Mind*, vol 59, 1950, pp. 433 – 460.

References

- [Wang92] Wang, Z. and Klir, G., " Fuzzy Measure Theory", New York, Plenum Press, 1992.
- [Zadeh95] L. A. Zadeh, "Fuzzy Sets", *Information and Contro*l, 8(3), 1965,pp 338 -353.

Sansanee Auephanwiriyakul, *Senior Member IEEE*

Introduction to Computational Intelligence

History

- Aristotle (384-322 BC) →explain and codify styles of deductive reasoning→syllogisms
- Ramon Llull (1235-1316) → Ars Magna→ machine consisting of a set of wheels supposed to be able to answer all questions.
- Gottfried Leibniz (1646 – 1716) → calculus philosophicus → a universal algebra that can be used to represent all knowledge in a deductive system
- George Boole (1854) → developed the foundations of propositional logic → part of AI tool
- Gottlieb Ferge (1879) → developed the foundations of predicate calculus→ part of AI tool
- Alan Turing(1950) → definition of AI→study how machinary could be used to mimic processes of the human brain→ Book called “The chemical Basis of Morphogenesis→ artificial life

History

- Artificial intelligence first coined at the Dartmoutn conference organized by John Maccarthy (1956) →father of AI
- 1956 – 1969
  - Biological neurons
    - Rosenblatt → perceptrons
    - Widrow and Hoff →adaline
    - Minsky and Papert (1969) →cause a major set back to artificial neural networks →conclude that the extension of simple perceptrons to multilayer perceptrons "is sterile"
    - NN → hibernation until the mid 1980s
    - Grossberg, Carpenter, Amari, Kohonen and Fukushima→continue researching in NN
    - Hopfield, Hinton and Rumelhart and McLelland (early and mid 1980s) → resurrection of NN research

History

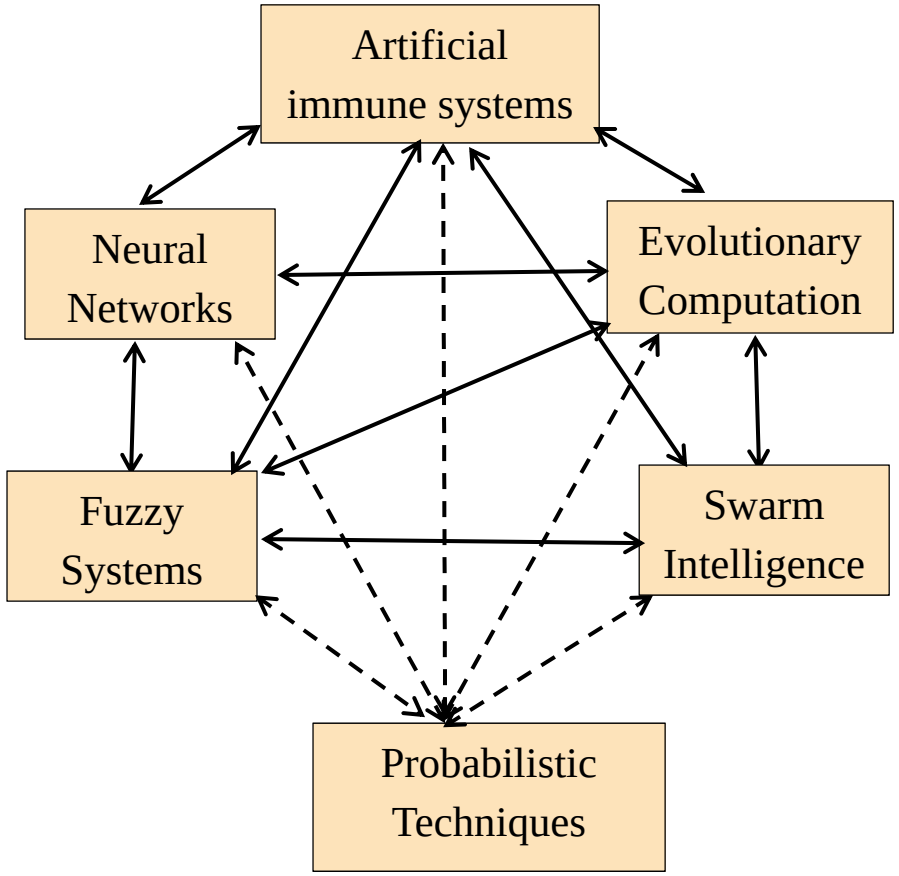
- Fraser, Bremermann and Reed (1950s) →Genetic algorithm
- John Holland → father of Evolutionary Computation (EC)→genetic algorithms
- Rechenberg (1960s) →evolutionary strategies (ES)
- Lawrence Fogel →evolutionary programming→evolved behavior models
- De Jong, Schaffer, Goldberge, Koza, Schwefel, Storn, and Price , →important contricutors in EC field
- Gautama Buddha (563 BC) →described things in shade of gray →starting point of fuzzy logic
- Aristotle→2-valued logic the birth of fuzzy logic
- Lukasiewicz (1920)→3-valued logic
- Max Black→quasi-fuzzy sets

History

- Lotfi Zadeh→contribute most in the field of fuzzy logic→father of Fuzzy Set
- Mamdani, Sugeno, Takagi and Bezdek → famous in the fuzzy field
- 1980 → dark age for fuzzy research→ revive again in the late 1980
- Pawlak (1991)→rough set theory
- Eugene N Marais (1871 – 1936) → poet→introduce swarm intelligence → Books called “The Soul of the White Ant” and “The Soul of the Ape”
- Eberhart and Kennedy (1995) → particle swarm optimization
- Marco Dorigo (1992)→ant colonies

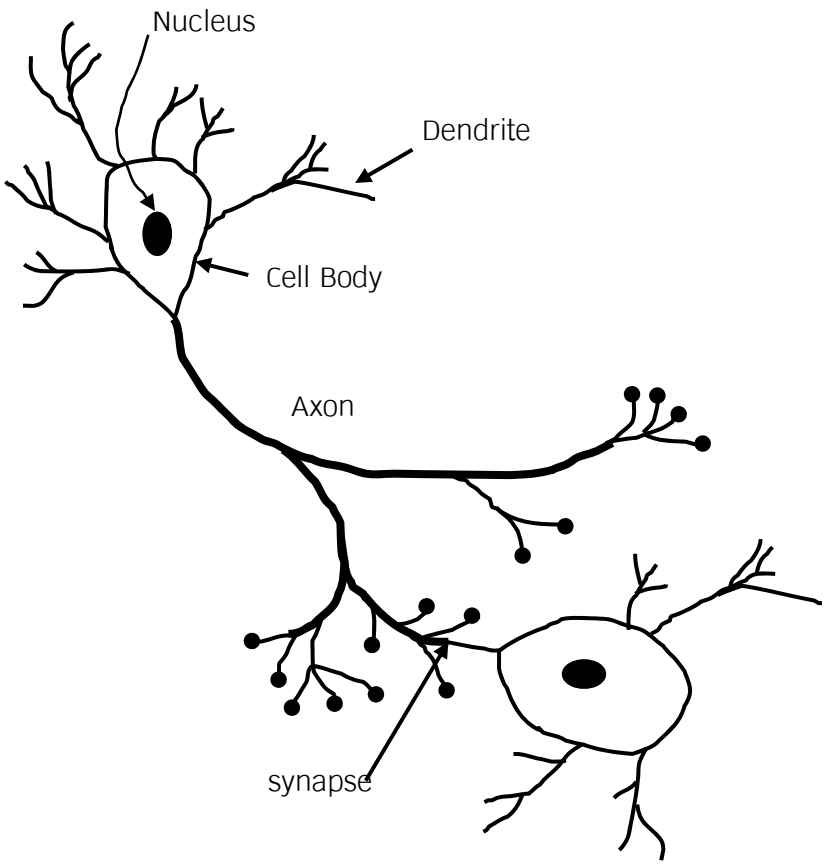
- ETC.
- A lot of people are working in the CI field

Connection in CI



CI → Study of adaptive mechanisms to enable or facilitate intelligent behavior in complex and changing environments

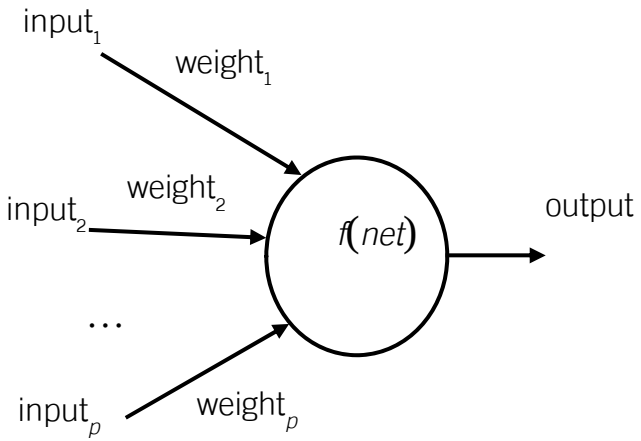
Neural Networks (NN)



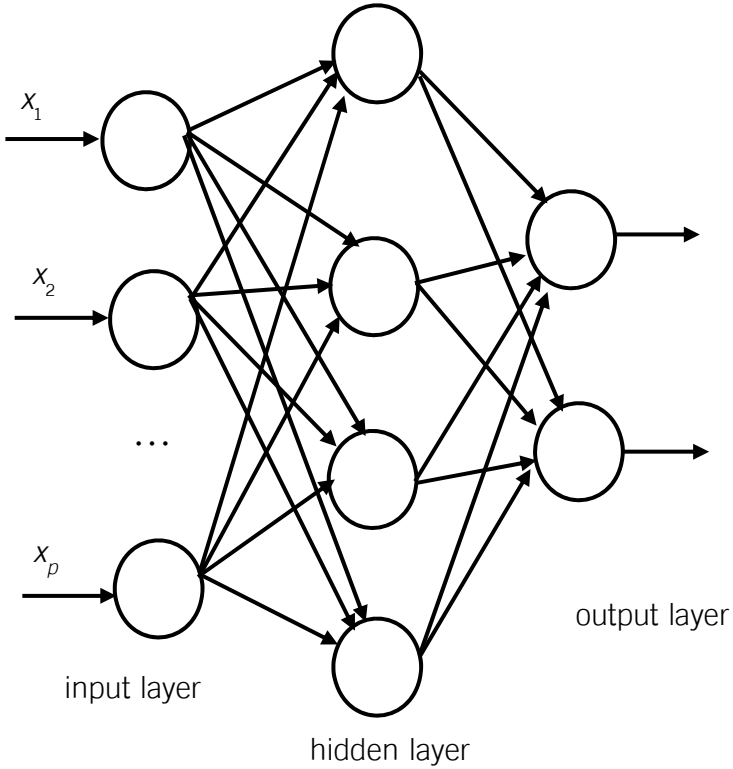
Biological Neural

NN

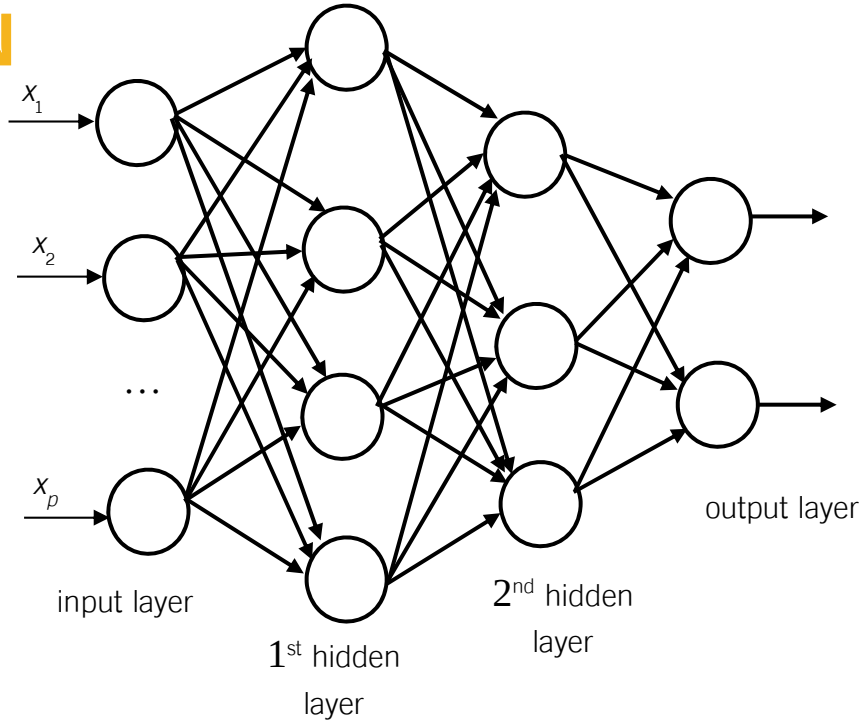
Artificial neuron



NN



p-4-2



p-4-3-2

p-m-2 :p input nodes, m hidden nodes, 2 output nodes

Number of hidden nodes or hidden layers will be shown here→ p-m1-m2-2 means that there are 2 hidden layers: one with m1 nodes and the second one is with m2 nodes

Number of output nodes normally corresponds to the number of classes

Fuzzy Systems

- Not only 0 or 1, there is a gray area
- Unsharp boundary
- Cope with uncertainty
- Approximate reasoning

Evolutionary Computation

- Individual→chromosome→inherit characteristic
- Population of chromosome
- Each characteristic --<gene– allel(gene value)
- Each generation→individual compete to reproduce offspring
- Best survival capability have the best chance to reproduce
- Crossover→combine part of parents to generate offspring
- Mutation→alter some of allele of the chromosome
- Survuval strength→ measured using fitness function→ objective function
- Each generation individual→ culling or survive
- Behavior→phenotype →influence evolutionary process in genetic change and evolve separately

Evolutionary Computation

- EC algo
  - Genrtic algorithm→model genetic evolutiona
  - Genetic programming→similar to GA but individual are program
  - Evolutionary programming→derived from the simulation of adaptive behavior in evolution (phenotype evolution)
  - Evolution strategies→model strategic parameters that control variation in evolution
  - Differential evolution→similar to GA except reproduction mechanism used
  - Cultural evolution → model the evolution of culture of a population and how the culture influences the genetic and phenotypic evolution of individual
  - Coevolution→ “dump” individual evolve through cooperation or in competition with one another, acquiring the necessary characteristics to survive

Swarm Intelligence

- Study of colonies or swarm of social organisma
- Study of social behavior of organisms (individuals) in swarm prompted the design f very efficient optimization and clustering algorithm
- Particle swarm optimization (PSO) →global optimization approach modeled on social behavior of bird flocks
  - A population based search procedure where the individuals (particles) are grouped into a swarm
  - Particle→individual→candidate solution to the optimization problem
    - Flown through the multidimensional search space→adjusting its position in search space according to its own experience and that of neighboring particles
    - Best position encounter by itself and that of its neighbor→position itself to the global minimum
    - Performance→measured according to a predefined fitness function which related to the problem being solved
- Study of ant colonies→modeling of pheromone depositing by ants in their search for the shortest paths to food sources resulted in the development of shortest path optimization algorithm

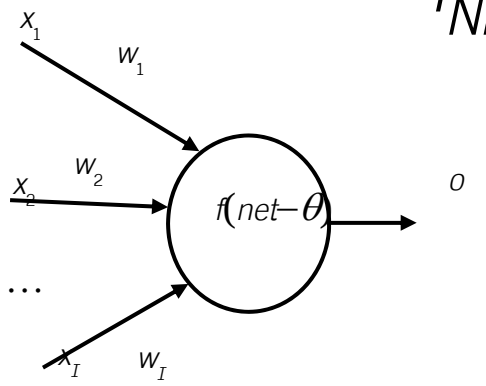
Sansanee Auephanwiriyakul, *Senior Member IEEE*

Introduction to Artificial Neural Networks

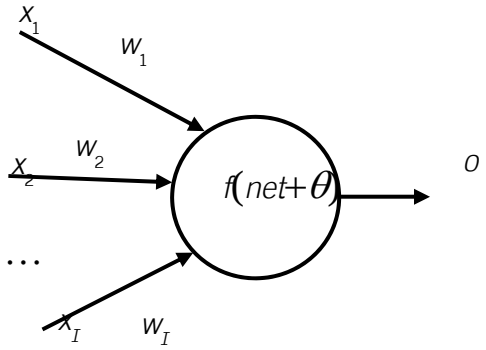
Neural Networks (NN)

- Mapping function form I input space to K output space

$$f_{NN} : \Re^I \rightarrow \Re^K$$



or



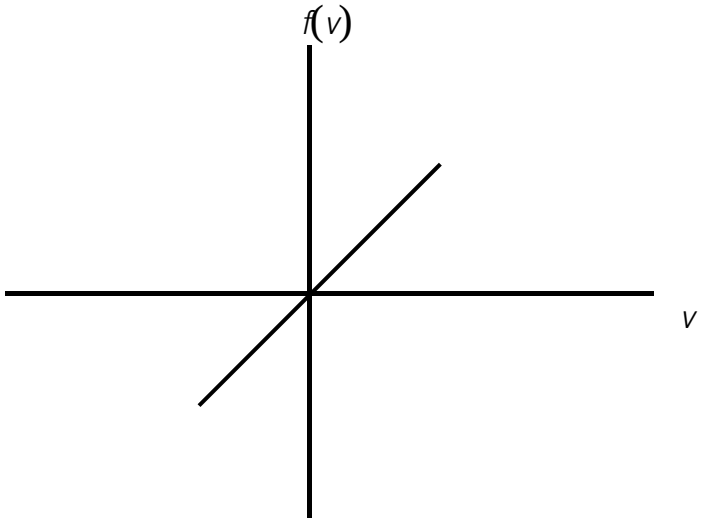
$$net = \sum_{i=1}^{I} x_i w_i \quad \text{or} \quad net = \prod_{i=1}^{I} x_i^{w_i}$$

and θ called bias or threshold

NN

- Activation function→ $f(-\infty)=0$ or $f(-\infty)=-1$ and $f(\infty)=1$
  - linear function

    $f(v)=\beta v$ where β is a constant

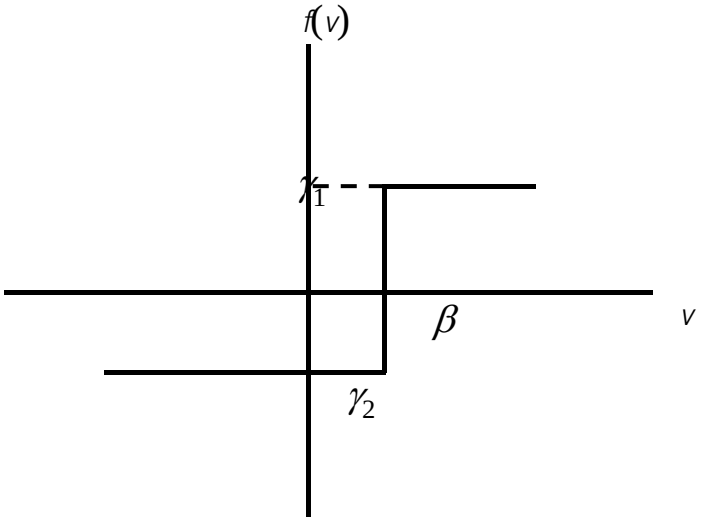


NN

- Step function or unit step function

$$f(v)=\begin{cases}\gamma_1 & \text{if } v \geq \beta \\ \gamma_2 & \text{if } v < \beta\end{cases}$$

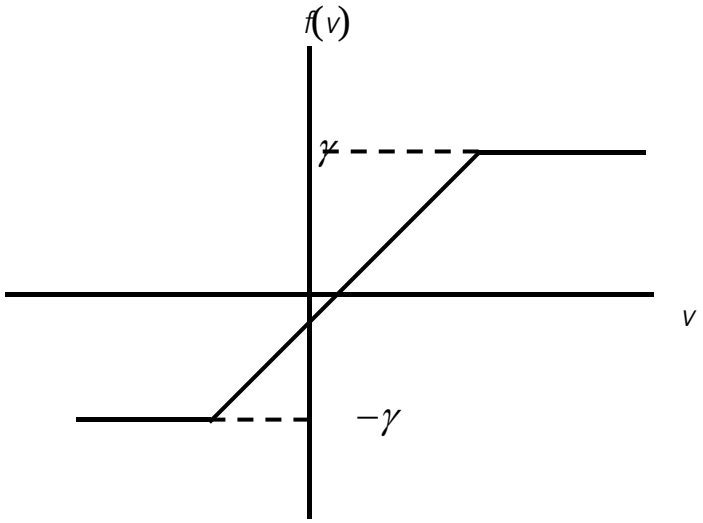
normally γ_1=1 and γ_2=0 or –1



NN

- Ramp function

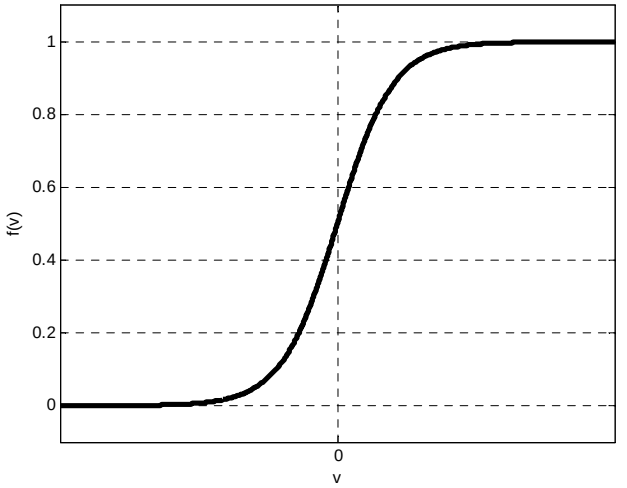
$$f(v) = \begin{cases} \gamma & \text{if } v \geq \beta \\ v & \text{if } -\beta < v < \beta \\ -\gamma & \text{if } v \leq -\beta \end{cases}$$



NN

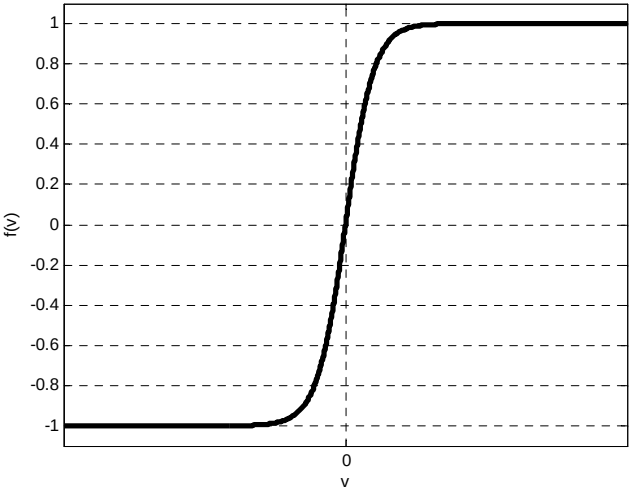
- Sigmoid function
  - Logistic function

$$f(v) = \frac{1}{1 + \exp(-av)}$$



  - Hyperbolic tangent function

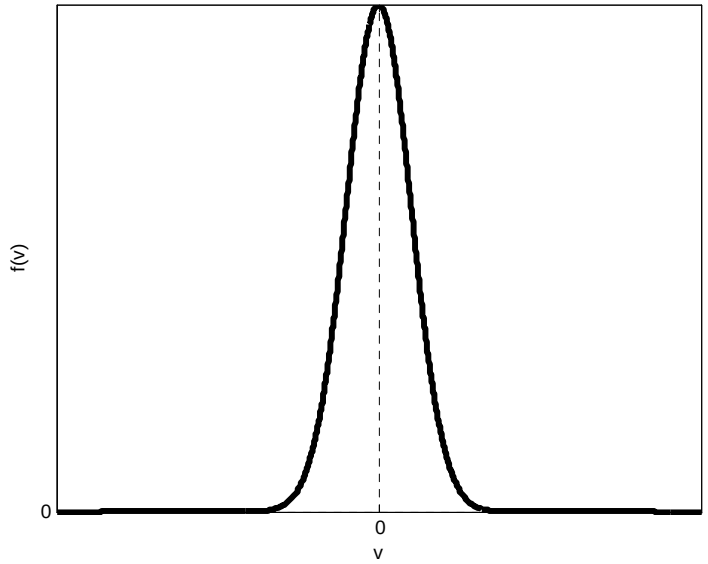
$$f(v) = \tanh\left(\frac{v}{2}\right) = \frac{1 - \exp(-v)}{1 + \exp(-v)} = \frac{2}{1 + \exp(-v)} - 1$$



NN

- Gaussian function

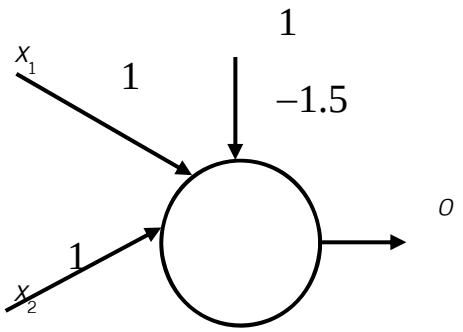
$$f(v) = \exp\left(-\frac{(v-\mu)^2}{\sigma^2}\right)$$ where $\mu \rightarrow$ mean and $\sigma \rightarrow$ standard deviation



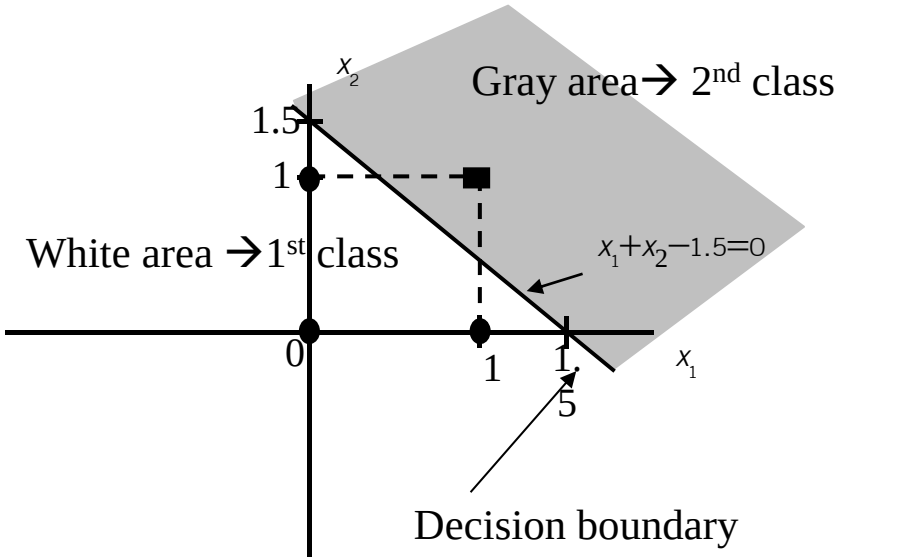
Multi-layer perceptrons

- Example: AND Logic → input vector $[x_1,x_2]^t$ with output y =0 are in the same class (1st class), one with output 1 are in another class (2nd class) → use unit step function with γ_1=1, γ_2=0 and β=0as activation function

x_1	x_2	Y (desired output)
0	0	**0**
0	1	**0**
1	0	**0**
1	**1**	**1**



$$v = x_1 + x_2 - 1.5$$

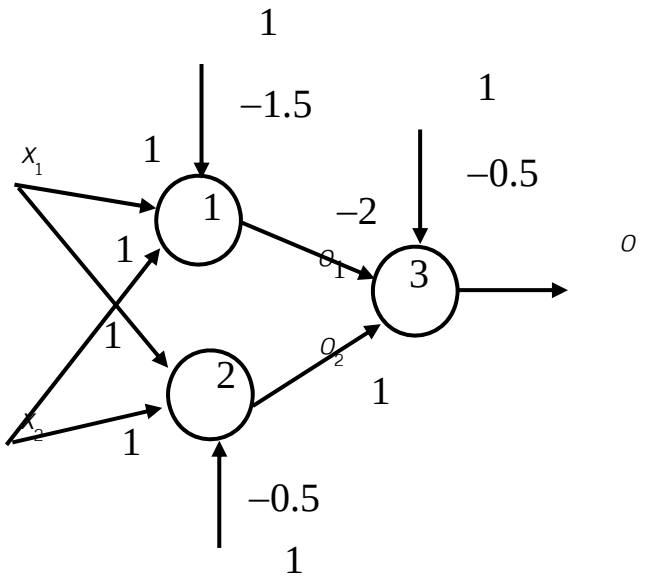


x_1	x_2	v	o (program output)
0	0	−1.5	**0**
0	1	−0.5	**0**
1	0	−0.5	**0**
1	**1**	**0.5**	**1**

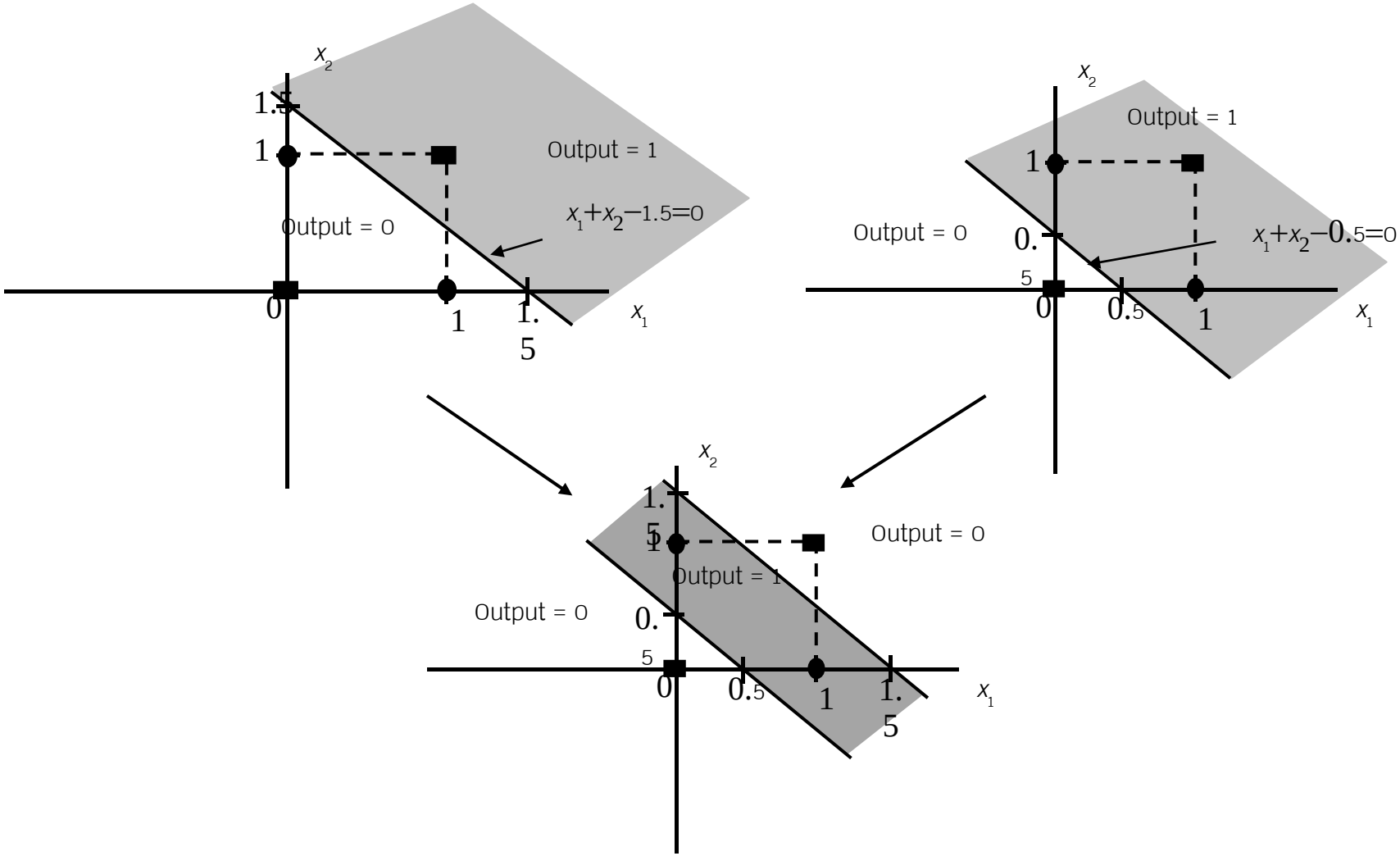
Multi-layer perceptrons

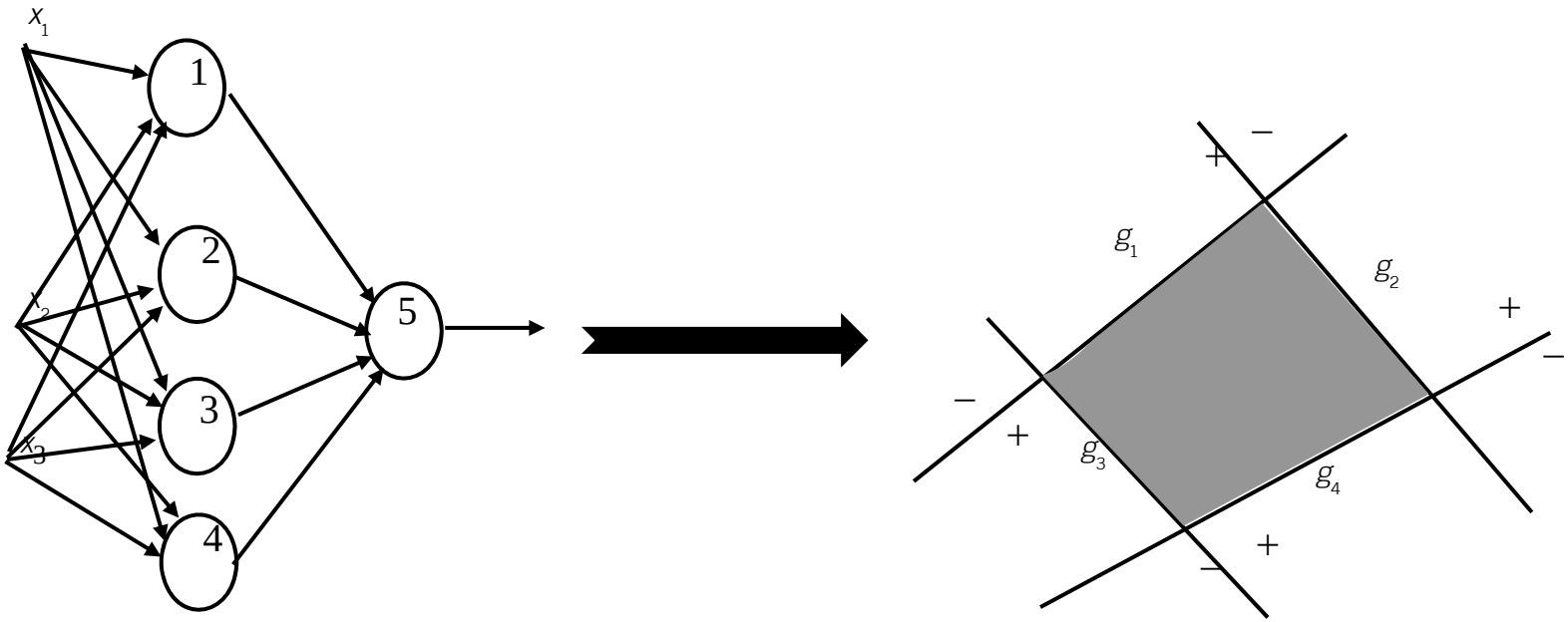
- Example : XOR Logic → input vector $[x_1,x_2]^t$ with output y =0 are in the same class (1st class), one with output 1 are in another class (2nd class) → use unit step function with γ_1=1, γ_2=0 and β=0 as activation function

x_1	x_2	Y (desired output)
0	0	**0**
0	1	**1**
1	0	**1**
1	**1**	**0**

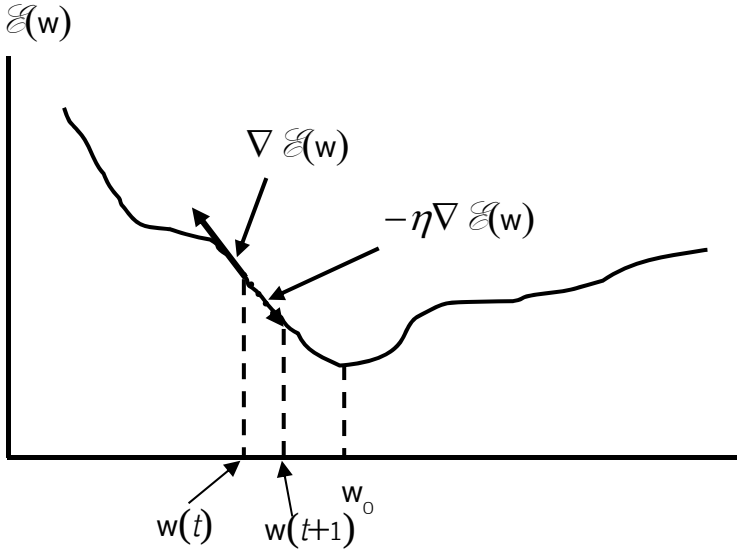


x_1	x_2	v_1	o_1	v_2	o_2	o (program output)
0	0	–1.5	0	–0.5	0	–0.5
0	1	–0.5	0	0.5	1	0.5
1	0	–0.5	0	0.5	1	0.5
1	**1**	0.5	1	1.5	1	–1.5





One hidden node create one decision boundary→ 4 hidden nodes creates 4 decision boundary→ these boundary is combined at the output node to create decision for the decision making



$$\mathscr{E}\left(w(t+1)\right) < \mathscr{E}\left(w(t)\right)$$

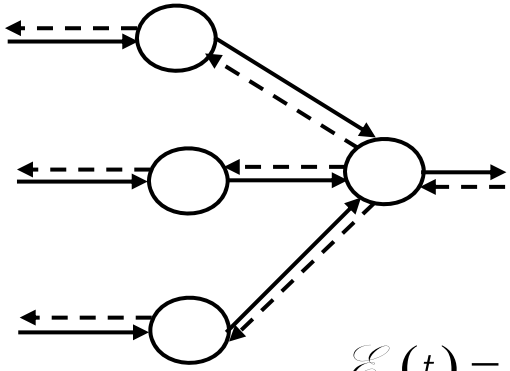
Gradient descent learning rule

Direction of gradient vector (($\nabla\mathscr{E}(\mathbf{w})$) is in the increasing direction→if we invert the direction we will have

$$\mathrm{w}(t+1) = \mathrm{w}(t) - \eta\nabla\mathscr{E}(w)$$

New weight that is in the decreasing direction of the error space

- if η is small, the transient response of the algorithm will be overdamped
- If η is large , the transient response of the algorithm will be underdamped
- If η is larger than the critical value, the algorithm will be unstable and may be diverged



$$\mathscr{E}(t) = \frac{1}{2}\sum_{j \in C} e_j^2(t)$$

$$\mathscr{E}_{av} = \frac{1}{N}\sum_{n=1}^{N} \mathscr{E}(n)$$

Backpropagation→ 2 passes;
Forward pass and Backward pass

Multilayer Perceptrons

- Initialization
- Forward Computation
  - Output from neuron j in layer l

$$y_j^{(l)}(n) = \varphi_j\left(v_j\left(n\right)\right) \quad \text{where} \quad v_j^{(l)}(n) = \sum_{i=0}^{m_{l-1}} w_{ji}^{(l)}(n) y_i^{(l-1)}(n)$$

where $y_i^{(l-1)}(n)$ is the output signal of neuron i in the previous layer l–1

and $w_{ji}^{(l)}(n)$ is the synaptic weight of neuron j in layer l that is fed from neuron i in layer l–1

if $l = 1$, $y_j^{(0)}(n) = x_j\left(n\right)$ and if $l = L$, $y_j^{(0)}(n) = o_j\left(n\right)$

compute the error: $e_j(n) = d_j(n) - o_j(n)$

- Backward computation
  - Local gradients

$$\delta_j^{(l)}(n) = \begin{cases} e_j^{(L)}(n)\varphi_j'\left(v_j^{(L)}(n)\right) & \text{for neuron } j \text{ in output layer } L \\ \varphi_j'\left(v_j^{(l)}(n)\right)\sum_k \delta_k^{(l+1)}(n) w_{kj}^{(l+1)}(n) & \text{for neuron } j \text{ in output layer } l \end{cases}$$

where $\varphi_j'(\cdot)$ denotes differentiation with respect to the argument

  - Update weights

$$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \alpha\left[\Delta w_{ji}^{(l)}(n-1)\right] + \eta\delta_j^{(l)}(n) y_j^{(l-1)}(n)$$

$$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \alpha\left[w_{ji}^{(l)}(n) - w_{ji}^{(l)}(n-1)\right] + \eta\delta_j^{(l)}(n) y_j^{(l-1)}(n)$$

- Iteration: until the stopping criterion is met

- For

$$y_j^{(l)}(t) = \varphi_j^{(l)}\left(v_j^{(l)}(t)\right) = \frac{1}{1+\exp\left(-v_j^{(l)}(t)\right)}$$

$$\frac{\partial y_j^{(l)}(t)}{\partial v_j^{(l)}(t)} = \varphi_j^{(l)\prime}\left(v_j^{(l)}(t)\right) = \frac{\exp\left(-v_j^{(l)}(t)\right)}{\left[1+\exp\left(-v_j^{(l)}(t)\right)\right]^2}$$

$$\varphi_j^{(l)\prime}\left(v_j^{(l)}(t)\right) = \frac{1}{1+\exp\left(-v_j^{(l)}(t)\right)}\left[1-\frac{1}{1+\exp\left(-v_j^{(l)}(t)\right)}\right]$$

$$\varphi_j^{(l)\prime}\left(v_j^{(l)}(t)\right) = y_j^{(l)}(t)\left[1-y_j^{(l)}(t)\right]$$

output layer $$\delta_j^{(L)}(t) = e_j^{(L)}(t)\left[o_j(t)\left[1-o_j(t)\right]\right]$$

hidden layer $$\delta_j^{(l)}(t) = y_j^{(l)}(t)\left[1-y_j^{(l)}(t)\right]\sum_k \delta_k^{(l+1)}(t) w_{kj}^{(l+1)}(t)$$

- For $$y_j^{(l)}(t)=\varphi_j^{(l)}\left(v_j^{(l)}(t)\right)=\tanh\left(\frac{v_j^{(l)}(t)}{2}\right)=\frac{2}{1+\exp\left(-v_j^{(l)}(t)\right)}-1$$

$$\frac{\partial y_j^{(l)}(t)}{\partial v_j^{(l)}(t)}=\varphi_j^{(l)\prime}\left(v_j^{(l)}(t)\right)=\frac{2\exp\left(-v_j^{(l)}(t)\right)}{\left[1+\exp\left(-v_j^{(l)}(t)\right)\right]^2}$$

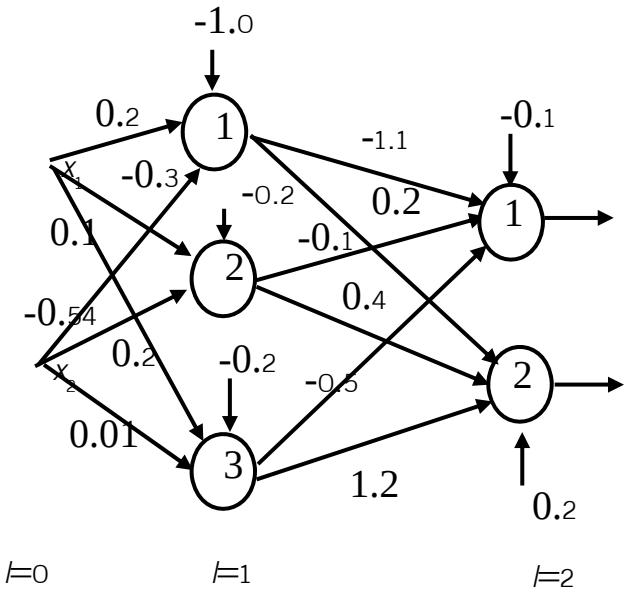
$$\varphi_j^{(l)\prime}\left(v_j^{(l)}(t)\right)=\frac{2}{1+\exp\left(-v_j^{(l)}(t)\right)}\left[1-\frac{1}{1+\exp\left(-v_j^{(l)}(t)\right)}\right]$$

$$\varphi_j^{(l)\prime}\left(v_j^{(l)}(t)\right)=2y_j^{(l)}(t)\left[1-y_j^{(l)}(t)\right]$$

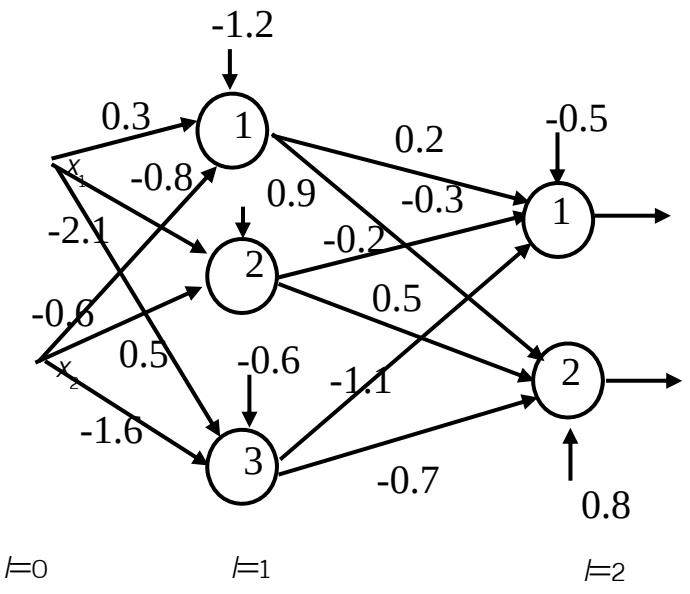
output layer $$\delta_j^{(L)}(t)=e_j^{(L)}(t)\left[2o_j(t)\left[1-o_j(t)\right]\right]$$

hidden layer $$\delta_j^{(l)}(t)=2y_j^{(l)}(t)\left[1-y_j^{(l)}(t)\right]\sum_k\delta_k^{(l+1)}(t)w_{kj}^{(l+1)}(t)$$

- Example: suppose at iteration t $x(t)=\begin{bmatrix}1\\ \\1\end{bmatrix}$ and $d(t)=\begin{bmatrix}0\\ \\1\end{bmatrix}$ and η=0.2, α =0.1. Find $w_{32}^{(1)}(t+1)$



$t-1$



t

$$v_1^{(1)}(t) = \sum_{i=0}^{2} w_{1i}^{(1)} x_i = -1.2 + (0.3)(1) + (-0.6)(1) = -1.5$$

$$y_1^{(1)}(t) = \frac{1}{1 + \exp\left(-v_1^{(1)}(t)\right)} = \frac{1}{1 + \exp(1.5)} = 0.1824$$

$$v_2^{(1)}(t) = \sum_{i=0}^{2} w_{2i}^{(1)} x_i = 0.9 + (-0.8)(1) + (0.5)(1) = 0.6$$

$$y_2^{(1)}(t) = \frac{1}{1 + \exp(-0.6)} = 0.6457$$

$$v_3^{(1)}(t) = \sum_{i=0}^{2} w_{3i}^{(1)} x_i = -0.6 + (0.5)(1) + (-1.6)(1) = -1.7$$

$$y_3^{(1)}(t) = \frac{1}{1 + \exp(1.7)} = 0.1545$$

$$v_1^{(2)}(t) = \sum_{i=0}^{3} w_{1i}^{(2)} y_i^{(1)} = -0.5 + (0.2)(0.1824) + (-0.2)(0.6457) + (-1.1)(0.1545) = -0.7626$$

$$y_1^{(2)}(t) = \frac{1}{1 + \exp(0.7626)} = 0.3181 \longrightarrow e_1^{(2)}(t) = 0 - 0.3181 = -0.3181$$

$$v_2^{(2)}(t) = \sum_{i=0}^{3} w_{1i}^{(2)} y_i^{(1)} = 0.8 + (-0.3)(0.1824) + (0.5)(0.6457) + (-0.7)(0.1545) = 0.96$$

$$y_2^{(2)}(t) = \frac{1}{1+\exp(-0.96)} = 0.7231 \longrightarrow e_2^{(2)}(t) = 1 - 0.7231 = 0.2769$$

$$\delta_1^{(2)}(t) = e_1^{(2)}(t)\left[o_1(t)\left[1-o_1(t)\right]\right] = (-0.3181)\left[0.3181\left(1-0.3181\right)\right] = -0.0690$$

$$\delta_2^{(2)}(t) = e_2^{(2)}(t)\left[o_2(t)\left[1-o_2(t)\right]\right] = (0.2769)\left[0.7231(1-0.7231)\right] = 0.0554$$

$$\delta_3^{(1)}(t) = y_3^{(1)}(t)\left[1-y_3^{(1)}(t)\right]\sum_{k=1}^{2}\delta_k^{(2)}(t)w_{kj}^{(2)}(t)$$

$$\delta_3^{(1)}(t) = 0.1545\left[1-0.1545\right]\left[(-0.0690)(-1.1)+(0.0554)(-0.7)\right] = 0.0048$$

$$\Delta w_{12}^{(2)}(t) = \alpha\Delta w_{12}^{(2)}(t-1) + \eta\delta_1^{(2)}(t)\,y_2^{(1)}(t)$$

$$\Delta w_{12}^{(2)}(t) = 0.1\left(-0.2-(-0.1)\right) + 0.2(-0.0690)(0.6457) = -0.0189$$

$$w_{12}^{(2)}(t+1) = w_{12}^{(2)}(t) + \Delta w_{12}^{(2)}(t) = -0.2 - 0.0189 = -0.2189$$

$$\Delta w_{32}^{(1)}(t) = \alpha\Delta w_{32}^{(1)}(t-1) + \eta\delta_3^{(1)}(t)\,x_2(t)$$

$$\Delta w_{32}^{(1)}(t) = 0.1(-1.6-0.01) + 0.2(0.0048)(1) = -0.16$$

$$w_{32}^{(1)}(t+1) = w_{32}^{(1)}(t) + \Delta w_{32}^{(1)}(t) = -1.6 - 0.16 = -1.76$$

- Generalization
  - Overtrain→look up table→do not want
  - Properly fit→want
  - Factor
    - Size of training data set
    - Architecture of NN
    - Physical complexity of the problem at hand
- Performance factor
  - Data preparation
    - Missing value→discard if have a lot of train data or replace the missing value with the average of that feature or with the most frequent
    - Coding input value
    - Outlier → discard that outlier if known and have a lot of train data if not, adjust the objective function so that it can cope with the outlier→robustness

- Scaling and normalization→ if some features dominate other features→ normalize each feature with its own mean and standard deviation so that every features are in the same range

  k feature of sample i → $\hat{x}_{ik} = \frac{x_{ik} - \overline{x}_k}{\sigma_k}$ where $\overline{x}_k = \frac{1}{N}\sum_{i=1}^{N} x_{ik}$ for $k = 1, 2, \ldots, p$

  and $\sigma_k^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(x_{ik} - \overline{x}_k\right)^2$

  - output value should be (0.1 or 0.9) or (-0.9 or 0.9) because if we use logistic function or hyperbolic tangent the computed valued will never goes to 0 or 1 (or -1 or 1)→ suppose there are 3 class: sample in class 1 will have the desire output be 0.9 0.1 0.1, that in class 2 will be 0.1 0.9 0.1 and that in class 3 will be 0.1 0.1 0.9→ if use tanh, it should be 0.9 -0.9 0.9 (class 1), -0.9 0.9 -0.9 (class 2) and -0.9 -0.9 0.9 (class 3)

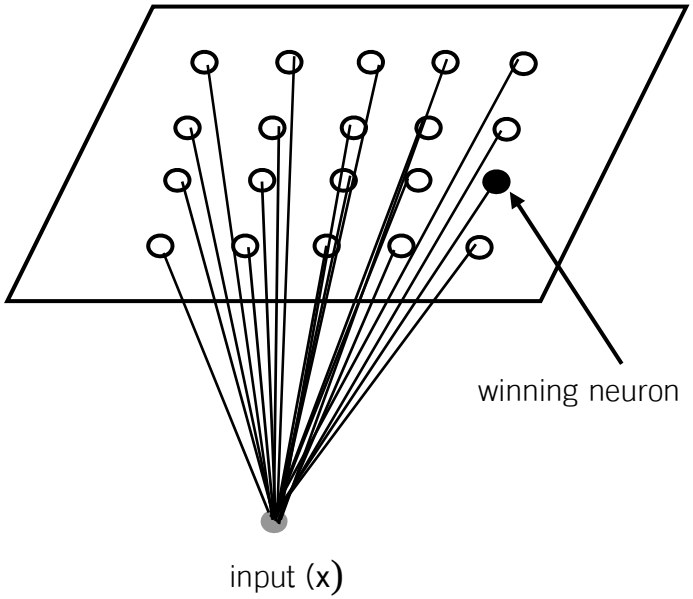
- Or
  - Noise injection→around decision boundary
  - Training set manipulation→ selective presentation→typical pattern and confusing pattern → need priori information

- Initialize weight with

$$\left[\frac{-1}{\sqrt{fanin}}, \frac{1}{\sqrt{fanin}}\right]$$

- Learning rate and momentum rate

Self organizing feature map (SOFM)



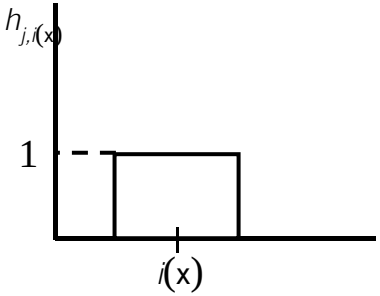
- Competitive

Input $\mathbf{x} = [x_1, x_2, \ldots, x_m]^t$ neuron j with $\mathbf{w}_j = [w_{j1}, w_{j2}, \ldots, w_{jm}]^t$

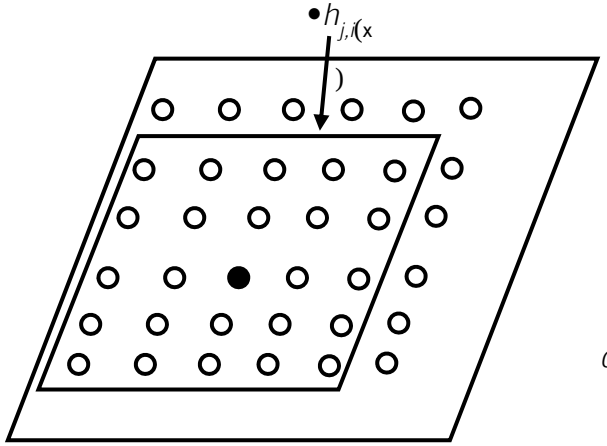
There are l neurons

Winning neuron $i(x) = \arg\min_j \|\mathbf{x} - \mathbf{w}_j\|$ for $j \in \mathscr{A}$

- Cooperation



1-d lattice



$d_{j,i}^2 = \|\mathbf{r}_j - \mathbf{r}_i\|^2$

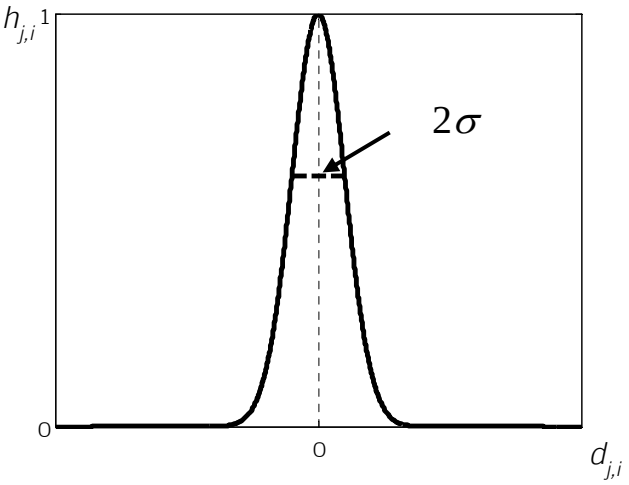
2-d lattice

- $h_{j,i(\mathbf{x})}$: symmetric about the maximum point defined by $d_{ji}=0$ (at winning neuron)
- Amplitude of $h_{j,i(\mathbf{x})}$ decrease monotonically with increasing lateral distance d_{ji} , decaying to 0 for $d_{ji}=\infty \rightarrow$ necessary condition for convergence

$$h_{j,i(x)} = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2}\right) \quad \text{for } j \in \mathscr{A}$$

$$\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_1}\right) \quad \text{for } n = 0,1,2,\ldots,$$

$$h_{j,i(x)}(n) = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2(n)}\right) \quad \text{for } n = 0,1,2,\ldots,$$



Where $d_{j,i} = |j - i|$ between neuron j and winning neuron i in case of 1-d lattice and $d_{j,i}^2 = \|r_j - r_i\|^2$ for 2-d lattice between r_i and r_j are vector of neuron i and j

- Synaptic adaptation→since update in 1 direction→might saturate, hence, include forgetting term ($g(y_j)$)

$$\Delta \mathbf{w}_j = \eta y_j \mathbf{x} - g\left(y_j\right)\mathbf{w}_j \quad \text{for } j \in \mathcal{A}$$

[ ] and $y_j = h_{j,i(\mathbf{x})}$

$$\Delta \mathbf{w}_j = \eta h_{j,i(\mathbf{x})}\left(\mathbf{x} - \mathbf{w}_j\right) \quad \text{for } j \in \mathcal{A}$$

If neighborhood function is rectangle function

$$\Delta \mathbf{w}_j = \begin{cases} \eta\left(\mathbf{x} - \mathbf{w}_j\right) & \text{if } j \text{ is inside neighborhood function} \\ 0 & \text{if } j \text{ is outside neighborhood function} \end{cases} \quad \text{for } j \in \mathcal{A}$$

$$\mathbf{w}_j(t+1) = \begin{cases} \mathbf{w}_j(t) + \eta(t)\left(\mathbf{x}(t) - \mathbf{w}_j(t)\right) & \text{if j is inside the neighborhood function} \\ \mathbf{w}_j(t) & \text{if j is not inside the neighborhood function} \end{cases}$$

If neighborhood function is guassian

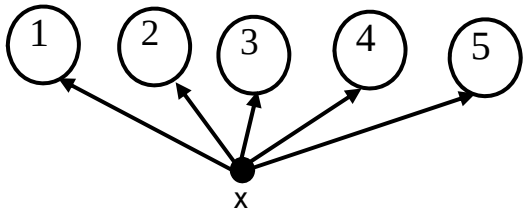
[ ]

$$\eta(n) = \eta_0 \exp\left(-\frac{n}{\tau_2}\right) \quad \text{for } n = 0,1,2,\ldots,$$

- 2 phase
  - Self-organizing or ordering phase
    - η might start from 0.1and decrease till the value is around 0.01 never goes to 0
    - Neighborhood function start from all neuron centered on the winning neuron and shrink slowly with time
  - Convergence phase
    - η maintained at a small value on the order of 0.01 do not decrease to 0
    - Neighborhood function contain only the nearest neighbor of a winning neuron eventually reduce to 1 or 0 neighbor

- Example

$$w_1 = \begin{bmatrix} 0.5 \\ 0.9 \end{bmatrix} \quad w_2 = \begin{bmatrix} -0.3 \\ -0.1 \end{bmatrix} \quad w_3 = \begin{bmatrix} -0.9 \\ 0.9 \end{bmatrix} \quad w_4 = \begin{bmatrix} 0.3 \\ -0.6 \end{bmatrix} \quad w_5 = \begin{bmatrix} 1.3 \\ -1.6 \end{bmatrix}$$



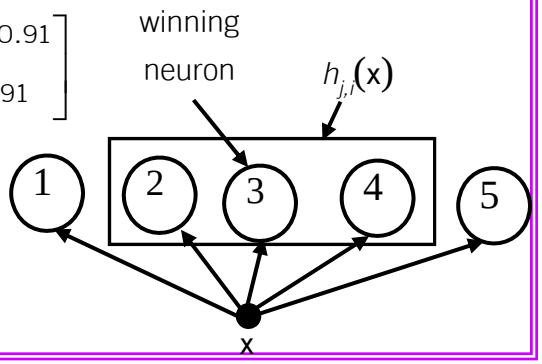
Input $x = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$ learning rate 0.1→ use $d = \sqrt{(x_1 - w_1)^2 + (x_2 - w_2)^2}$ and $h_{ji(\mathbf{x})}$ has the radius of 1

d(1) = 1.5033, d(2) =1.3038, d(3) = 0.1414, d(4) = 2.0616, d(5) = 3.4713

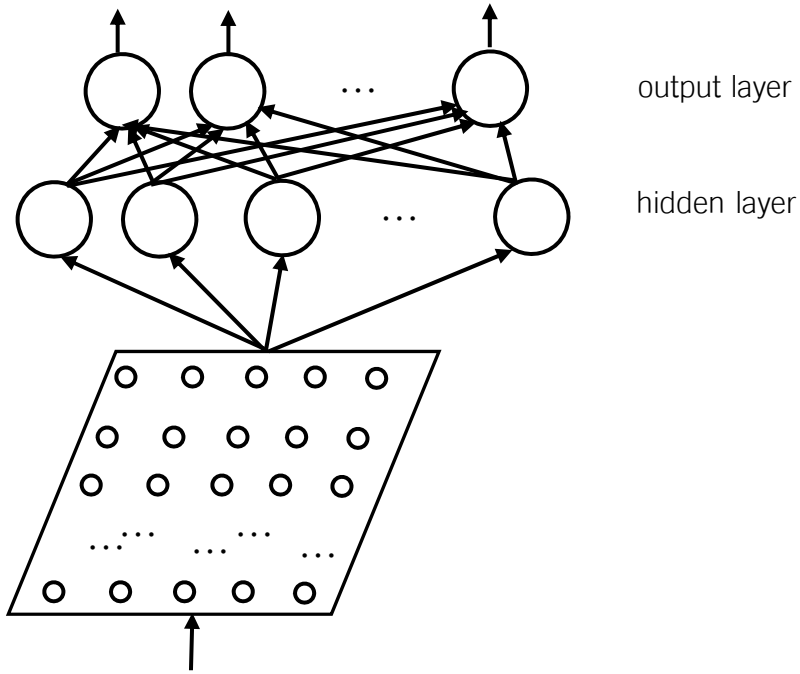
Winning neuron is neuron 3 ($i(\mathbf{x}) = 3$), neighbor with radius of 1 will be node 2 and 4 only, hence, update weight at nodes 2, 3,and 4

$$w_2 = \begin{bmatrix} -0.3 \\ -0.1 \end{bmatrix} + 0.1\left(\begin{bmatrix} -1 \\ 1 \end{bmatrix} - \begin{bmatrix} -0.3 \\ -0.1 \end{bmatrix}\right) = \begin{bmatrix} -0.37 \\ 0.01 \end{bmatrix} \quad w_3 = \begin{bmatrix} -0.9 \\ 0.9 \end{bmatrix} + 0.1\left(\begin{bmatrix} -1 \\ 1 \end{bmatrix} - \begin{bmatrix} -0.9 \\ 0.9 \end{bmatrix}\right) = \begin{bmatrix} -0.91 \\ 0.91 \end{bmatrix}$$

$$w_4 = \begin{bmatrix} 0.3 \\ -0.6 \end{bmatrix} + 0.1\left(\begin{bmatrix} -1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0.3 \\ -0.6 \end{bmatrix}\right) = \begin{bmatrix} 0.17 \\ -0.44 \end{bmatrix} \quad w_1 = \begin{bmatrix} 0.5 \\ 0.9 \end{bmatrix} \quad w_5 = \begin{bmatrix} 1.3 \\ -1.6 \end{bmatrix}$$



- SOFM properties
  - Approximation of the input space
  - Topological ordering
  - Density matching
  - Feature selection



Sansanee Auephanwiriyakul, *Senior Member IEEE*

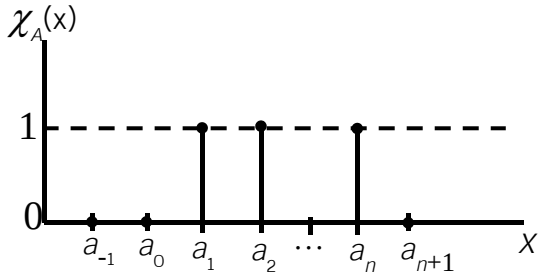
Introduction to Fuzzy Systems

Fuzzy systems

- Uncertainty and complexity
  - Driving in the unfamiliar condition
- Natural language
  - Cold: people in the north and in the south know what cold is but when it happens people in these two area will say differently→ people in the north might say that 10 degree celsius is cold but people in the south might say that 20 degree celsius is cold
- Heap paradox

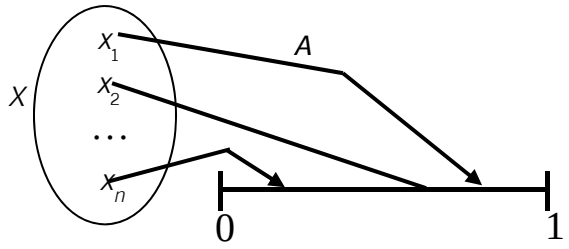
Fuzzy Set Foundation

- Characteristic function→crisp set

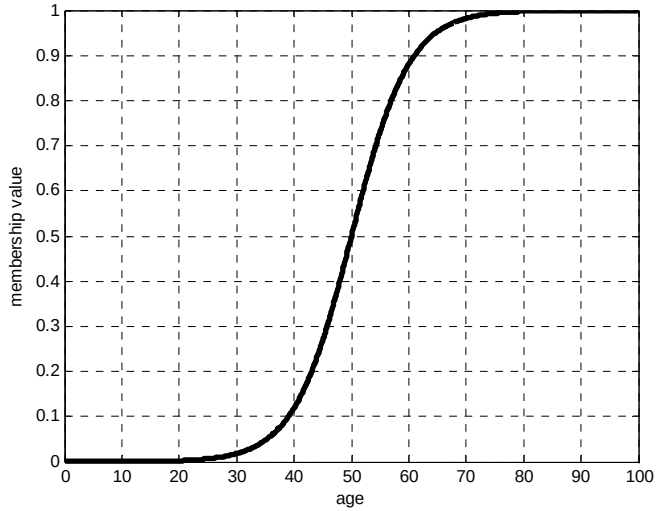


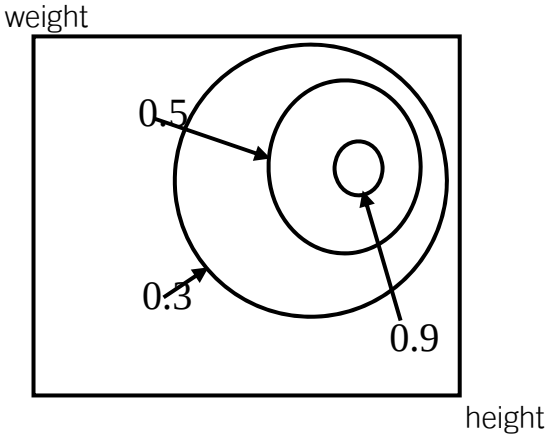
- Membership function →fuzzy set

  $\boldsymbol{A}: X \rightarrow [0,1]$ or $\mu_{\boldsymbol{A}}: X \rightarrow [0,1]$



Membership function of "old"

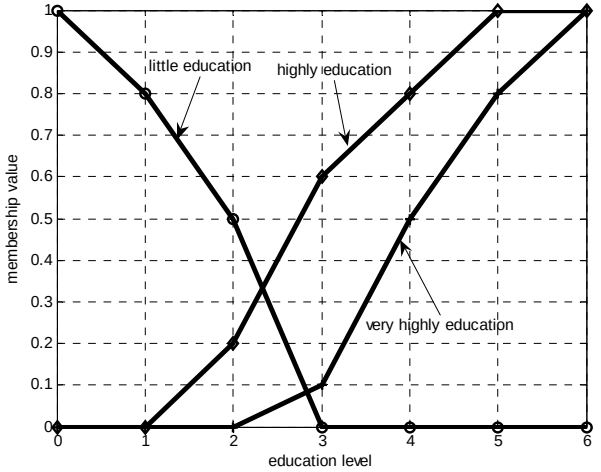




Membership function of "big": Countour diagram

No.	Education level
0	No education
1	Elementary school
2	High school
3	2-year college
4	Bachelor's degree
5	Master's degree
6	Doctoral degree

Universal set



3 fuzzy sets ("little", "high" and "very high" education)

How to write membership function

name (symbol)	Membership value in fuzzy set A
Carry (x_1)	0.8
Bill (x_2)	0.3
J-H (x_3)	0.5
Wabei (x_4)	0.9

$$\longrightarrow \quad A = \frac{0.8}{\text{Carry}} + \frac{0.3}{\text{Bill}} + \frac{0.5}{\text{J-H}} + \frac{0.9}{\text{Wabei}}$$

How to write membership function as an equation

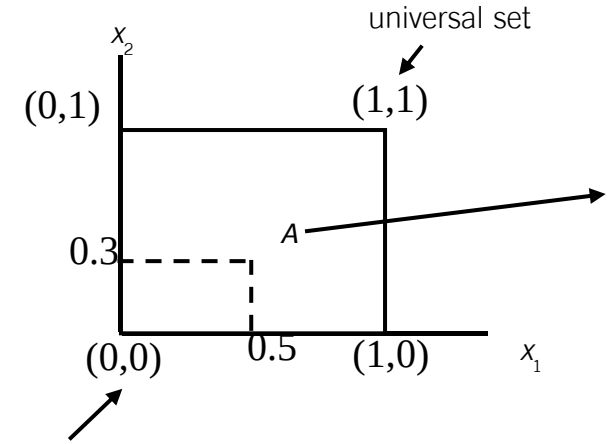
$$A = \sum_{x} \frac{A(x)}{x}$$

discrete

$$A = \int_{x} \frac{A(x)}{x}$$

continuous

Geometric representation



Fuzzy empty set

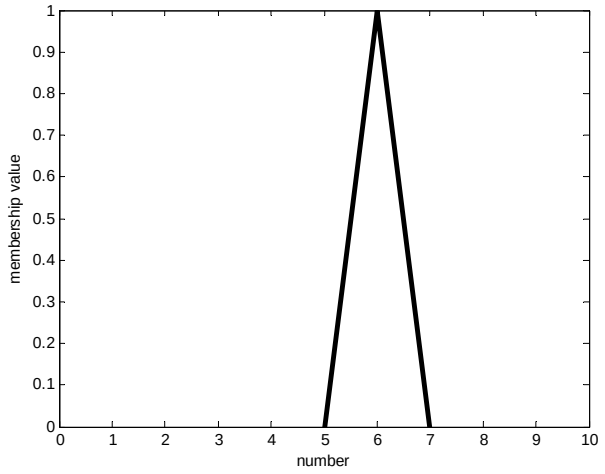
$$A = \frac{0.5}{x_1} + \frac{0.3}{x_2}$$

Every point in this area are fuzzy set→including 4 corner points

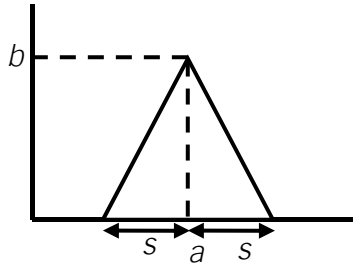
Analytic representation : function

example

$$A(x) = \begin{cases} x-5 & 5 \le x \le 6 \\ 7-x & 6 \le x \le 7 \\ 0 & \text{else} \end{cases}$$

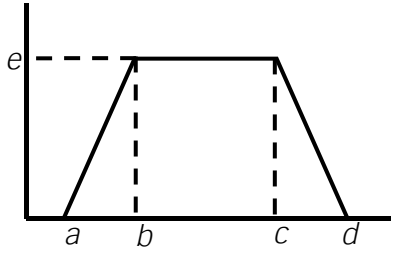


Analytic representation



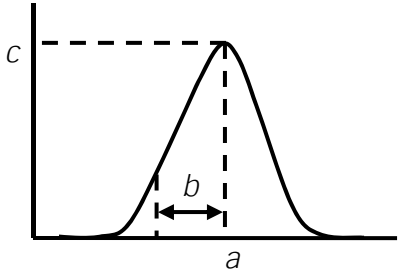
$$A(x) = \begin{cases} b\left(1 - \dfrac{|x-a|}{s}\right) & a-s \le x \le a+s \\ 0 & \text{else} \end{cases}$$

Triangular shape



$$A(x) = \begin{cases} \dfrac{(a-x)e}{a-b} & a \le x \le b \\ e & b \le x \le c \\ \dfrac{(d-x)e}{d-c} & c \le x \le d \\ 0 & \text{else} \end{cases}$$

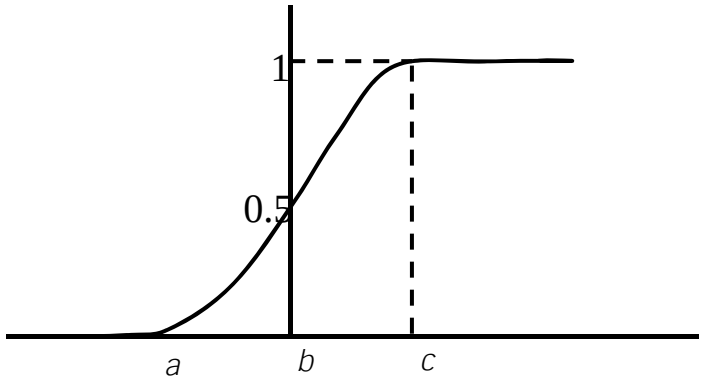
Trapezoidal shape



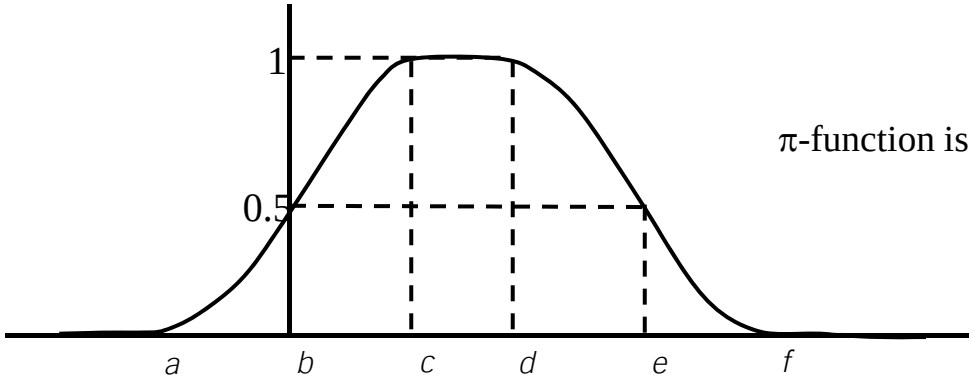
$$A(x) = ce^{\frac{-(x-a)^2}{b}}$$

Bell-shape

S-function



$$s(x)=\begin{cases}0 & 0\le x\le a\\ \frac{1}{2}\left(\frac{x-a}{b-a}\right)^2 & a\le x\le b\\ 1-\frac{1}{2}\left(\frac{x-c}{c-b}\right)^2 & b\le x\le c\end{cases}$$

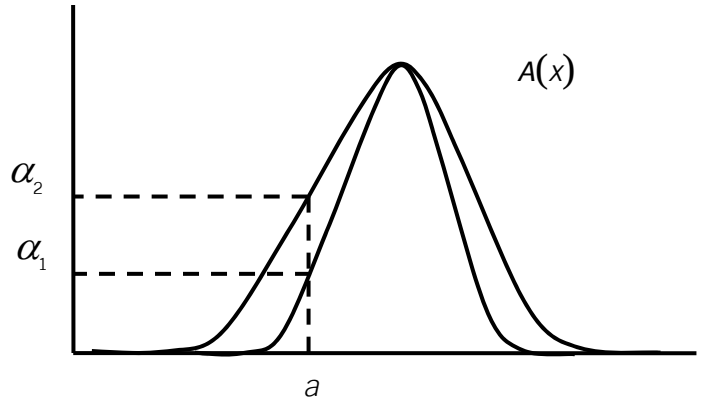


π-function is from S-function + reflected of S-function

Interval-valued fuzzy set

$$\boldsymbol{A} : X \rightarrow \mathcal{E}([0,1])$$

$\mathcal{E}([0,1])$ denote the family of all closed interval of real number in [0,1] ($\mathcal{E}([0,1]) \subset \mathrm{P}([0,1])$)

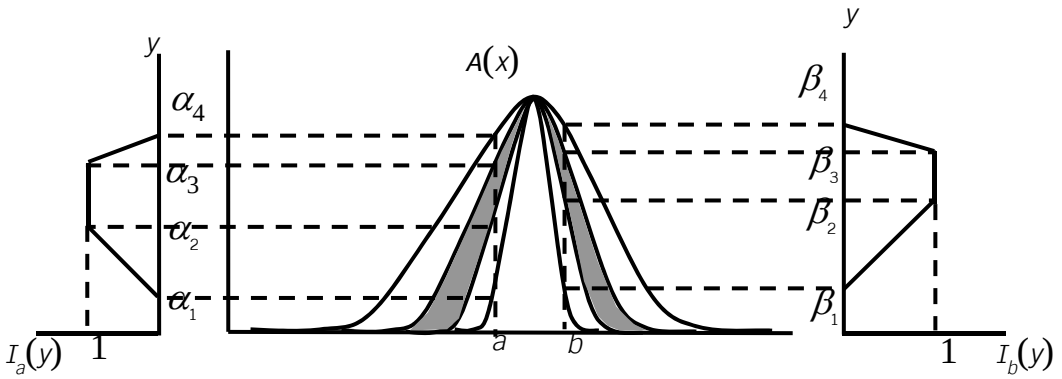


$$\boldsymbol{A}(a) = [\alpha_1, \alpha_2]$$

Type-2 fuzzy set

$$\boldsymbol{A} : X \rightarrow \tilde{\mathcal{P}}([0,1])$$

$\tilde{\mathcal{P}}([0,1])$ denote the set of all ordinary fuzzy sets that can be defined within the universal [0,1]



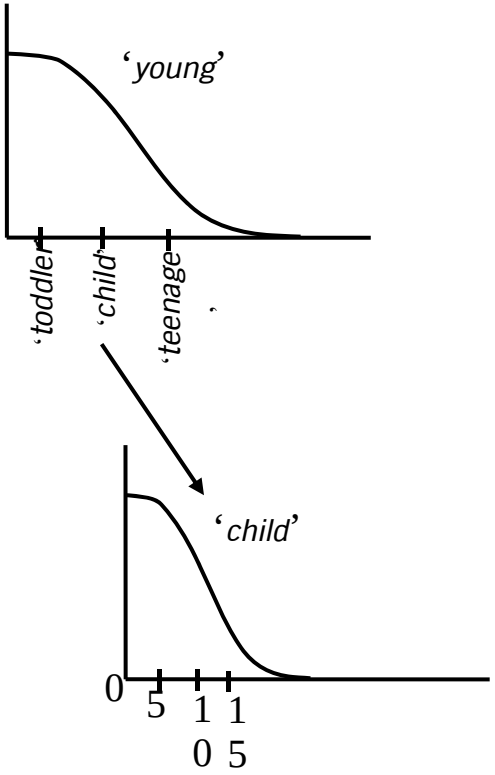
$\boldsymbol{A}(a)$ = fuzzy set (α_1, α_2, α_3, α_4)

$\boldsymbol{A}(b)$ = fuzzy set (β_1, β_2, β_3, β_4)

Level 2 fuzzy set

$$A : \tilde{\mathcal{P}}(X) \rightarrow ([0,1])$$

$\tilde{\mathcal{P}}(X)$ denote fuzzy power set of **X**

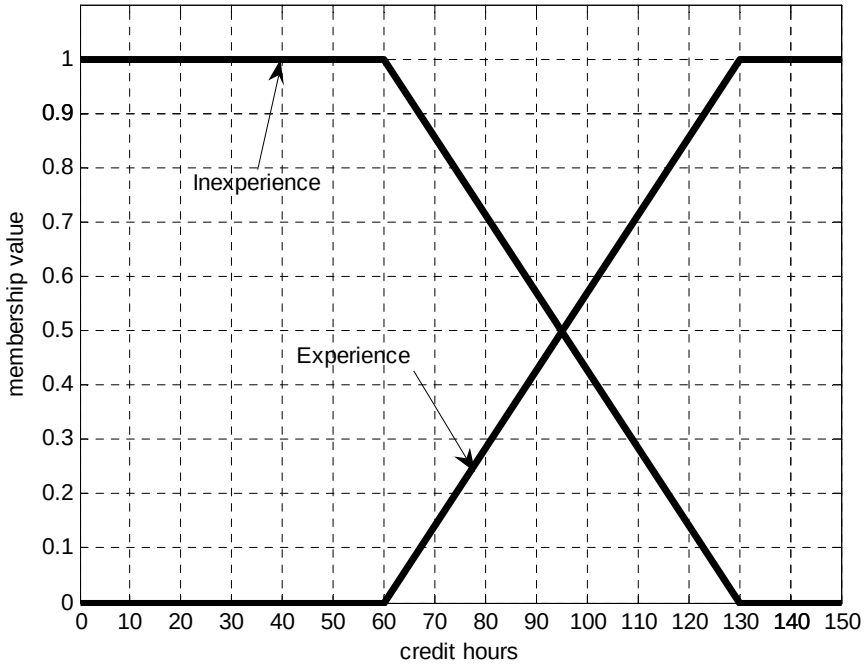


Standard fuzzy complement

$\boldsymbol{A}(x)$ express degree to which x belong to fuzzy set $\boldsymbol{A}$

$\overline{A}(x)$ express degree to which x does not belong to fuzzy set $\boldsymbol{A}$

$$\overline{A}(x) = 1 - A(x) \quad \forall x \in X$$



Standard fuzzy union

$$(\boldsymbol{A}\cup\boldsymbol{B})(x) = \max[\boldsymbol{A}(x), \boldsymbol{B}(x)]$$

patient	high blood pressure($\boldsymbol{A}$)	High fever ($\boldsymbol{B}$)	High blood pressure or high fever ($\boldsymbol{A}\cup\boldsymbol{B}$)
1	1	1	1
2	0.5	0.6	0.6
3	1	0.1	1
⋮	⋮	⋮	⋮
n	0.1	0.7	0.7

Do not satisfy the law of excluded middle

$(A \cup \overline{A})(x)$ will not equal to universal set

Standard fuzzy intersection

$$(\boldsymbol{A}\cap\boldsymbol{B})(x) = \min[\boldsymbol{A}(x), \boldsymbol{B}(x)]$$

river	Long river (***A***)	Nevigable (***B***)	Long and nevigable (***A***$\cap$***B***)
Amazon	1	0.8	0.8
Nile	0.9	0.7	0.7
Yang-Tsi	0.8	0.8	0.8
Danube	0.5	0.6	0.5
Rhine	0.4	0.3	0.3

Do not satisfy the law of contradiction

$(A \cap \overline{A})(x)$ will not equal to fuzzy empty set

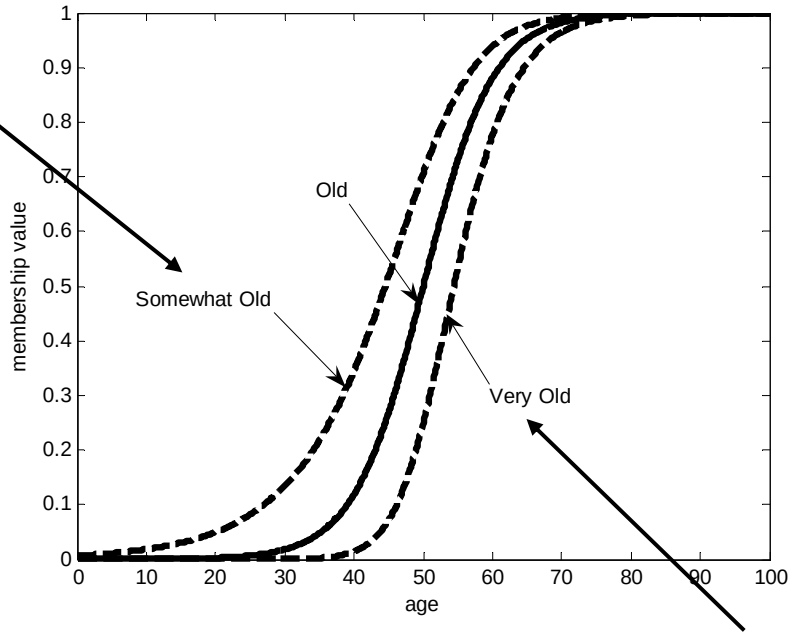
Inclusion: $\boldsymbol{A} \subseteq \boldsymbol{B}$ if $\boldsymbol{A}(x) \leq \boldsymbol{B}(x)\ \forall x \in \mathbf{X}$

Equality: $\boldsymbol{A} = \boldsymbol{B}$ if $\boldsymbol{A}(x) = \boldsymbol{B}(x)\ \forall x \in \mathbf{X}$

Unary operation: $\boldsymbol{A}^a(x) = (\boldsymbol{A}(x))^a$

if $a>1$ → more specific, if $a<1$ → less specific

somewhat old are from old$^{0.5}$



Very old are from old^2

Support of fuzzy set $\boldsymbol{A}$ (supp($\boldsymbol{A}$)):

$$\text{supp}(\boldsymbol{A}) = \{x \in X \mid \boldsymbol{A}(x) > 0\}$$

Height of fuzzy set $\boldsymbol{A}$ (h($\boldsymbol{A}$)) : $h(A) = \sup_x A(x)$

Largest value of membership value obtained by any element in that set that is >0

If h($\boldsymbol{A}$)=1 → normal fuzzy set
If h($\boldsymbol{A}$)<1 → supnormal fuzzy set
If h($\boldsymbol{A}$)>1 → supernormal fuzzy set

Core of fuzzy set $\boldsymbol{A}$ (core($\boldsymbol{A}$)) :

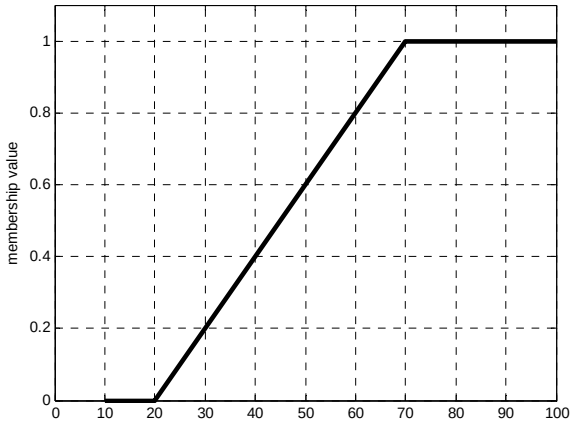
core($\boldsymbol{A}$)=$\{x \in X \mid \boldsymbol{A}(x) \geq h(\boldsymbol{A})\}$ since height is h($\boldsymbol{A}$) then we can have
core($\boldsymbol{A}$)=$\{x \in X \mid \boldsymbol{A}(x) = h(\boldsymbol{A})\}$

α-cut of fuzzy set $\boldsymbol{A}$ ($^{\alpha}\boldsymbol{A}$): $^{\alpha}\boldsymbol{A} = \{x \in X \mid \boldsymbol{A}(x) \geq \alpha\}$

Strong α-cut of fuzzy set $\boldsymbol{A}$ ($^{\alpha+}\boldsymbol{A}$): $^{\alpha+}\boldsymbol{A} = \{x \in X \mid \boldsymbol{A}(x) > \alpha\}$

If $\alpha_1 < \alpha_2$ then $^{\alpha 1}\boldsymbol{A} \supseteq {}^{\alpha 2}\boldsymbol{A}$
and $^{\alpha 1}\boldsymbol{A} \cap {}^{\alpha 2}\boldsymbol{A} = {}^{\alpha 2}\boldsymbol{A}$, $^{\alpha 1}\boldsymbol{A} \cup {}^{\alpha 2}\boldsymbol{A} = {}^{\alpha 1}\boldsymbol{A}$

If $\alpha_1 < \alpha_2$ then $^{\alpha 1+}\boldsymbol{A} \supseteq {}^{\alpha 2+}\boldsymbol{A}$
and $^{\alpha 1+}\boldsymbol{A} \cap {}^{\alpha 2+}\boldsymbol{A} = {}^{\alpha 2+}\boldsymbol{A}$, $^{\alpha 1+}\boldsymbol{A} \cup {}^{\alpha 2+}\boldsymbol{A} = {}^{\alpha 1+}\boldsymbol{A}$



$^{0}\boldsymbol{E} = [0,100]$ or $^{0.2}\boldsymbol{E} = [30,100]$ or $^{1}\boldsymbol{E} = [70,100]$ (core($\boldsymbol{E}$))

$^{0+}\boldsymbol{E} = (20,100]$ (supp($\boldsymbol{E}$)) or $^{0.2+}\boldsymbol{E} = (30,100]$ or $^{1+}\boldsymbol{E} = \varnothing$

Level set of fuzzy set $\boldsymbol{A}$ ($L_{\boldsymbol{A}}$ or $\wedge_{\boldsymbol{A}}$): $L_{\boldsymbol{A}} = \wedge_{\boldsymbol{A}} = \{\alpha \mid \boldsymbol{A}(x)=\alpha; \exists x \in X\}$

Special fuzzy set from α-cut (${}_{\alpha}\boldsymbol{A}$)

$$ {}_{\alpha}\boldsymbol{A}(x) = \alpha({}^{\alpha}\boldsymbol{A}(x)) \qquad \forall x \in \mathbf{X} $$

Example

$\boldsymbol{A} = 0.2/x_1 + 0.4/x_2 + 0.6/x_3 + 0.8/x_4 + 1/x_5$

We have $L_{\boldsymbol{A}} = \{0.2, 0.4, 0.6, 0.8, 1\}$

α-cut of $\boldsymbol{A}$ will be

${}^{0.2}\boldsymbol{A} = \{x_1, x_2, x_3, x_4, x_5\} = 1/x_1 + 1/x_2 + 1/x_3 + 1/x_4 + 1/x_5$

${}^{0.4}\boldsymbol{A} = 0/x_1 + 1/x_2 + 1/x_3 + 1/x_4 + 1/x_5$

${}^{0.6}\boldsymbol{A} = 0/x_1 + 0/x_2 + 1/x_3 + 1/x_4 + 1/x_5$

${}^{0.8}\boldsymbol{A} = 0/x_1 + 0/x_2 + 0/x_3 + 1/x_4 + 1/x_5$

${}^{1}\boldsymbol{A} = 0/x_1 + 0/x_2 + 0/x_3 + 0/x_4 + 1/x_5$

Special fuzzy set will be

${}_{0.2}\boldsymbol{A} = 0.2/x_1 + 0.2/x_2 + 0.2/x_3 + 0.2/x_4 + 0.2/x_5$

${}_{0.4}\boldsymbol{A} = 0/x_1 + 0.4/x_2 + 0.4/x_3 + 0.4/x_4 + 0.4/x_5$

${}_{0.6}\boldsymbol{A} = 0/x_1 + 0/x_2 + 0.6/x_3 + 0.6/x_4 + 0.6/x_5$

${}_{0.8}\boldsymbol{A} = 0/x_1 + 0/x_2 + 0/x_3 + 0.8/x_4 + 0.8/x_5$

${}_{1}\boldsymbol{A} = 0/x_1 + 0/x_2 + 0/x_3 + 0/x_4 + 1/x_5$

Decomposition theorem

From special fuzzy set in the previous example

$${}_{0.2}\boldsymbol{A} = 0.2/\ x_1 + 0.2/\ x_2 + 0.2/\ x_3 + 0.2/\ x_4 + 0.2/\ x_5$$
$${}_{0.4}\boldsymbol{A} = 0/\ x_1 + 0.4/\ x_2 + 0.4/\ x_3 + 0.4/\ x_4 + 0.4/\ x_5$$
$${}_{0.6}\boldsymbol{A} = 0/\ x_1 + 0/\ x_2 + 0.6/\ x_3 + 0.6/\ x_4 + 0.6/\ x_5$$
$${}_{0.8}\boldsymbol{A} = 0/\ x_1 + 0/\ x_2 + 0/\ x_3 + 0.8/\ x_4 + 0.8/\ x_5$$
$${}_{1}\boldsymbol{A} = 0/\ x_1 + 0/\ x_2 + 0/\ x_3 + 0/\ x_4 + 1/\ x_5$$

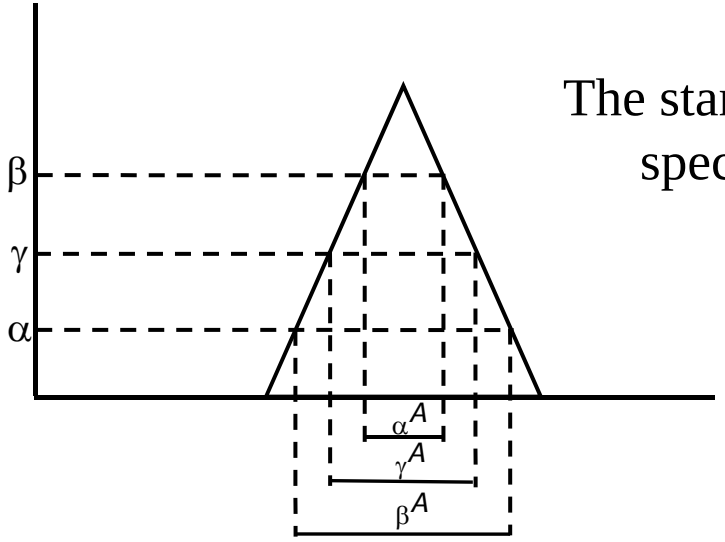
We can see that $({}_{0.2}\boldsymbol{A} \cup {}_{0.4}\boldsymbol{A} \cup {}_{0.6}\boldsymbol{A} \cup {}_{0.8}\boldsymbol{A} \cup {}_{1}\boldsymbol{A}) \rightarrow \boldsymbol{A}$

First theorem:

For and $\boldsymbol{A} \in \tilde{\mathcal{P}}(X)$,

$$A = \bigcup_{\alpha \in [0,1]} {}^{\alpha}A$$

where ${}_{\alpha}\boldsymbol{A}(x) = \alpha({}^{\alpha}\boldsymbol{A}(x))$, $\forall x \in \mathbf{X}$ and $\cup$ is a standard fuzzy union



The standard fuzzy union of 3 special fuzzy sets → ***A***

Second theorem:

For and $\boldsymbol{A} \in \tilde{\mathcal{P}}(x)$, $A = \bigcup_{\alpha \in [0,1]} {}_{\alpha+}A$

where ${}_{\alpha+}\boldsymbol{A}(x) = \alpha({}^{\alpha+}\boldsymbol{A}(x))$, $\forall x \in \mathbf{X}$ and $\cup$ is a standard fuzzy union

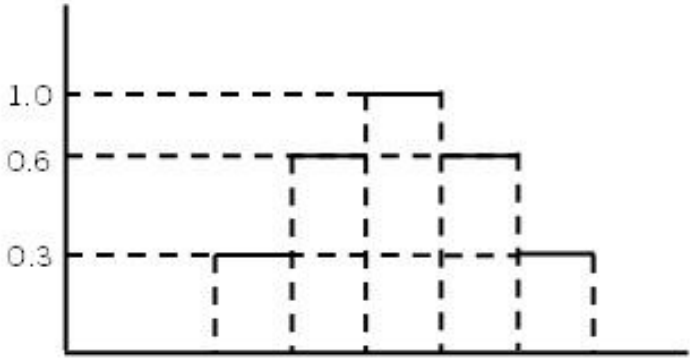
Third theorem:

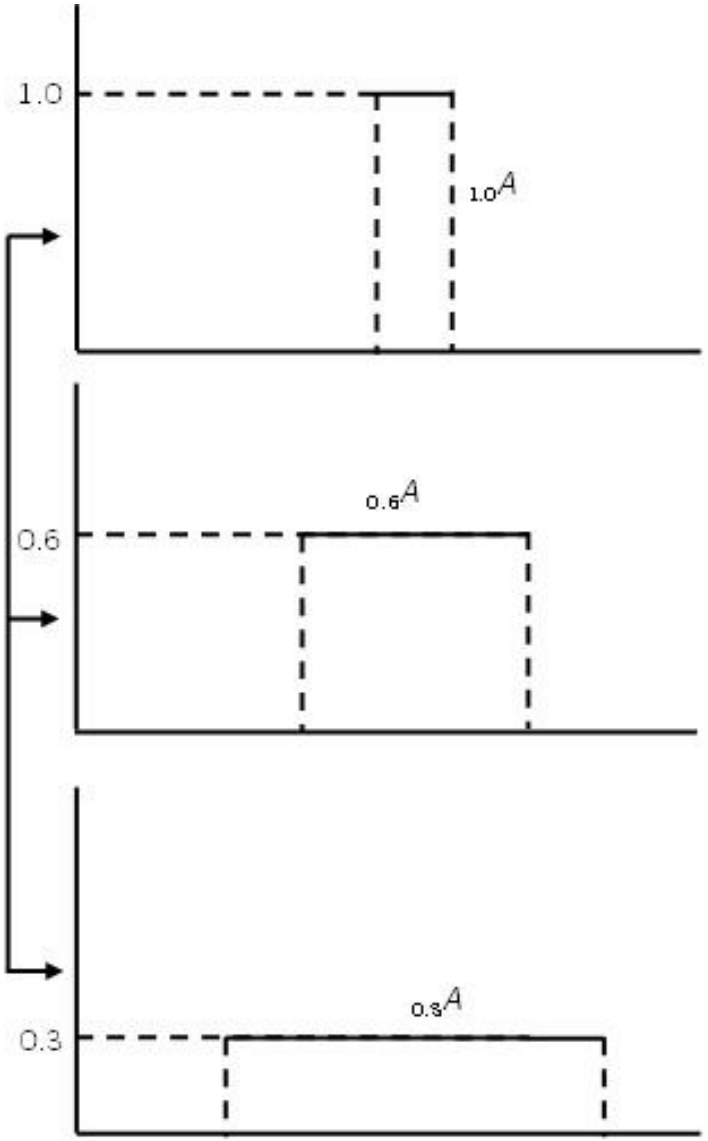
For and $\boldsymbol{A} \in \tilde{\mathcal{P}}(x)$,

$$A = \bigcup_{\alpha \in \wedge_A} {}_{\alpha}A$$

where ${}_{\alpha}\boldsymbol{A}(x) = \alpha({}^{\alpha}\boldsymbol{A}(x))$, $\forall x \in \mathbf{X}$ and $\cup$ is a standard fuzzy union

This example shows that for discrete fuzzy set only 1st and 3rd theorem will work fine





$\wedge_A = \{0, 0.3, 0.6, 1\}$ and $_0\boldsymbol{A} = \varnothing$ therefore $\boldsymbol{A}$ is a union of $_{0.3}\boldsymbol{A}$ $_{0.6}\boldsymbol{A}$ and $_1\boldsymbol{A}$

Convex fuzzy set

A is a convex fuzzy set if

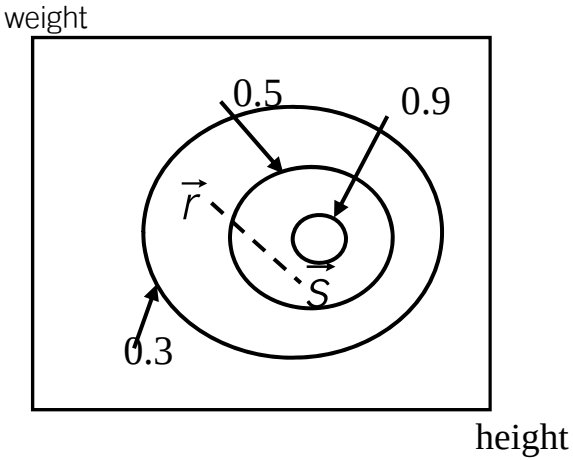
$$A\left(\lambda\vec{r}+(1-\lambda)\vec{s}\right)\geq \min\left(A(\vec{r}),A(\vec{s})\right) \quad \text{where } 0\leq\lambda\leq 1$$

Meaning that membership values of all the points on the line connecting $\vec{r}$ and $\vec{s}$ is bigger than or equal to the minimum membership values of $\vec{r}$ and $\vec{s}$

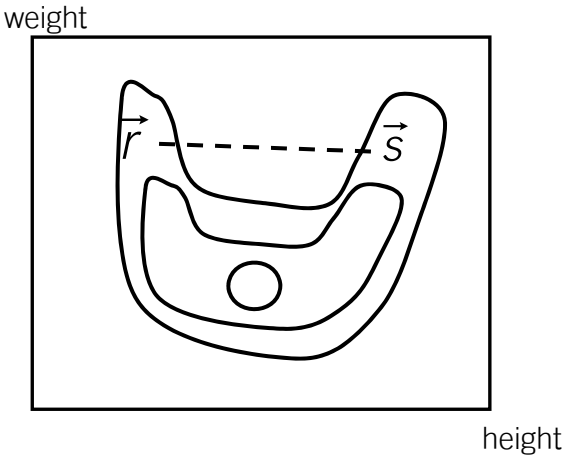
Or

If ***A*** is a convex fuzzy set then (1) α-cut for all α are not disconnected, and (2)each α-cut is convex in crisp sense

Convex fuzzy set



All the points of the dashed line have a membership values bigger than or equal to the minimum membership value of vector *r* and vector *s*



Not convex fuzzy set

Fuzzy Inference System

- Hedge
  - Dilation: $\boldsymbol{HA}(x) = (\boldsymbol{A}(x))^{1/2}$ → more or less of $\boldsymbol{A}$
  - Concentration: $\boldsymbol{HA}(x) = (\boldsymbol{A}(x))^{2}$ → very $\boldsymbol{A}$
  - Plus: $\boldsymbol{HA}(x) = (\boldsymbol{A}(x))^{1.25}$
  - Intensification:

$$HA(x) = \begin{cases} 2A(x)^2 & \text{if } A(x) \in \left[0, \frac{1}{2}\right] \\ 1 - 2\left(1 - A(x)\right)^2 & \text{otherwise} \end{cases}$$

→ slightly

  - Vague

$$\boldsymbol{HA}(x) = \begin{cases} \sqrt{\frac{A(x)}{2}} & \text{if } A(x) \le 0.5 \\ 1 - \sqrt{\frac{1 - A(x)}{2}} & \text{if } A(x) > 0.5 \end{cases}$$

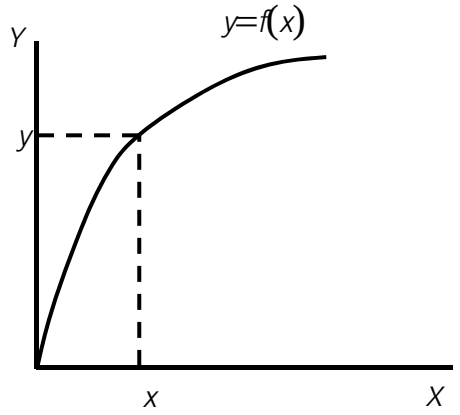
→ seldom

- Generalized modus ponen

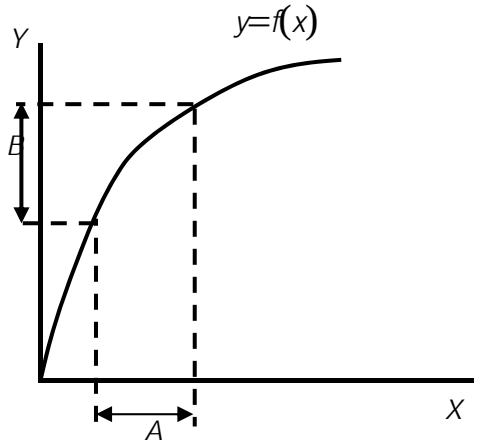
implication: If χ is $\boldsymbol{A}$, then Y is $\boldsymbol{B}$

premise: χ is $\boldsymbol{A}'$

conclusion: Y is $\boldsymbol{B}'$



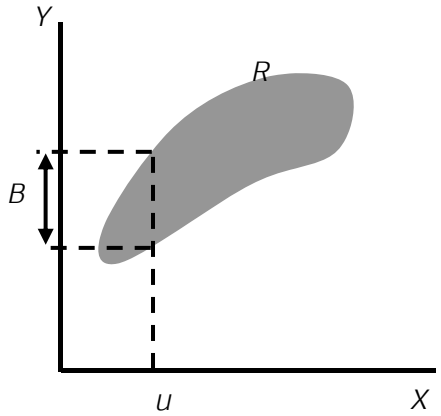
$\chi = x$ we get $Y = y = f(x)$

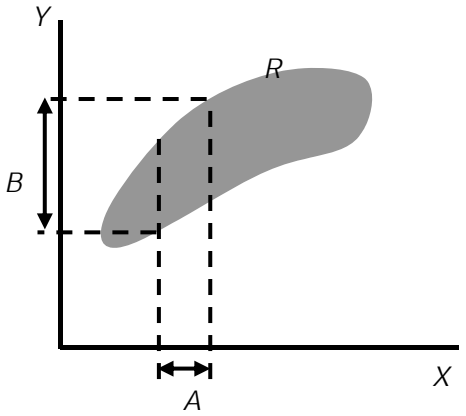


χ is in A we get Y in

$$B = \{ y \in Y \mid y = f(x), x \in A \}$$

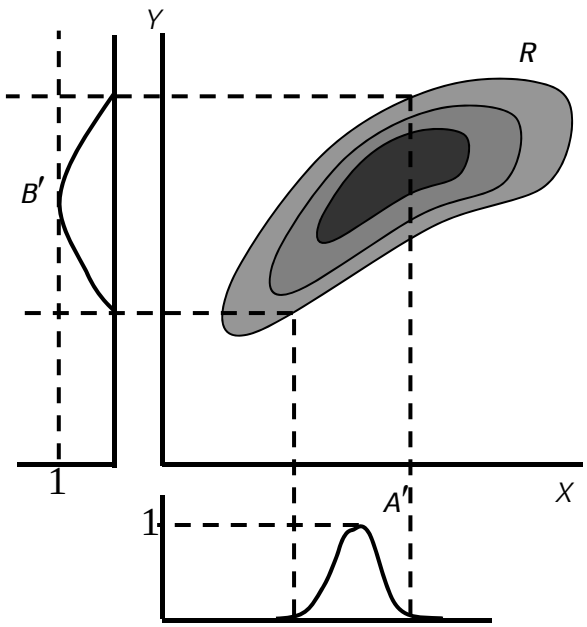
$$\mu_B(y) = \sup_{x \in X} \min\left[\mu_A(x), \mu_R(x,y)\right]$$





Relation R on $X \times Y$ if $\chi = u$ and R we get $Y \in B$ where $B = \{y \in Y \mid \langle x,y \rangle \in R\}$

Relation R on $X \times Y$ if $\chi \in A$ we get $Y \in B$ where
$B = \{ y \in Y \mid \langle x,y \rangle \in R, x \in A\}$



If ***R*** is a fuzzy relation on $X \times Y$ where ***A*** and ***A′*** are fuzzy sets on X, **B** and ***B′*** are fuzzy sets on Y

If ***R*** (relation between ***A*** and ***B***) and ***A′*** are given, we can find ***B′*** using

$$B'(y) = \sup_{x \in X} \min\left[A'(x), R(x,y)\right]$$

Or $B' = A' \bullet R$ operator $\bullet$ is a composition operator
Hence, this is called compositional rule of inference

- Example

  implication: if x and y are approximately equal

  premise: x is little

  conclusion: ?

Suppose little➔ $\boldsymbol{A} = \{(1,1), (2,0.6), (3,0.2), (4,0)\}$

Suppose

$$R=\begin{bmatrix} 1.0 & 0.5 & 0.0 & 0.0 \\ 0.5 & 1.0 & 0.5 & 0.0 \\ 0.0 & 0.5 & 1.0 & 0.5 \\ 0.0 & 0.0 & 0.5 & 1.0 \end{bmatrix}$$

← if x and y are approximately equal

From $\boldsymbol{B} = \boldsymbol{A} \bullet \boldsymbol{R}$

$$B=\begin{bmatrix} 1.0 & 0.6 & 0.2 & 0.0 \end{bmatrix} \bullet \begin{bmatrix} 1.0 & 0.5 & 0.0 & 0.0 \\ 0.5 & 1.0 & 0.5 & 0.0 \\ 0.0 & 0.5 & 1.0 & 0.5 \\ 0.0 & 0.0 & 0.5 & 1.0 \end{bmatrix}$$

→ $B = \begin{bmatrix} 1.0 & 0.6 & 0.5 & 0.2 \end{bmatrix}$

Max-min composition

- Several ways to create relation from implication:

  – If χ is $\boldsymbol{A}$, then Y is $\boldsymbol{B}$ for $x \in X$ and $y \in Y$, relation will be

$$\boldsymbol{R}(x,y) = \mathrm{I}(\boldsymbol{A}(x), \boldsymbol{B}(y))$$

Where I is fuzzy implication

- Lukasiewicz: $\mathrm{I}(\boldsymbol{A}(x),\boldsymbol{B}(y)) = \min[1, 1-\boldsymbol{A}(x)+\boldsymbol{B}(y)]$
- Zadeh: $\mathrm{I}(\boldsymbol{A}(x),\boldsymbol{B}(y)) = \max((1-\boldsymbol{A}(x)), \min(\boldsymbol{A}(x),\boldsymbol{B}(y)))$
- Kleen-Dienes: $\mathrm{I}(\boldsymbol{A}(x),\boldsymbol{B}(y)) = \max((1-\boldsymbol{A}(x)),\boldsymbol{B}(y))$
- Mamdani (correlation-min): $\mathrm{I}(\boldsymbol{A}(x),\boldsymbol{B}(y)) = \min(\boldsymbol{A}(x),\boldsymbol{B}(y))$
- Correlation-product: $\mathrm{I}(\boldsymbol{A}(x),\boldsymbol{B}(y)) = \boldsymbol{A}(x)\times\boldsymbol{B}(y)$
- Goedel:

$$I(A(x),B(y)) = \begin{cases} 1 & \text{if} \quad A(x) \le B(y) \\ B(y) & \text{if} \quad A(x) > B(y) \end{cases}$$

- If χ is $\boldsymbol{A}$, then Y is $\boldsymbol{B}$ →If χ is $\boldsymbol{A}$, then Y is $\boldsymbol{B}$ else V is ***UNKNOWN*** where χ, Y and V are variables in X, Y and V and $\boldsymbol{A}$, $\boldsymbol{B}$ and ***UNKNOWN*** are fuzzy sets on X, Y and V

$$\boldsymbol{R} = \boldsymbol{A} \times \boldsymbol{B} + \overline{\boldsymbol{A}} \times \boldsymbol{UNKNOWN}$$

$\times$ → minimum, $+$ → maximum

- If χ is $\boldsymbol{A}$, then Y is $\boldsymbol{B}$ else Z is $\boldsymbol{C}$ →

$$\boldsymbol{R} = \boldsymbol{A} \times \boldsymbol{B} + \overline{\boldsymbol{A}} \times \boldsymbol{C}$$

- Example

$\boldsymbol{A} = 1/1 + 0.4/2$, $\boldsymbol{B} = 0.4/2 + 1/3$ and $\boldsymbol{C} = 1/1 + 0.6/2$

If χ is $\boldsymbol{A}$, then Y is $\boldsymbol{B}$ else Z is $\boldsymbol{C}$ →

$$R = \begin{bmatrix} 1.0 \\ 0.4 \\ 0.0 \end{bmatrix} \begin{bmatrix} 0.0 & 0.4 & 1.0 \end{bmatrix} + \begin{bmatrix} 0.0 \\ 0.6 \\ 1.0 \end{bmatrix} \begin{bmatrix} 1.0 & 0.6 & 0.0 \end{bmatrix}$$

$$R = \begin{bmatrix} 0.0 & 0.4 & 1.0 \\ 0.0 & 0.4 & 0.4 \\ 0.0 & 0.0 & 0.0 \end{bmatrix} + \begin{bmatrix} 0.0 & 0.0 & 0.0 \\ 0.6 & 0.6 & 0.0 \\ 1.0 & 0.6 & 0.0 \end{bmatrix}$$

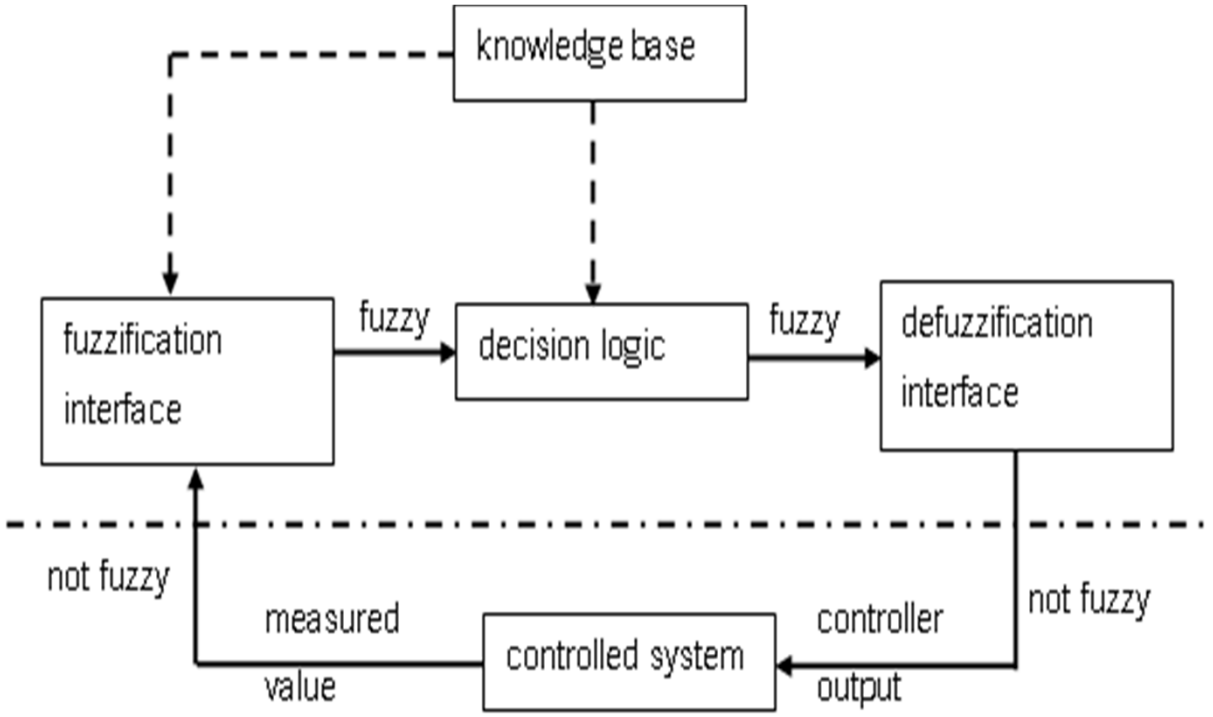
$$R = \begin{bmatrix} 0.0 & 0.4 & 1.0 \\ 0.6 & 0.6 & 0.4 \\ 1.0 & 0.6 & 0.0 \end{bmatrix}$$

If χ is $\boldsymbol{A}$, then Y is $\boldsymbol{B}$ →

$$R=\begin{bmatrix}1.0\\0.4\\0.0\end{bmatrix}\begin{bmatrix}0.0 & 0.4 & 1.0\end{bmatrix}+\begin{bmatrix}0.0\\0.6\\1.0\end{bmatrix}\begin{bmatrix}1.0 & 1.0 & 1.0\end{bmatrix}$$

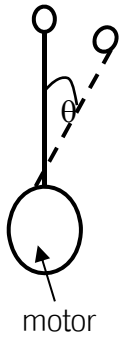
$$R=\begin{bmatrix}0.0 & 0.4 & 1.0\\0.0 & 0.4 & 0.4\\0.0 & 0.0 & 0.0\end{bmatrix}+\begin{bmatrix}0.0 & 0.0 & 0.0\\0.6 & 0.6 & 0.6\\1.0 & 1.0 & 1.0\end{bmatrix}$$

$$R=\begin{bmatrix}0.0 & 0.4 & 1.0\\0.6 & 0.6 & 0.6\\1.0 & 1.0 & 1.0\end{bmatrix}$$



Fuzzy Control: Simple Diagram

- Example: Inverted pendulum



input: angle θ and angular velocity Δθ

output: current v_t

membership functions→ same name for θ , Δθ and v_t

negative large (***NL***), negative medium (***NM***), negative small (***NS***), zero (***ZE***), positive small (***PS***), positive medium (***PM***), and positive large (***PL***)

rule *j*: if θ is $\boldsymbol{A}_j$ and Δθ is $\boldsymbol{B}_j$ then v_t is $\boldsymbol{C}_j$

e.g. if θ is ***NL*** and Δθ is ***NL*** then v_t is ***PL*** → (***NL***,***NL***;***PL***)

or if θ is ***ZE*** and Δθ is ***ZE*** then v_t is ***ZE*** → (***ZE***,***ZE***;***ZE***)

θ / $\Delta\theta$	*NL*	*NM*	*NS*	*ZE*	*PS*	*PM*	*PL*
NL				***PL***			
NM				***PM***			
NS				***PS***			
ZE	***PL***	***PM***	***PS***	***ZE***	***NS***	***NM***	***NL***
PS				***NS***			
PM				***NM***			
PL				***NL***			

Total number of rules→ $7 \times 7 \times 7 = 343$ but some of rules do not make sense. Hence, the number of usable rules will be less than that

- Fuzzy associative memory (FAM)

rule 1: If χ is $\boldsymbol{A}_1$, then Y is $\boldsymbol{B}_1 \rightarrow (\boldsymbol{A}_1,\boldsymbol{B}_1) \rightarrow \boldsymbol{R}_1$

rule 2: If χ is $\boldsymbol{A}_2$, then Y is $\boldsymbol{B}_2 \rightarrow (\boldsymbol{A}_2,\boldsymbol{B}_2) \rightarrow \boldsymbol{R}_2$

•
•
•

rule n: If χ is $\boldsymbol{A}_n$, then Y is $\boldsymbol{B}_n \rightarrow (\boldsymbol{A}_n,\boldsymbol{B}_n) \rightarrow \boldsymbol{R}_n$

premise: χ is $\boldsymbol{A}'$

conclusion: Y is $\boldsymbol{B}'$

$$\boldsymbol{B}' = w_1\boldsymbol{B}_1' + w_2\boldsymbol{B}_2' + \ldots + w_n\boldsymbol{B}_n' \text{ where } \boldsymbol{B}_i' = \boldsymbol{A}' \bullet \boldsymbol{R}_i$$

Supposed: $\boldsymbol{A}_i = a_1/x_1 + a_2/x_2 + ... + a_m/x_m \rightarrow \boldsymbol{A}_i = [a_1, a_2, ..., a_m]$

Correlation-min:

$$R_i = A_i^T \circ B_i \quad \rightarrow \quad R_i = \begin{bmatrix} a_1 \wedge B_i \\ a_2 \wedge B_i \\ \vdots \\ a_m \wedge B_i \end{bmatrix}$$

OR

$$R_i = \begin{bmatrix} b_1 \wedge A_i^T & b_2 \wedge A_i^T \cdots b_p \wedge A_i^T \end{bmatrix}$$

Correlation-product :

$$R_i = A_i^T \circ B_i \quad \rightarrow \quad R_i = \begin{bmatrix} a_1 B_i \\ a_2 B_i \\ \vdots \\ a_m B_i \end{bmatrix}$$

OR

$$R_i = \begin{bmatrix} b_1 A_i^T & b_2 A_i^T \cdots b_p A_i^T \end{bmatrix}$$

Conclude→ $\boldsymbol{B'} = \boldsymbol{A'} \bullet \boldsymbol{R}$

$$= \boldsymbol{A'} \bullet \bigcup_{j \in \mathbb{N}_n} R_j$$

$$= \boldsymbol{A'} \bullet \sup_{j \in \mathbb{N}_n} R_j$$

$$= \sup_{j \in \mathbb{N}_n} \left(A' \bullet R_j \right)$$

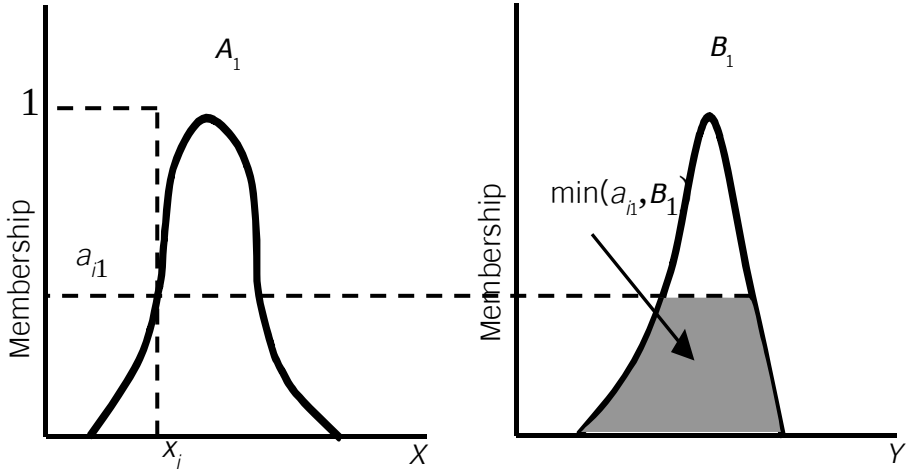
$$= \sup_{j \in \mathbb{N}_n} \left(A' \bullet A_j^{\mathsf{T}} \circ B_j \right)$$

$$= \sup_{j \in \mathbb{N}_n} \left(\left(A' \bullet A_j^{\mathsf{T}} \right) \circ B_j \right)$$

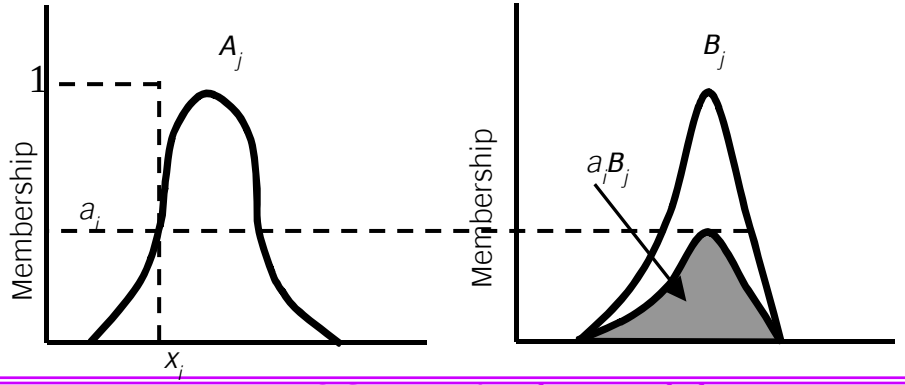
$$B = \sup \left[\left(A' \bullet A_1^{\mathsf{T}} \right) \circ B_1, \left(A' \bullet A_2^{\mathsf{T}} \right) \circ B_2, \ldots, \left(A' \bullet A_i^{\mathsf{T}} \right) \circ B_i, \ldots, \left(A' \bullet A_n^{\mathsf{T}} \right) \circ B_n \right]$$

Example: $\boldsymbol{A}' = 0/x_1 + 0/x_2 + ... + 1/x_i + 0/x_{i+1} + ... + 0/x_m$

Correlation-min→ $(\boldsymbol{A}' \bullet \boldsymbol{A}_j^T) \circ \boldsymbol{B}_j = \min(\boldsymbol{A}_i(x_i), \boldsymbol{B}_j)$



Correlation-product→ $(\boldsymbol{A}' \bullet \boldsymbol{A}_j^T) \circ \boldsymbol{B}_j = \boldsymbol{A}_i(x_i)\boldsymbol{B}_j$

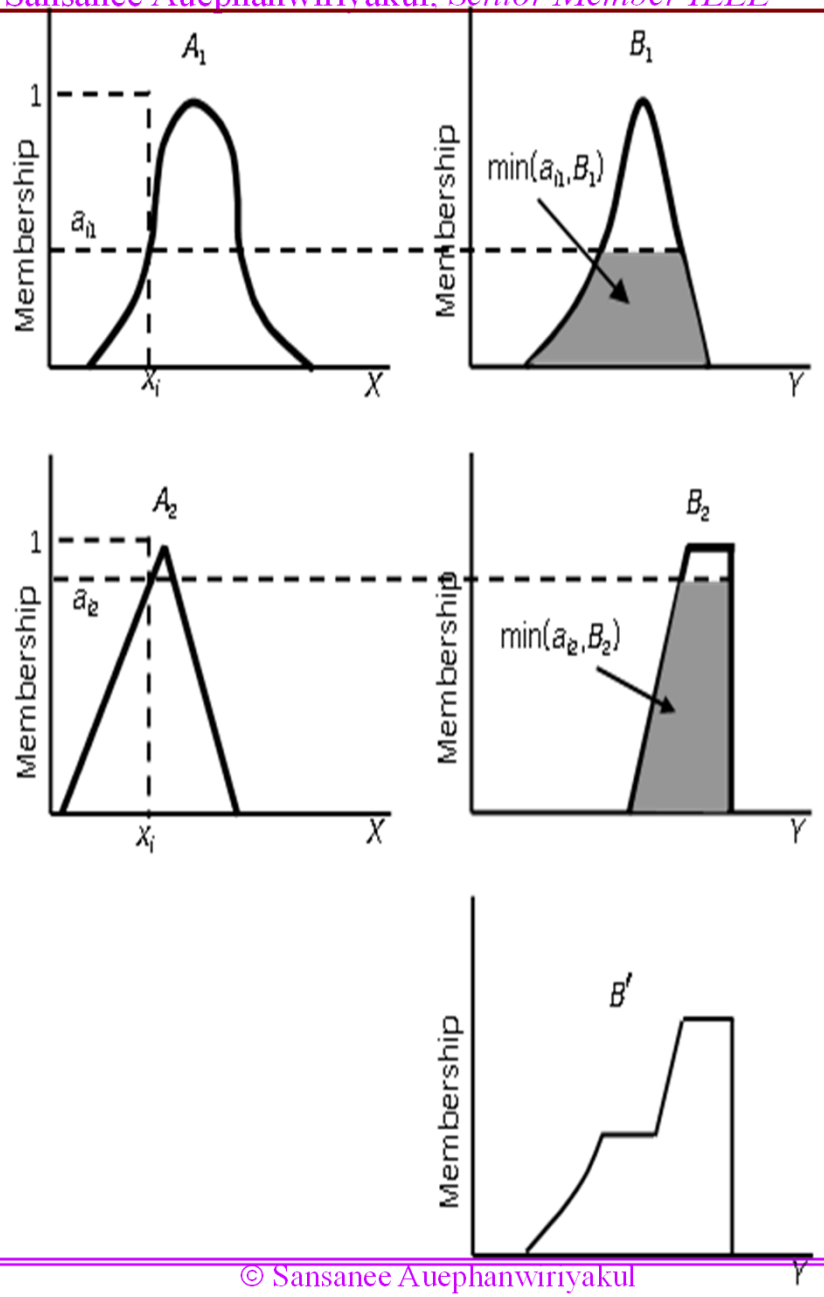


- Example: $\boldsymbol{A}' = 0/x_1 + 0/x_2 + ... + 1/x_i + 0/x_{i+1} + ... + 0/x_m$

Supposed that there are 2 rules $(\boldsymbol{A}_1;\boldsymbol{B}_1)$ and $(\boldsymbol{A}_2;\boldsymbol{B}_2)$

$$\boldsymbol{B}' = \sup\left[(\boldsymbol{A}' \bullet \boldsymbol{A}_1^{\mathrm{T}}) \circ \boldsymbol{B}_1, (\boldsymbol{A}' \bullet \boldsymbol{A}_2^{\mathrm{T}}) \circ \boldsymbol{B}_2\right]$$

where $(\boldsymbol{A}' \bullet \boldsymbol{A}_1^{\mathrm{T}}) = a_{i1}$ and $(\boldsymbol{A}' \bullet \boldsymbol{A}_2^{\mathrm{T}}) = a_{i2}$



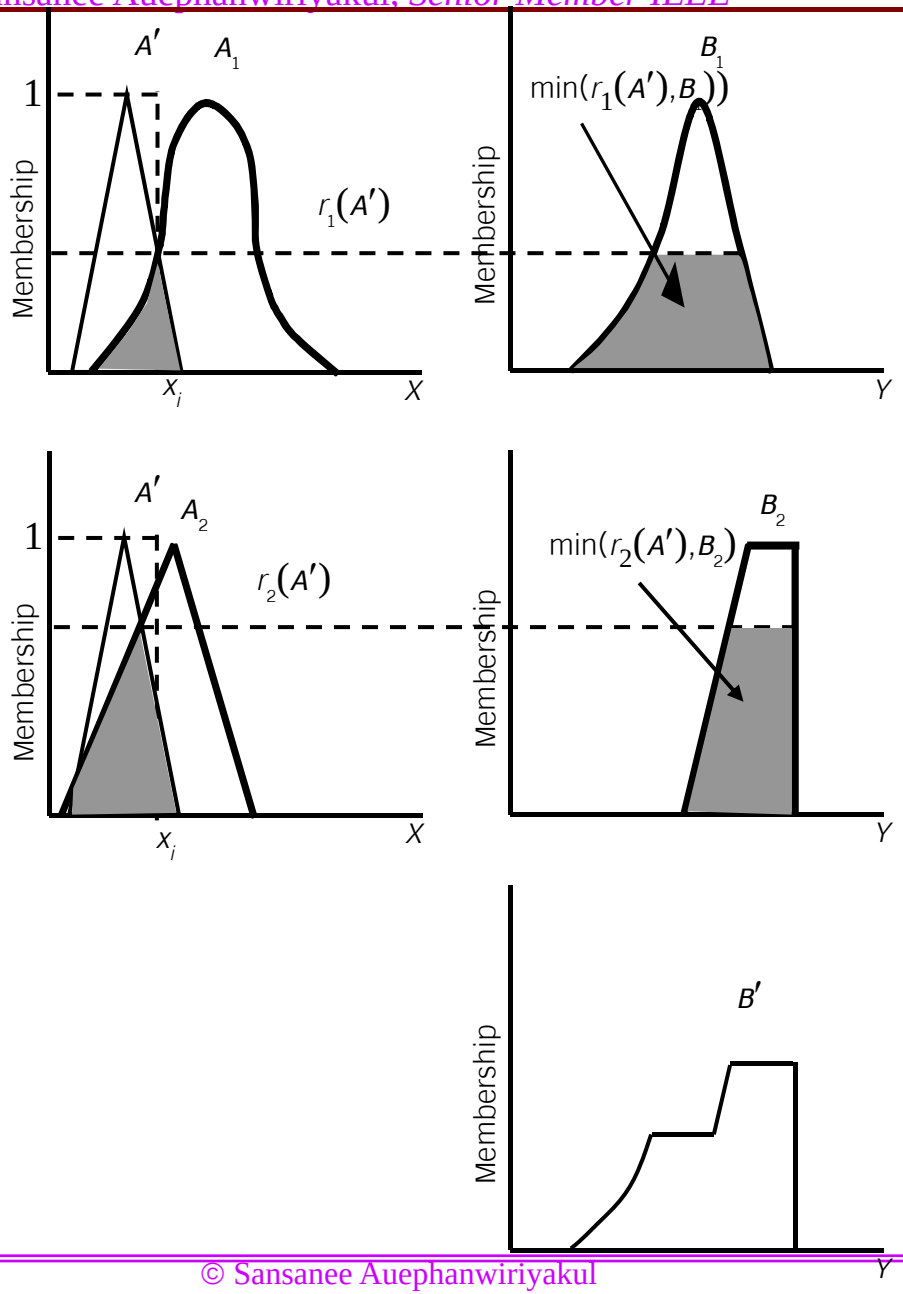
- Example

$\boldsymbol{A}'$ is a fuzzy set not crisp set

Supposed that there are 2 rules($\boldsymbol{A}_1$;$\boldsymbol{B}_1$) and ($\boldsymbol{A}_2$; $\boldsymbol{B}_2$)

$\boldsymbol{B}' = \sup\left[(\boldsymbol{A}' \bullet \boldsymbol{A}_1^{\mathrm{T}}) \circ \boldsymbol{B}_1, (\boldsymbol{A}' \bullet \boldsymbol{A}_2^{\mathrm{T}}) \circ \boldsymbol{B}_2\right]$

where $r_j(A') = \left(A' \bullet A_j^T\right) = \sup_{x \in X} \min\left[A'(x), A_j(x)\right]$



- If there are more than 1 input→ $(\boldsymbol{A},\boldsymbol{B};\boldsymbol{C})$

$\boldsymbol{A} = a_1/x_1 + a_2/x_2 + ... + a_m/x_m$ and $\boldsymbol{B} = b_1/y_1 + b_2/y_2 + ... + b_m/y_m$

split into $(\boldsymbol{A};\boldsymbol{C})$ and $(\boldsymbol{B};\boldsymbol{C})$ →

$$M_{AC} = A^T \circ C \qquad \text{and} \qquad M_{BC} = B^T \circ C$$

fact: $\boldsymbol{A}' = \mathrm{I}_x^i = 0/x_1 + 0/x_2 + ... + 1/x_i + 0/x_{i+1} + ... + 0/x_m$ and

$\boldsymbol{B}' = \mathrm{I}_y^j = 0/y_1 + 0/y_2 + ... + 1/y_j + 0/j_{i+1} + ... + 0/y_m$

compute: $\mathrm{F}(\boldsymbol{A}', \boldsymbol{B}') = \boldsymbol{C}' = [\boldsymbol{A}' \bullet \boldsymbol{M}_{\boldsymbol{AC}}] \cap [\boldsymbol{B}' \bullet \boldsymbol{M}_{\boldsymbol{BC}}]$

where $\boldsymbol{A}' \bullet \boldsymbol{M}_{\boldsymbol{AC}} = \mathrm{I}_x^i \bullet \boldsymbol{M}_{\boldsymbol{AC}}$

$$= \mathrm{I}_x^j \bullet \begin{bmatrix} a_1 \wedge C \\ a_2 \wedge C \\ \vdots \\ a_m \wedge C \end{bmatrix} = a_i \wedge \boldsymbol{C}$$

and $\boldsymbol{B'} \bullet \boldsymbol{M_{BC}} = \mathrm{I}_y^j \bullet \boldsymbol{M_{BC}}$

$$= \mathrm{I}_y^j \bullet \begin{bmatrix} b_1 \wedge C \\ b_2 \wedge C \\ \vdots \\ b_m \wedge C \end{bmatrix} \qquad = b_j \wedge \boldsymbol{C}$$

Hence, the conclusion will be $\boldsymbol{C'} = (a_i \wedge \boldsymbol{C}) \cap (b_j \wedge \boldsymbol{C}) = (\min(a_i, b_j)) \wedge \boldsymbol{C}$

If use correlation-product

$$\boldsymbol{C'} = (a_i \boldsymbol{C}) \cap (b_j \boldsymbol{C}) = (\min(a_i, b_j)) \boldsymbol{C}$$

- Mamdani model

  Rule: If ξ_1 is $\boldsymbol{A}^{(1)}$, and ξ_2 is $\boldsymbol{A}^{(2)}$ and ... and ξ_n is $\boldsymbol{A}^{(n)}$ then η is $\boldsymbol{B}$

where $\boldsymbol{A}^{(1)}$, $\boldsymbol{A}^{(2}$ and ... and $\boldsymbol{A}^{(n)}$ are linguistic terms in $\boldsymbol{T}(x_i)$ for $1 \le i \le n$ and $\boldsymbol{B}$ is linguistic term in $\boldsymbol{T}(y)$ and

suppose there are more than 1 rule

Let $\boldsymbol{T}(x_1) \rightarrow \boldsymbol{A}_1^{(1)}, \boldsymbol{A}_2^{(1)}, \ldots, \boldsymbol{A}_{N1}^{(1)}$

$\boldsymbol{T}(x_2) \rightarrow \boldsymbol{A}_1^{(2)}, \boldsymbol{A}_2^{(2)}, \ldots, \boldsymbol{A}_{N2}^{(2)}$

$\vdots$

$\boldsymbol{T}(x_n) \rightarrow \boldsymbol{A}_1^{(n)}, \boldsymbol{A}_2^{(n)}, \ldots, \boldsymbol{A}_{Nn}^{(n)}$

and $\boldsymbol{T}(y) \rightarrow \boldsymbol{B}_1, \boldsymbol{B}_2, \ldots, \boldsymbol{B}_{N0}$

rule j: If ξ_1 is $\boldsymbol{A}_{i1,j}^{(1)}$, and ξ_2 is $\boldsymbol{A}_{i2,j}^{(2)}$ and ... and ξ_n is $\boldsymbol{A}_{in,j}^{(n)}$ then η is $\boldsymbol{B}_{i,j}$

where $i1 \in \{1,2,...,N1\}$, $i2 \in \{1,2,...,N2\}$,..., $in \in \{1,2,...,Nn\}$ and

$i \in \{1,2,...,N0\}$

Firing strength of rule j will be

$$\alpha_j = \min\{\boldsymbol{A}_{i1,j}{}^{(1)}(x_1), \boldsymbol{A}_{i2,j}{}^{(2)}(x_2), \ldots, \boldsymbol{A}_{in,j}{}^{(n)}(x_n)\}$$

Output of rule j will be

$$OUT_{x_1,x_2,\ldots,x_n}^{(j)}(y) = \min\left[A_{1,j}^{(1)}(x_1), A_{2,j}^{(2)}(x_2), \ldots, A_{in,j}^{(n)}(x_n), B_{i,j}(y)\right]$$

Overall output

$$OUT_{x_1,x_2,\ldots,x_n}(y) = \max_{j\in\{1,2,\ldots,k\}} \min\left[A_{1,j}^{(1)}(x_1), A_{2,j}^{(2)}(x_2), \ldots, A_{in,j}^{(n)}(x_n), B_{i,j}(y)\right]$$

- Defuzzification

  – Max membership→ select de_y where $\boldsymbol{B}'(de\_y) \geq \boldsymbol{B}'(de\_y')$ for all $de\_y' \in Y$
  if $M=\{y \in [y_1, y_2] \mid \boldsymbol{B}'(y)=h(\boldsymbol{B}')\}$ where $h(\boldsymbol{B}')$ is a height of $\boldsymbol{B}'$

$$de\_y = \frac{\sum_{y_k \in M} y_k}{|M|}$$

  – Center of area (centroid)

$$de\_y = \frac{\sum_k B'(y_k) y_k}{\sum_k B'(y_k)}$$

– Simplified centroid

$$de\_y = \frac{\sum_{j \in N_n} c_j y^0_{B_j}}{\sum_{j \in N_n} c_j}$$

where c_j is firing strength (a_{ij} or $r_j(\boldsymbol{A}')$) and $y^0_{\boldsymbol{B}j}$ is the mode of output of rule j

- Example

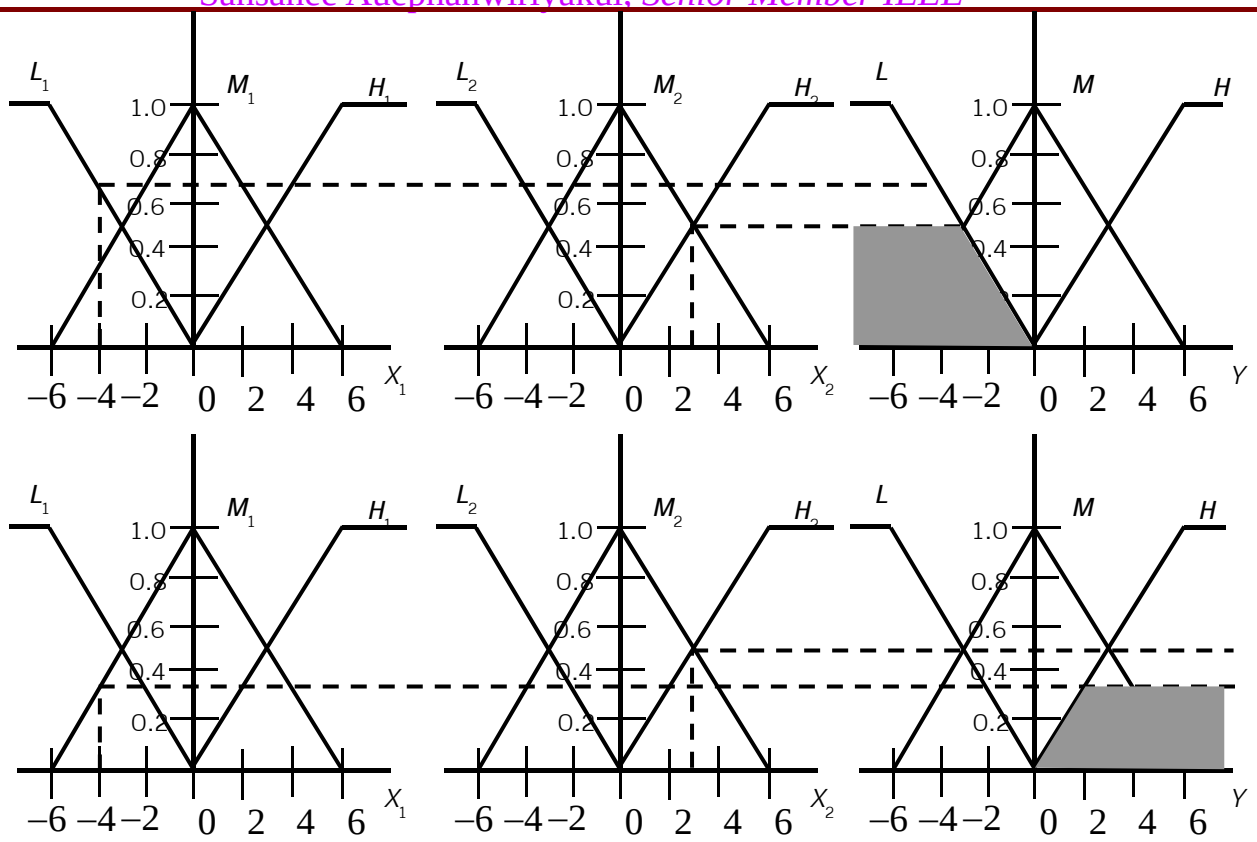
  rule 1: If x_1 is $\boldsymbol{L}_1$ และ x_2 is $\boldsymbol{H}_2$, then y is $\boldsymbol{L}$

  rule 2: If x_1 is $\boldsymbol{M}_1$ และ x_2 is $\boldsymbol{M}_2$, then y is $\boldsymbol{H}$

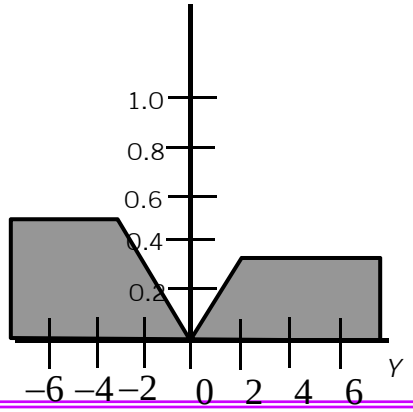
input (-4,3)

$$\alpha_1 = \min(\boldsymbol{L}_1(-4), \boldsymbol{H}_2(3)) = \min(0.67, 0.5) = 0.5$$

$$\alpha_2 = \min(\boldsymbol{M}_1(-4), \boldsymbol{M}_2(3)) = \min(0.33, 0.5) = 0.5$$



Defuzzify the output fuzzy set
to get the final output



- Takagi-Sugeno model

Rule: If ξ_1 is $\boldsymbol{A}^{(1)}$, and ξ_2 is $\boldsymbol{A}^{(2)}$ and ... and ξ_n is $\boldsymbol{A}^{(n)}$ then η is $\boldsymbol{B}$

where $\boldsymbol{A}^{(1)}$, $\boldsymbol{A}^{(2}$ and ... and $\boldsymbol{A}^{(n)}$ are linguistic terms in $\boldsymbol{T}(x_i)$ for $1 \le i \le n$ and $\boldsymbol{B}$ is linguistic term in $\boldsymbol{T}(y)$ and

suppose there are more than 1 rule

Let $\boldsymbol{T}(x_1) \rightarrow \boldsymbol{A}_1^{(1)}, \boldsymbol{A}_2^{(1)}, \ldots, \boldsymbol{A}_{N1}^{(1)}$

$\boldsymbol{T}(x_2) \rightarrow \boldsymbol{A}_1^{(2)}, \boldsymbol{A}_2^{(2)}, \ldots, \boldsymbol{A}_{N2}^{(2)}$

$\vdots$

$\boldsymbol{T}(x_n) \rightarrow \boldsymbol{A}_1^{(n)}, \boldsymbol{A}_2^{(n)}, \ldots, \boldsymbol{A}_{Nn}^{(n)}$

rule j: If ξ_1 is $\boldsymbol{A}_{i1,j}^{(1)}$, and ξ_2 is $\boldsymbol{A}_{i2,j}^{(2)}$ and ... and ξ_n is $\boldsymbol{A}_{in,j}^{(n)}$ then

$\eta_j = f_j(\xi_1, \xi_2, \ldots, \xi_n)$

where $i1 \in \{1,2,\ldots,N1\}$, $i2 \in \{1,2,\ldots,N2\}$, ..., $in \in \{1,2,\ldots,Nn\}$ and

$$f_j(x_1, x_2, \ldots, x_n) = a_1^j x_1 + a_2^j x_2 + \ldots + a_n^j x_n + a_0^j$$

overall output→

$$\eta = \frac{\sum_{j=1}^{k} \alpha_j f_j\left(x_1, x_2, \ldots x_n\right)}{\sum_{j=1}^{k} \alpha_j}$$

where α_j is the firing degree of rule j

No need for defuzzification

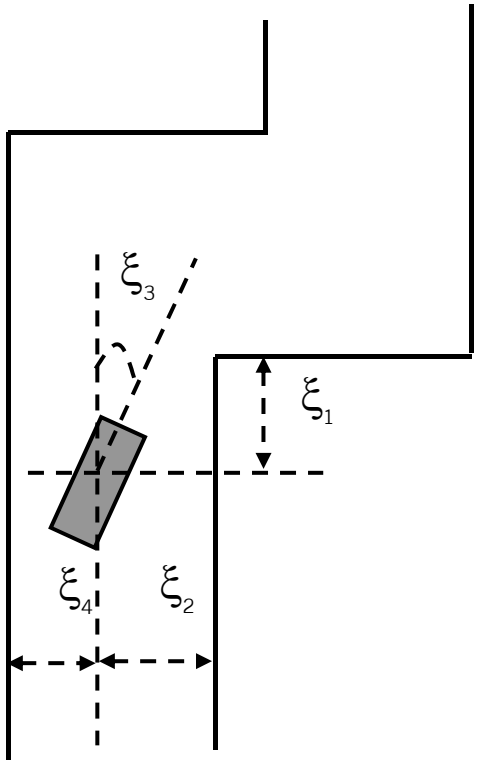
- Example → turning steering wheel control system

  input: ξ_1, ξ_2, ξ_3 and ξ_4

  universal set: X_1=[0, 150]cm, X_2=[0, 150] cm,

  X_3=[–90, 90] degree and X_4=[0, 150] cm

  output: rotation speed (η) of the steering wheel



Rule j: If ξ_1 is $\boldsymbol{A}_{i1,j}^{(1)}$, และ ξ_2 is $\boldsymbol{A}_{i2,j}^{(2)}$ และ ξ_3 is $\boldsymbol{A}_{j3,j}^{(3)}$ และ ξ_4 is $\boldsymbol{A}_{j4,j}$ then $\eta = p_0 + p_1\xi_1 + p_2\xi_2 + p_3\xi_3 + p_4\xi_4$

rule	ξ_1	ξ_2	ξ_3	ξ_4	p_0	p_1	p_2	p_3	p_4
1	–	–	outwards	small	3.000	0.000	0.000	-0.045	-0.004
2	–	–	forward	small	3.000	0.000	0.000	-0.030	-0.090
3	small	small	outwards	–	3.000	-0.041	0.004	0.000	0.000
4	small	small	forward	–	0.303	-0.026	0.061	-0.050	0.000
5	small	small	inwards	–	0.000	-0.025	0.070	-0.075	0.000
6	small	big	outwards	–	3.000	-0.066	0.000	-0.034	0.000
7	small	big	forward	–	2.990	-0.017	0.000	-0.021	0.000
8	small	big	inwards	–	1.500	0.025	0.000	-0.050	0.000
9	medium	small	outwards	–	3.000	-0.017	0.005	-0.036	0.000
10	medium	small	forward	–	0.053	-0.038	0.080	-0.034	0.000
11	medium	small	inwards	–	-1.220	-0.016	0.047	-0.018	0.000
12	medium	big	outwards	–	3.000	-0.027	0.000	-0.044	0.000
13	medium	big	forward	–	7.000	-0.049	0.000	-0.041	0.000
14	medium	big	inwards	–	4.000	-0.025	0.000	-0.100	0.000
15	big	small	outwards	–	0.370	0.000	0.000	-0.007	0.000
16	big	small	forward	–	-0.900	0.000	0.034	-0.030	0.000
17	big	small	inwards	–	-1.500	0.000	0.005	-0.100	0.000
18	big	big	outwards	–	1.000	0.000	0.000	-0.013	0.000
19	big	big	forward	–	0.000	0.000	0.000	-0.006	0.000
20	big	big	inwards	–	0.000	0.000	0.000	-0.010	0.000

Input

$\xi_1 = 10$ cm, $\xi_2 = 30$ cm, $\xi_3 = 0$ degree (straight forward) and $\xi_4 = 50$ cm

Only rule 4 and 7→ fire

$\alpha_4 = 0.25$ and $\eta_4 = 0.303 - 0.026(10) + 0.061(30) - 0.050(0) + 0.000(50) = 1.873$

$\alpha_7 = 0.167$ and $\eta_7 = 2.990 - 0.017(10) + 0.000(30) - 0.021(0) + 0.000(50) = 2.820$

Output of the system will be

$$\eta = \frac{0.25(1.873) + 0.167(2.820)}{0.25 + 0.167} = 2.252$$

Fuzzy measure & fuzzy integral

- Fuzzy measure

Address the ambiguity axis of uncertainty →how likely can the answer to a question be found in various subsets of the sources of information

$X = \{x_1, x_2, \ldots x_n\}$, g: $2^X \rightarrow [0,1]$

Properties

1. $g(\phi) = 0$
2. $g(A) \leq g(B)$ if $A \subseteq B$

$g^i = g\{x_i\}$ → fuzzy density function→ importance of the single information source x_i in determine the answer to a particular question

- Sugeno Lambda measure→satisfy 2 properties with additional one

  3. For all A,B $\subseteq$X with A$\cap$B = ϕ

$$g_\lambda(A\cap B)=g_\lambda(A)+g_\lambda(B)+\lambda g_\lambda(A)g_\lambda(B) \quad \exists\lambda>-1$$

From $X=\bigcup_{i=1}^{n}x_i$ and $g_\lambda(X)=1$ →

$$(1+\lambda)=\prod_{i=1}^{n}(1+\lambda g^i)$$

$$=1+\lambda\sum_{j=1}^{n}g^j+\lambda^2\sum_{j=1}^{n-1}\left(\sum_{k=j+1}^{n}g^jg^k\right)+\lambda^3\sum_{j=1}^{n-2}\left(\sum_{k=1}^{n-1}\sum_{i=1}^{n}g^jg^kg^i\right)+\ldots+\lambda^n g^1g^2..g^n$$

- Example

X=$\{x_1, x_2, x_3\}$→ fuzzy densities $g^1=0.2$, $g^2=0.3$, $g^3=0.1$

find λ using $(1+\lambda)=(1+0.2\lambda)(1+0.3\lambda)(1+0.1\lambda)$

$$0.006\lambda^2+0.11\lambda-0.4=0$$

$$\lambda=\frac{-0.11\pm\sqrt{0.11^2-4(0.006)(-0.4)}}{2(0.006)}=3.2,-21.44$$

Hence $\lambda=3.2$

Subset	g_λ
ϕ	0→ from property
$\{x_1\}$	0.2
$\{x_2\}$	0.3
$\{x_3\}$	0.1
$\{x_1, x_2\}$	0.69
$\{x_1, x_3\}$	0.36
$\{x_2, x_3\}$	0.5
$\{x_1, x_2, x_3\}$	1 → from property

$g(\{x_1, x_2\}) = 0.2+0.3+3.2(0.2)(0.3) =$

$g(\{x_1, x_3\}) = 0.2+0.1+3.2(0.2)(0.1) =$

$g(\{x_2, x_3\}) = 0.3+0.1+3.2(0.3)(0.1) =$

$g(\{x_1, x_2, x_3\}) = 0.69+0.1+3.2(0.69)(0.1)$

$= 1.01\approx 1$

- Fuzzy integral
  - Sugeno fuzzy integral

  Contiuous case:

$$\int h(x) \circ g = \sup_{\alpha \in [0,1]} \min\left[\alpha, g(A_\alpha)\right] \quad \text{where } A_\alpha = \left\{x \middle| h(x) \geq \alpha\right\}$$

  Finite case: $X=\{x_1, x_2, \ldots, x_n\}$ → reorder

$$X = \left\{x_{(1)}, x_{(2)}, \ldots x_{(n)}\right\} \quad \text{so that } h\left(x_{(1)}\right) \geq h\left(x_{(2)}\right) \geq \ldots \geq h\left(x_{(n)}\right)$$

  Hence, sugeno fuzzy integral →

$$S_g(h) = \max_{i=1}^{n} \min\left[h\left(x_{(i)}\right), g\left(A_{(i)}\right)\right] \quad \text{where } A_{(i)} = \left\{x_{(1)}, x_{(2)}, \ldots x_{(i)}\right\}$$

- Example

X=$\{x_1, x_2, x_3\}$ → $h(x_1) = 0.7$, $h(x_2) = 0.9$, $h(x_3) = 0.2$

and assume that g are the same as previous example

Reorder $h(x_2) > h(x_1) > h(x_3)$

$S_g = (0.9 \wedge g(\{x_2\})) \vee (0.7 \wedge g(\{x_1, x_2\})) \vee (0.2 \wedge g(\{x_1, x_2, x_3\})) = 0.69$

– Choquet fuzzy integral

Contiuous case:

$$\int_X h(x) \circ g = \int_0^1 g(A_\alpha)\, d\alpha \qquad \text{where } A_\alpha = \{x \mid h(x) \geq \alpha\}$$

Finite case: $X=\{x_1, x_2, \ldots, x_n\}$ → reorder

$$X = \{x_{(1)}, x_{(2)}, \ldots x_{(n)}\} \text{ so that } h(x_{(1)}) \geq h(x_{(2)}) \geq \ldots \geq h(x_{(n)})$$

Hence, choquet fuzzy integral →

$$C_g(h) = \sum_{i=1}^{n} \left[h(x_{(i)}) - h(x_{(i+1)}) \right] g(A_{(i)}) \quad \text{where } A_{(i)} = \{x_{(1)}, x_{(2)}, \ldots x_{(i)}\} \text{ and } h(x_{(n+1)}) = 0$$

Let $\delta_i(g) = g(A_i) - g(A_{i-1})$ →

$$C_g(h) = \sum_{i=1}^{n} \left[h\left(x_{(i)}\right) - h\left(x_{(i+1)}\right) \right] g\left(A_{(i)}\right)$$

$$= \left[h\left(x_{(1)}\right) - h\left(x_{(2)}\right) \right] g\left(A_{(1)}\right) + \left[h\left(x_{(2)}\right) - h\left(x_{(3)}\right) \right] g\left(A_{(2)}\right) + \ldots + \left[h\left(x_{(n)}\right) - h\left(x_{(n+1)}\right) \right] g\left(A_{(n)}\right)$$

$$= h\left(x_{(1)}\right) g\left(A_{(1)}\right) + h\left(x_{(2)}\right) \left[g\left(A_{(2)}\right) - g\left(A_{(1)}\right) \right] + \ldots + h\left(x_{(n)}\right) \left[g\left(A_{(n)}\right) - g\left(A_{(n-1)}\right) \right] - h\left(x_{(n+1)}\right) g\left(A_{(n)}\right)$$

Hence

$$C_g(h) = \sum_{i=1}^{n} \delta_i(g) h\left(x_{(i)}\right) \text{ where } \delta_i(g) = \left[g\left(A_{(i)}\right) - g\left(A_{(i-1)}\right) \right]$$

We can have linear order statistic

$$LOS_{w_k}\left(h(x)\right) = \sum_{x_k \in w_k} w_k h\left(x_{(k)}\right)$$

where $\{w_1, w_2, \ldots, w_n\}$ satisfy $w_i \in [0,1]$ and $\sum_{k=1}^{n} w_k = 1$

- Example

X=$\{x_1, x_2, x_3\}$ → $h(x_1) = 0.7$, $h(x_2) = 0.9$, $h(x_3) = 0.2$

and assume that g are the same as previous example

Reorder $h(x_2) > h(x_1) > h(x_3)$

C_g=(0.9-0.7)(0.3)+(0.7-0.2)(0.69+(0.2-0.1)(1) = 0.605

- Optimization training

Let training set → T={(o_j, α_j)|j=1,2,…,m } where o_j=j^{th} object and α_j= desired output of j^{th} object, and there are m training samples

Suppose→ there are n sensors/algorithms

Output from choquet integral of j^{th} object will be

$$C_g\left(h(o_j)\right)=\sum_{i=1}^{n}\left[h\left(o_j;x_{(i)}\right)-h\left(o_j;x_{(i+1)}\right)\right]g\left(A_{(i)}\right)$$

Want to find fuzzy measure $\vec{g}=\begin{bmatrix} g_1 \\ g_2 \\ \dots \\ g_n \\ g_{12} \\ g_{13} \\ \dots \\ g_X \end{bmatrix}$

No	$h(x_1)$	$h(x_2)$	…	$h(x_n)$	α
1					
2					
…					
m					

Training set with m samples each has desired output (α). There are n sensors/algorithms

such that $C_g(h(o_j))$ is close to α_j → $E^2 = \sum_{j=1}^{m} \left[C_g\left(h(o_j)\right) - \alpha_j \right]^2$

Let $v_{j,A} = \begin{cases} h(o_j; x_{(i)}) - h(o_j; x_{(i+1)}) & \text{if } A = A_i \ \exists i \\ 0 & \text{else} \end{cases}$

We have $\vec{v}_j = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ h(o_j; x_{(1)}) - h(o_j; x_{(2)}) \\ \vdots \\ h(o_j; x_{(n-1)}) - h(o_j; x_{(n)}) \\ 0 \\ \vdots \\ 0 \end{bmatrix}$

$\vec{v}_j \in R^{2n-2}$ with only $n-1$ possible nonzero

Then $C_g\left(h(o_j)\right) = \vec{v}_j^{\,T} \vec{g} + h(o_j; x_{(n)})$

Objective function

$$E^2 = \frac{1}{2}\sum_{j=1}^{m}\left[C_g\left(h\left(o_j\right)\right)-\alpha_j\right]^2$$

$$= \frac{1}{2}\sum_{j=1}^{m}\left[\vec{v}_j^{\,T}\vec{g}+h\left(o_j;x_{(n)}\right)-\alpha_j\right]^T\left[\vec{v}_j^{\,T}\vec{g}+h\left(o_j;x_{(n)}\right)-\alpha_j\right]$$

$$= \frac{1}{2}\sum_{j=1}^{m}\left[\vec{g}^T\vec{v}_j\vec{v}_j^{\,T}\vec{g}+2\left[h\left(o_j;x_{(n)}\right)-\alpha_j\right]\vec{v}_j^{\,T}\vec{g}+\left[h\left(o_j;x_{(n)}\right)-\alpha_j\right]^2\right]$$

$$= \frac{1}{2}\vec{g}^T\left[\sum_{j=1}^{m}\vec{v}_j\vec{v}_j^{\,T}\right]\vec{g}+\left[\sum_{j=1}^{m}\left[h\left(o_j;x_{(n)}\right)-\alpha_j\right]\vec{v}_j^{\,T}\right]\vec{g}+\frac{1}{2}\sum_{j=1}^{m}\left[h\left(o_j;x_{(n)}\right)-\alpha_j\right]^2$$

Hence

$$E^2 = \frac{1}{2}\vec{g}^T\mathbf{D}\vec{g}+\vec{v}^T\vec{g}+\beta^2 \quad \text{where } \mathbf{D}=\sum_{j=1}^{m}\vec{v}_j\vec{v}_j^{\,T},$$

$$\vec{v}=\sum_{j=1}^{m}\left[h\left(o_j;x_{(n)}\right)-\alpha_j\right]\vec{v}_j^{\,T} \text{ and } \beta^2=\frac{1}{2}\sum_{j=1}^{m}\left[h\left(o_j;x_{(n)}\right)-\alpha_j\right]^2$$

Similar to

$$\underset{\vec{g}}{\text{minimize}} \frac{1}{2}\vec{g}^T \mathbf{D}\vec{g} + \vec{v}^T\vec{g}$$

subject to $\mathbf{A}\ \vec{g} \le \vec{b}$ and $\vec{0} \le \vec{g} \le \vec{1}$

From

$g_1 - g_{12} \leq 0$

$g_1 - g_{13} \leq 0$

…

$g_1 - g_{12\ldots n} \leq 0$

…

and

$g_{123\ldots n\text{-}1} - g_{123\ldots n} \leq 0$

…

$g_{23\ldots n} - g_{123\ldots n} \leq 0$

→

$$\mathbf{A} = \begin{bmatrix} 1\,0\cdots 0\,{-1}\,0\cdots 0\cdots & 0 \\ 1\,0\cdots 0\;\;0\;{-1}\;\;0\cdots & 0 \\ \vdots & \\ 0\,0\cdots 0\cdots & 0\,1\,0\cdots\;0 \\ \vdots & \\ 0\,0\cdots 0\cdots & 0\,0\,0\cdots 1 \end{bmatrix} \qquad \vec{b} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

- Example

Let

No	$h(x_1)$	$h(x_2)$	$h(x_3)$	α
1	0.68	0.53	0.81	0.743
2	0.74	0.99	0.86	0.926
3	0.45	0.07	0.08	0.301

find $\vec{g}=\begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ g_{12} \\ g_{13} \\ g_{23} \end{bmatrix}$ such

that E^2 is minimized

O_1: $C_g(O_1) = (0.81 - 0.68)\, g_3 + (0.68 - 0.53)\, g_{13} + (0.53) = 0.13 g_3 + 0.15\, g_{13} + 0.53$

O_2: $C_g(O_2) = (0.99 - 0.86)\, g_2 + (0.86 - 0.74)\, g_{23} + (0.74) = 0.13 g_2 + 0.12\, g_{23} + 0.74$

O_3: $C_g(O_3) = (0.45 - 0.08)\, g_1 + (0.08 - 0.07)\, g_{13} + (0.07) = 0.37 g_3 + 0.01\, g_{13} + 0.07$

Hence

$$\vec{v}_1 = \begin{bmatrix} 0 \\ 0 \\ 0.13 \\ 0 \\ 0.15 \\ 0 \end{bmatrix}, \vec{v}_2 = \begin{bmatrix} 0 \\ 0.13 \\ 0 \\ 0 \\ 0 \\ 0.12 \end{bmatrix}, \vec{v}_3 = \begin{bmatrix} 0.37 \\ 0 \\ 0 \\ 0 \\ 0.01 \\ 0 \end{bmatrix}$$

And

$$\mathbf{D} = \sum_{j=1}^{3} \vec{v}_j \vec{v}_j^T = \begin{bmatrix} 0.14 & 0 & 0 & 0 & 0.004 & 0 \\ 0 & 0.017 & 0 & 0 & 0 & 0.016 \\ 0 & 0 & 0.017 & 0 & 0.02 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0.004 & 0 & 0.02 & 0 & 0.023 & 0 \\ 0 & 0.016 & 0 & 0 & 0 & 0.014 \end{bmatrix} \quad \vec{v} = \sum_{j=1}^{3} \left[h\left(o_j; x_{(n)}\right) - \alpha_j \right] \vec{v}_j^{\,T} = \begin{bmatrix} -0.086 \\ -0.024 \\ -0.028 \\ 0 \\ -0.034 \\ -0.022 \end{bmatrix}$$

$$\mathbf{A}=\begin{bmatrix} 1 & 0 & 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad \vec{b}=\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

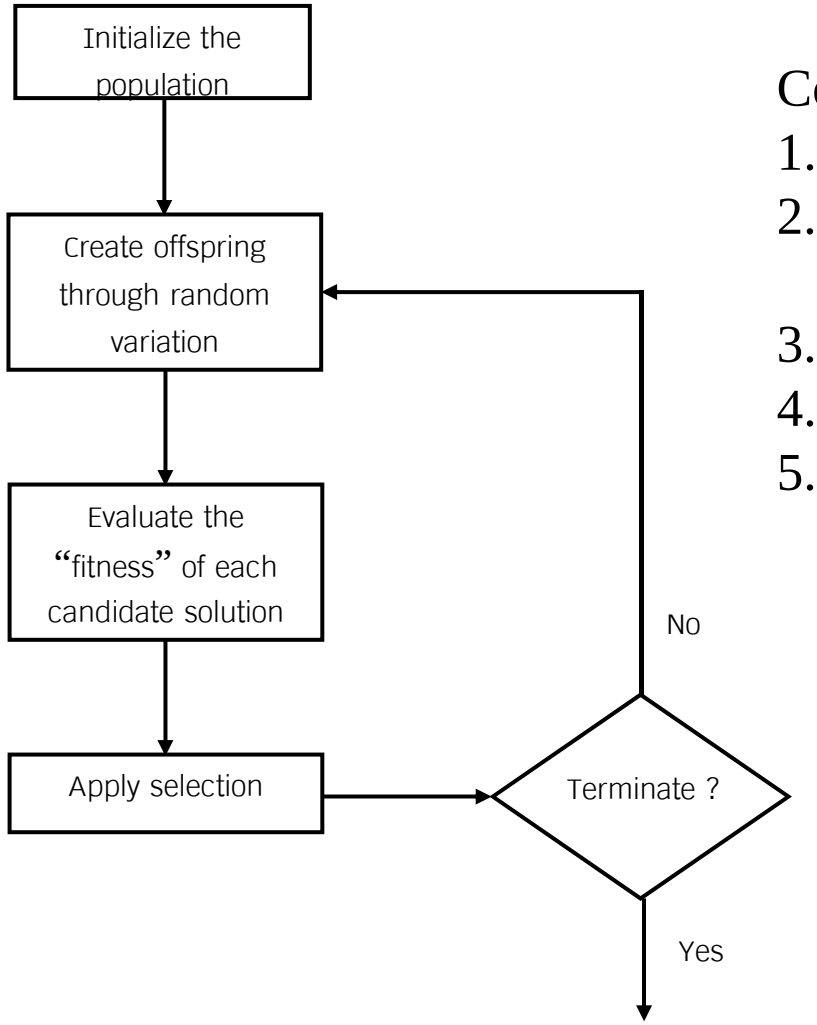
Now we can compute

$$\underset{\vec{g}}{\text{minimize}} \frac{1}{2}\vec{g}^T\mathbf{D}\vec{g}+\vec{v}^T\vec{g}$$

subject to $\mathbf{A}\,\vec{g} \le \vec{b}$ and $\vec{0} \le \vec{g} \le \vec{1}$

Introduction to Evolutionary Computation

Evolutionary computing



Component

1. Chromosome encoding
2. Fitness value or survival strength of individual
3. Initial population
4. Selection operator
5. Reproduction operator

Without mutation→population tends to converge to a homogeneous state where individuals vary only slightly from each other

Introduction to Genetic Algorithm

- Genetic Algorithm

  Algo

  1. Set t=0
  2. Initial population $P(t)$
  3. Calculate each individual fitness value
  4. While not converge do
     1. Select individual to P^1 (intermediate population)→Mating Pool (MP)
     2. Select from MP to mate → P^2 → mutate chromosome in P^2 → P^3
     3. Select chromosome in P^3 and $P(t)$ for replacement→ $P(t+1)$
     4. Set t= t+1
  5. End while

If ($|P^3|$)=N then P^3 become $P(t+1)$ else if ($|P^3|$)<N select q missing chromosome from $P(t)$ or P^1

- Example

Chromosome *A*, *B*, *C*, *D* → each with 8 genes

Fitness function: number of 1 in the string

chromosome label	chromosome string	fitness value
A	00000110	2
B	11101110	6
C	00100000	1
D	00110100	3
Average fitness value		12/4

$P(t)$

Crossover *B* and *D* at crossing site: 1 get *E* and *F* And *B* and *C* are copied →

chromosome label	chromosome string	fitness value
B	11101110	6
C	00100000	1
E	10110100	4
F	01101110	5

B is mutated at 1st bit
E is mutated at 6th bit

$P(t+1)$

chromosome label	chromosome string	fitness value
B′	01101110	5
C	00100000	1
E′	10110000	3
F	01101110	5
Average fitness value		14/4

- Initial population
  - random gene values of each gene in each chromosome→uniform representation of the entire search space
  - Small population → small part of search space, time complexity is low, need more generations to converge →EA force to explore a larger search space by increasing the rate of mutation
  - Large population→ large area of search space, less generation to converge, time complexity is increased
- Selection operation
  - Selection techniques exist
    - Explicit fitness remapping→fitness values is mapped into a new range, e.g., normalization to [0,1]
    - Implicit fitness remapping→use actual fitness values for selection several selection operators

– Random selection

random → no reference to fitness→ each individual has an equal chance of being selected

– Proportional selection

$$P(t)=\left\{x^1,x^2,\ldots,x^N\right\} \qquad \text{total fitness } F=\sum_{i=1}^{N} f\left(x^i\right)$$

where $f(x^i)$→fitness

value of chromosome x^i →selection probability of x^i

$$p_i=\frac{f\left(x^i\right)}{F} \quad \text{for } i=1,2,\ldots,N$$

Expected number of copied of x^i → $n_i = Np_i$ for $i = 1, 2, \ldots, N$

Or

$$n_i=N\frac{f\left(x^i\right)}{F}=\frac{f\left(x^i\right)}{\bar{f}} \quad \text{for } i=1,2,\ldots,N$$

– Algorithm

For $i=1$ to N

Calculate

$$q_i = \sum_{k=1}^{i} p_k \quad \text{for } i = 1,2,\ldots,N$$

End for

For $i=1$ to N

random $\xi \in [0,1]$ → uniform distribution

If $0 \le \xi \le q_1$, chromosome x^1 is selected

If $q_{i-1} \le \xi \le q_i$ for $i=1,2,\ldots N$ then chromosome x^i *is* selected

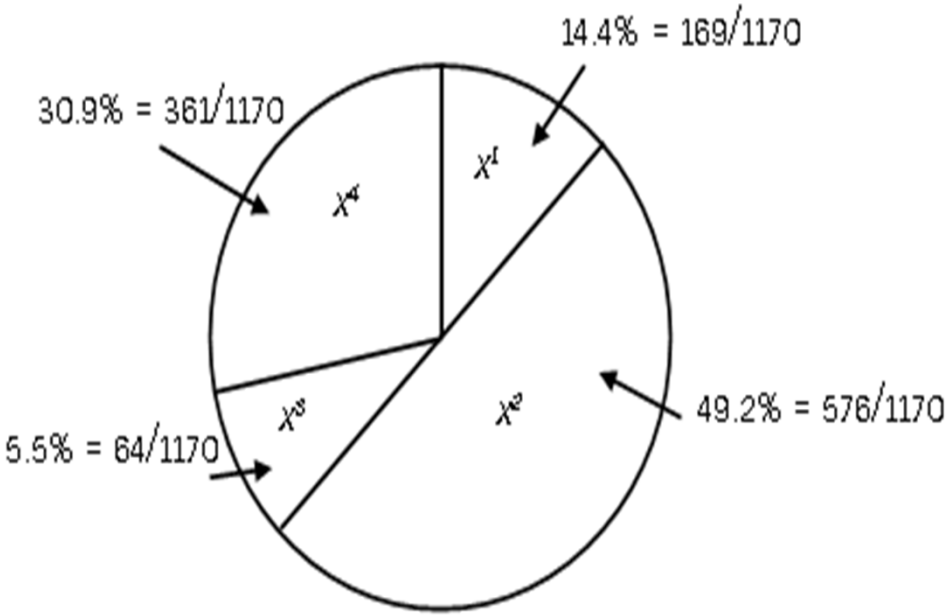
End for

- Example

$$\max_x f(x) = \max_x x^2$$ with x is varied between 0-31,

assume random ξ 4 times with the value of 0.8, 0.5, 0.1 and 0.6

No	Chromosome	x (in real number)	f(x)	p_i	q_i	N_i	Number of copies
1	01101	13	169	0.14	0.14	0.58	1
2	11000	24	576	0.49	0.63	1.97	2
3	01000	8	64	0.06	0.69	0.22	0
4	10011	19	361	0.31	1.00	1.23	1
		Total	1170	1.00			
		Average	293	0.25			
		maximum	576	0.49			



Crossover → $x^1$and x^2 at crossover site 4

$x^3$and x^4 at crossover site 2

Mutation probability is 0.001→ there are 20 bits→ expected number of bit undergoes mutation is 20(0.001) = 0.02 bits → suppose no mutation

Hence, New population will be

No	Chromosome (x)	mate	Crossover site	New population	X (in real number)	f(x)
1	01101	2	4	01100	12	144
2	11000	1	4	11001	25	625
3	11000	4	2	11011	27	729
4	10011	3	2	10000	16	256
					Total	1754
					Average	439
					maximum	729

- Disadvantage
  - Premature convergence
  - Slow convergence

Introduction to Genetic Algorithm

- Selection Based on Scaling and Ranking Mechanisms
  - Static Scaling Mechanisms
    - linear scaling

$$S(x) = ax + b$$

$$tf(x) = S(f(x))$$

Evaluation function: $eval(x) = tf(x) = S(f(x)) = af(x) + b$

    - Power law scaling

$$eval(x) = tf(x) = S(f(x)) = (f(x))^u$$

- Logarithmic scaling

$$tf(x) = S(f(x)) = e^{\frac{f(x)}{T}}$$

T is controlling the selective pressure →>decrease during search process

Selection pressure→degree to which highly fit individuals are allowed to produce offspring (copies) in the next generation

Low values of selection pressure→ provide each individual in the population with a reasonable selection prob.

High values of selection pressure→strongly favor the best individuals in the population→ population diversity decrease quickly

- Dynamic Scaling mechanism
  - Sigma truncation

$$T(x)=\begin{cases} x & \text{if } x>0 \\ o & \text{else} \end{cases}$$

$$S(x)=T\left(x-\left(m_t-c\sigma_t\right)\right)$$

$$eval(x)=tf(x)=S\left(f(x)\right)=T\left(f(x)-\left(m_t-c\sigma_t\right)\right)$$

$$eval(x)=\begin{cases} f(x)-\left(m_t-c\sigma_t\right) & \text{if } f(x)>\left(m_t-c\sigma_t\right) \\ 0 \quad \text{OR} & \text{if } f(x)\le\left(m_t-c\sigma_t\right) \end{cases}$$

$$eval(x)=\begin{cases} 1+\dfrac{f(x)-m_t}{\sigma_t} & \text{if } \sigma_t\neq 0 \\ 0 & \text{if } \sigma_t=0 \end{cases}$$

- Window scaling

$$eval(x) = f(x) - \min_{y \in P(t)} f(y)$$

OR

$$eval(x) = f(x) - \min_{y \in W} f(y)$$

where W is the set of all individuals contained in the last g generation, $g \geq 1$

- Fitness remapping for minimization problem

$$tf(x) = M - f(x)$$

where M is the upperbound of the objective function

$$eval(x(t)) = tf(x(t)) = M_t - f(x(t))$$

where $x(t) \in P(t)$ and M_t is max value of function f found so far or max value of f found within generation $P(t)$

OR

$$eval(x(t)) = \frac{1}{K + f(x(t)) - f_{\min}(t)}$$

where K is positive constant and $f_{min}(t)$ is the minimum observed value of function f up to time t

OR

$$eval(x) = \frac{1}{K + f(x)}$$

- Rank-based Selection

  let number of individuals in each generation → N ($|P(t)|=N$)

  – Linear Ranking selection

  selection probability of x^i

$$p_i = \frac{1}{N}\left[Min + (Max - Min)\frac{r_i - 1}{N-1} \right]$$

  where r_i → rank of individual x^i (ranked in increasing order)

  expected value of offspring of x^i

$$n_i = Np_i$$

And $\sum_{i=1}^{N} n_i = \sum_{i=1}^{N} Np_i = N$ Then $\sum_{i=1}^{N} Np_i = N$

$$\sum_{i=1}^{N} N\frac{1}{N}\left[Min + (Max - Min)\frac{r_i - 1}{N-1} \right] = \sum_{i=1}^{N} Min + \sum_{i=1}^{N} (Max - Min)\frac{r_i - 1}{N-1} = N$$

$$NMin + \frac{1}{N-1}(Max - Min)\sum_{i=1}^{N}(r_i - 1) = N$$

But, since $\sum_{i=1}^{N}(r_i - 1) = \sum_{i=1}^{N}(i-1) = \sum_{i=0}^{N-1} i = \frac{N(N-1)}{2}$

Hence, $Min + \frac{1}{2}(Max - Min) = 1$

$$Min = 2 - Max \quad \text{OR} \quad Max + Min = 2$$

Since $Min \geq 0$ and $Min \leq Max \rightarrow Max \leq 2$ and $Max \geq 1 \rightarrow 1 \leq Max \leq 2$

Hence $p_N = \frac{Max}{N}$ and $p_1 = \frac{2 - Max}{N} = \frac{Min}{N}$

Alternatively,

$$p_i = \frac{1}{N}\left[Max + 2(Max - 1)\left(1 - \frac{r_i - 1}{N - 1}\right)\right]$$

- nonlinear ranking selection

  Selection probability will be

$$p_i = \frac{1}{c}\left(1 - e^{1-r_i}\right)$$

- SUS Algorithm

1. *ptr* = Rand(); */*random within* [0,1] → *(uniform distribution)*/*
2. *For (sum=i=0; i<N;i++)*
   1. *For (sum +=n_i; sum>ptr; ptr++)*
   2. *Select(i);*
   3. *End For*
3. *End For*

guarantee that x^i will reproduce at least $\lfloor n_i \rfloor$ but not more than $\lceil n_i \rceil$

- Example

Chomosome	$f(x)=x^2$	r_1	p_i	n_i
$x^1=10$	100	1	$\frac{2-1.2}{3}=0.27$	3(0.27)=0.81
$x^2=15$	225	2	$\frac{1}{3}\left[0.8+(1.2-0.8)\frac{2-1}{3-1}\right]=0.33$	3(0.33)=0.99
$x^3=20$	400	3	0.4	3(0.4)=1.2

Suppose ptr is 0.8→ the selection will be x^1, x^3 and x^3 , respectively

- Tournament selection

  Similar to rank selection in terms of selection pressure → but more efficient

  – Binary tournament (deterministic)
    - 2 individuals directly compete for selection
    - With or without reinsertion of the competing individual into the original population
    - based on direct pairwise comparison of individuals
      – 2 chromosomes are chosen at random from the population
      – The fitness of the chosen chromosomes is calculated
      – The most successful chromosome (the winner) is selected→ to intermediate population

  – Probability tournament
      – Step 3→ random $R \in [0,1]$ if $R<p$ then the fitter of the two is selected; otherwise the less fit is selected

      where p is preselected or p varied with time →

$$p(t)=p_0 e^{-ct} \quad \text{or} \quad p(t)=p_0 \frac{c}{1+\ln t}$$ where $p_0 \in (0,1)$ and c is positive constant

– Boltzmann tournament

x → current solution, *y*→ alternative solution →binary tournament between *x* and *y* with

$$p(x) = \frac{1}{1 + e^{\frac{f(x) - f(y)}{T}}}$$

Where *T* is the temperature→ reduced in a predefined manner in successive iteration

The winner of the trial will be the current solution in the next trial

- Elitism→ selection of a set of individual from the current generation to survive to the next generation

The fraction of new individual at each generation has been called generation gap

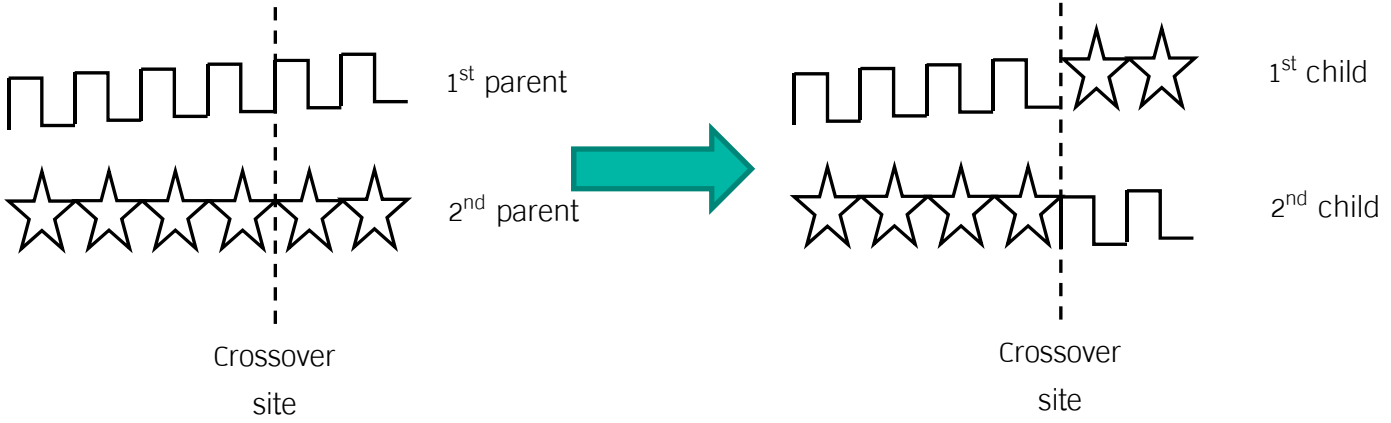
- Recombination operator

  cross over: $C: X \times X \rightarrow X \times X$ → $C(x,y)=(x',y')$

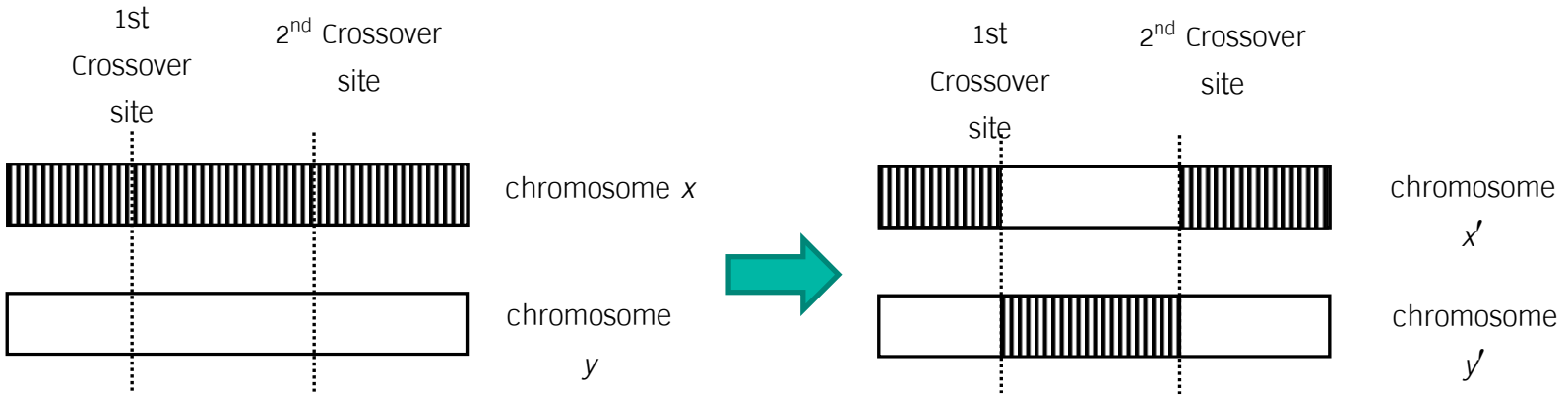
  – One-point crossover

  $x = x_1\, x_2\, x_3 \ldots\, x_L$ and $y = y_1\, y_2\, y_3 \ldots\, y_L$ after cross over

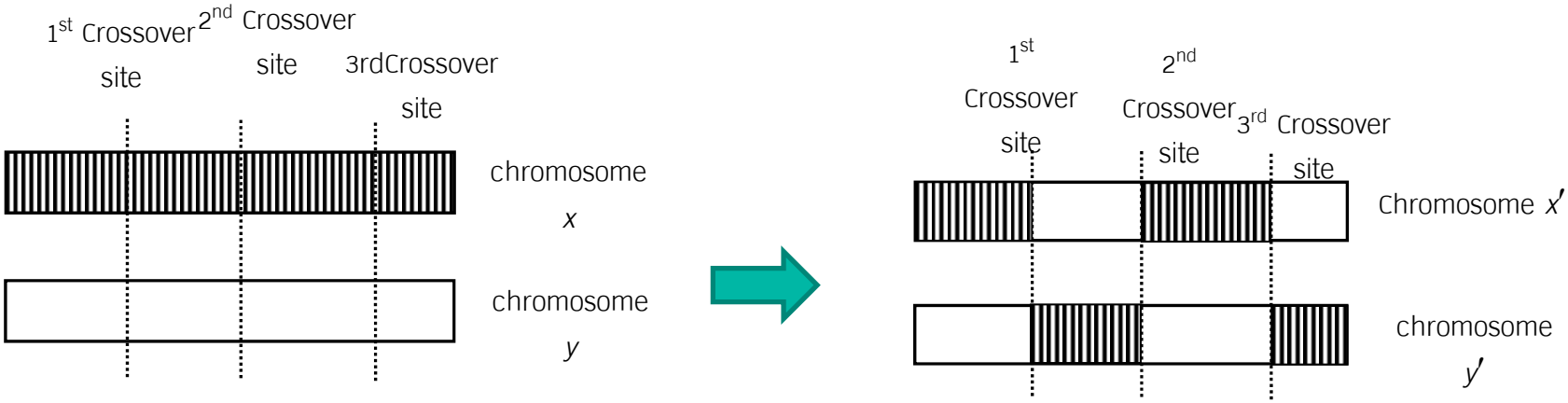
  $x' = x_1\, x_2\, x_3 \ldots\, x_{k-1}\, x_k\, y_{k+1} \ldots\, y_L$ and $y' = y_1\, y_2\, y_3 \ldots\, y_{k-1}\, y_k\, x_{k+1} \ldots\, x_L$



– Two-point cross over



– *N*-point cross over



– *N*-point cross over algorithm

P1: for each chromosome from intermediate population (P^1)

P1.1: random $q \in [0,1]$ →uniform distribution

P1.2: if $q < p_c$ (where p_c is mating probability) then respective chromosome is considered mating →go to mating pool else not mating

P2: let m^* →number of chromosome selected at P1

if m^* is even then select pair chromosome (random)

if m^* is odd then discard or select chromosome from P^1

P3: Cross over is applied on pairs in P2

P3.1: for each pair, generate k ($\leq k \leq L$)

and random crossing site $k_1, k_2, \ldots, k_N$ for ($N \geq 2$)

P3.2: descendants obtained at P3.1 become potential members of $P(t+1)$ → candidate for mutation

P3.3: parents of newly generated chromosome are discarded from P^1

P3.4: chromosomes left in P^1 are added to $P(t+1)$

– Punctuated crossover
  - Record the crossover points in the chromosome
  - If a certain crossover point has generated a very poor offspring this crossover point will not be considered further
  - If the fitness of the descendant is good, the crossover point is maintained as active
  - Cross-over point self adjust according to the previous dynamics of the search process

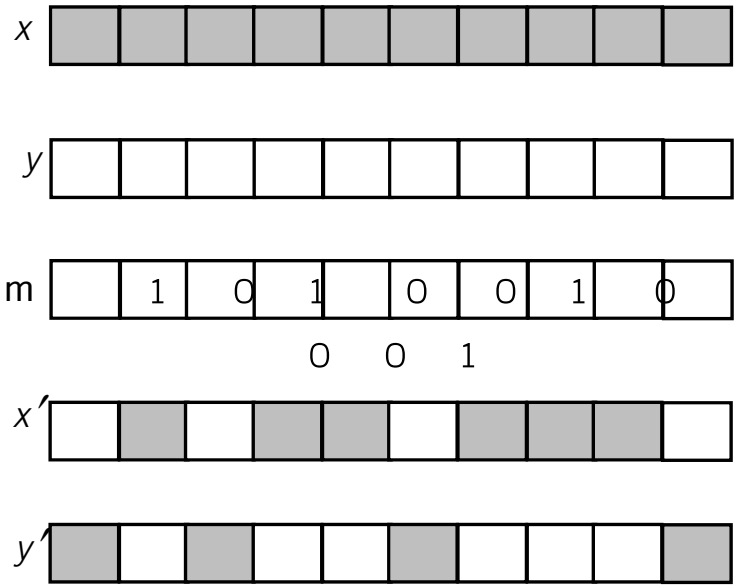
$C = x_1\, x_1 \ldots\, x_L k_1\, k_1 \ldots\, k_L$ when $k_i = 1$ → active crossover point, $k_i = 0$ → not a crossover point

During initialization probability of generating $k_i = 1$ is small

Set of crossover point used in the recombination of 2 chromosomes is the union of the parents

– Uniform crossover

1. *Set* $m_i = 0$ *for* $i = 1,2,\ldots,L$
2. *For each i*
   1. *random* $\xi \sim U(0,1)$
   2. *If* $\xi \le p_{at\_i}$ *then* $m_i = 1$ *where* p_{at_i} *is the crossover probability at* i^{th} *position*



- Mutation

  Average number of position that will undergo a mutation→ $B = NLp_m$
  where p_m →mutation probability ($p_m \in [0.001, 0.01]$),
  N→ number of chromosomes,
  L→length of chromosome

  Algorithm

  For each chromosome และ For each position of the chromosome

  1. *Generate a random number q* from *U*(0,1)
  2. *If q <p_m invert that position*
     If q≥p_m do not mutate

  Weak mutation

  step 2 changed to if $q < p_m$ then 1 of 0 or 1 is chosen at random

– Non-uniform mutation

$$p_m(t) = p_m e^{-\beta t} \qquad \text{where } \beta \geq 1$$

OR

$$p_m(t) = \left(\frac{\alpha}{\delta}\right)^{\frac{1}{2}} \left(\frac{e^{\frac{-\beta t}{2}}}{L^2 N^{\frac{1}{}}} \right)$$

where α, β, δ → Real-valued positive parameter

OR

$$p_m(t) = \frac{1}{2 + \frac{L-2}{T} t}$$ where $0 \le t \le T \rightarrow p_m(0) = 0.5$ and $p_m(T) = \frac{1}{L}$

OR

$$p_m(x) = \frac{1}{2(f(x)+1) - L}$$

- Inversion operator

  Acts on a single chromosome. It inverts the values between 2 position of a chromosome. The 2 positions are chosen at random

- Simple genetic algorithm

1. Set $t = 0$

Initialize ($P(0)$) → random

a selection mechanism is chosen and if necessary a scaling mechanism

2. The chromosome of $P(t)$ are evaluated using the fitness function. The most successful individual from $P(t)$is kept.

3. Selection operator is applied N times

Selected chromosome make up an intermediate population P^1 (having *N members as well) representing candidate from the mating pool (MP). Some chromosomes from* $P(t)$ may have more copies in P^1 *while some others may have none*

4. Crossover operator is applied to the chromosomes from MP

Newly generated chromosomes form P^2

Parents of the chromosomes obtained through crossover are discarded from P^1

Chromosomes that remained in P^1 become member of P^2

5. Mutation operator is applied to P^2 *outcome is the new generation*($P(t+1)$)

6. set $t = t+1$

if $t < T$ (T is the maximum number of generation) then go to step 2 otherwise stop

- Solution → best member of the last population or best individual in all generation

- Can involve inversion or other types of operators

- Example

$$\max_{x_1,x_2} f(x_1,x_2) \quad \text{where} \quad f(x_1,x_2)=\frac{1}{1+x_1^2+x_2^2}$$

chromosome	x_1	x_2	Fitness value
C_0	–1	2	0.167
C_1	–2	3	0.007
C_2	1.5	0.0	0.31
C_3	0.5	–1.0	0.44

C_3, C_3, C_2 and C_0 are selected to P^1

(C_3, C_2) and (C_3, C_0)

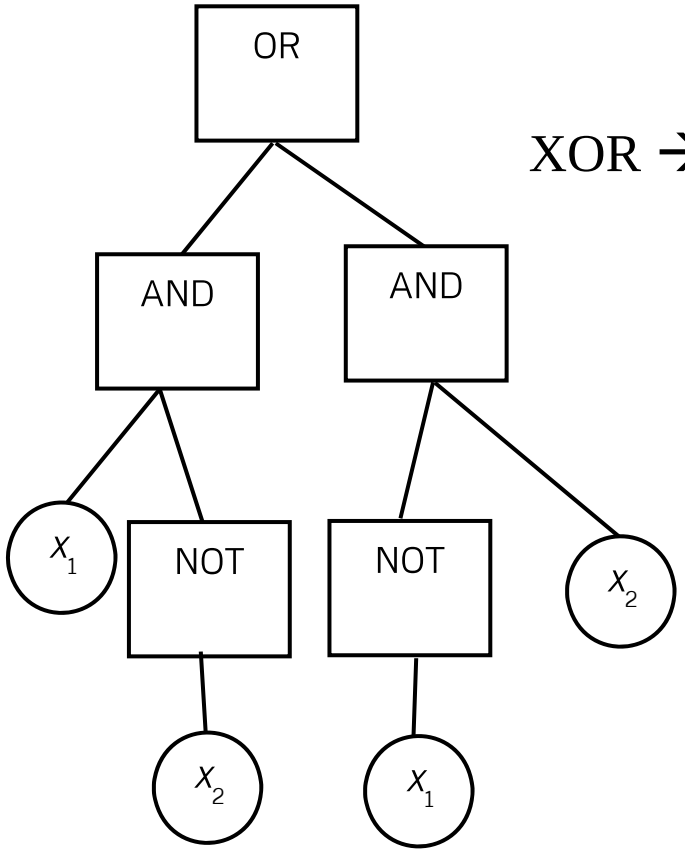
chromosome	x_1	x_2	Fitness value
C_0	0.6	0	0.8
C_1	1.6	–1.0	0.24
C_2	0.6	2.0	0.19
C_3	–1.0	–1.0	0.33

Introduction to Genetic Programming

- Each individual/chromosome → 1 computer program represented using a tree structure
- Terminal set → specifies all variables and constant → elements of the terminal set form the leaf modes
- Function set → all functions that can be applied to the elements of the terminal set may be mathematical arithmetic and/or Boolean function or if-then-else→form the non-leaf nodes.

- Example

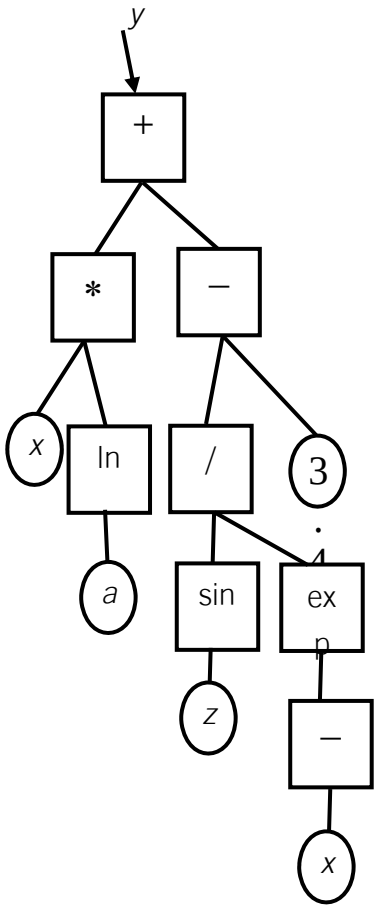
function {AND, OR, NOT} and terminal set $\{x_1, x_2\}$ where $x_1, x_2 \in \{0,1\}$



XOR → (x_1 AND NOT x_2) OR (NOT x_1 AND x_2)

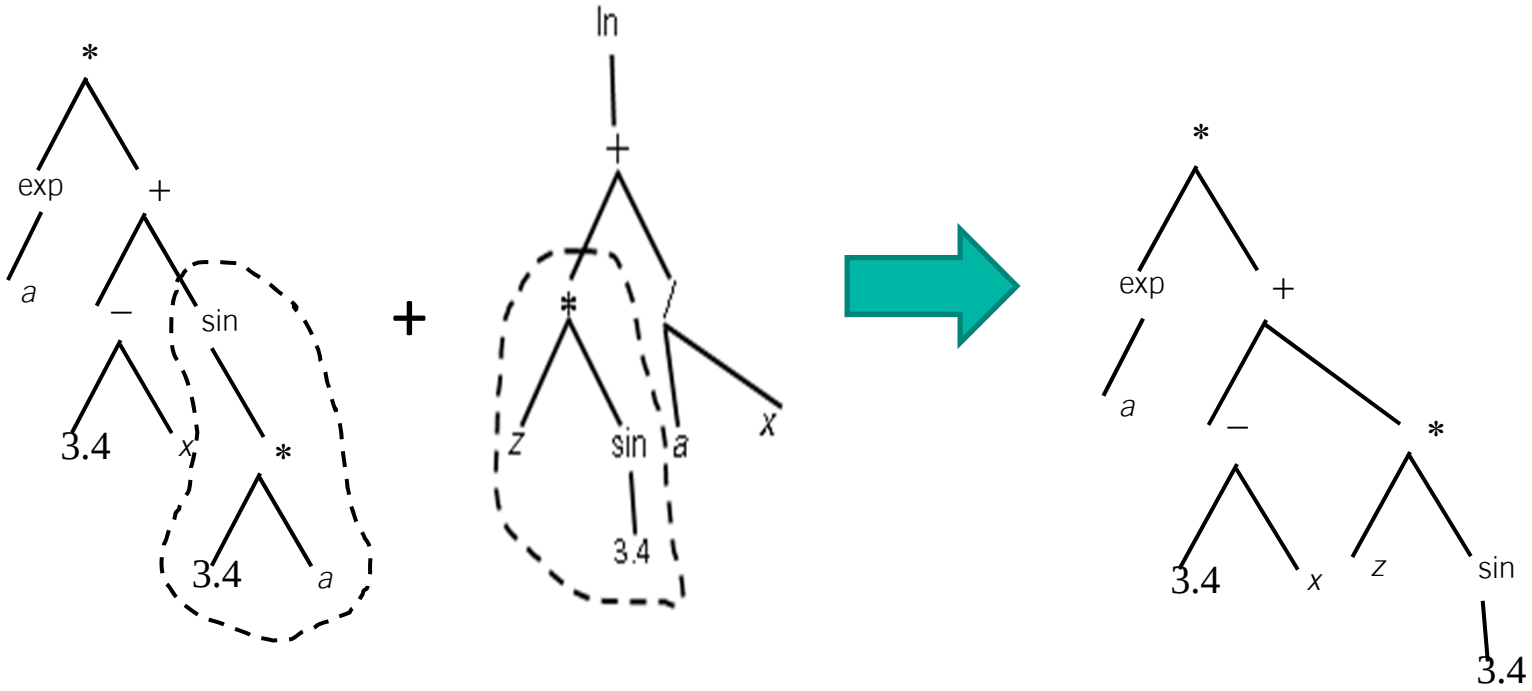
- Example

Function $\{-,+,*,/\}$ and terminal set $\{a,x,z,3.4\}$ where $a,x,z \in \Re$

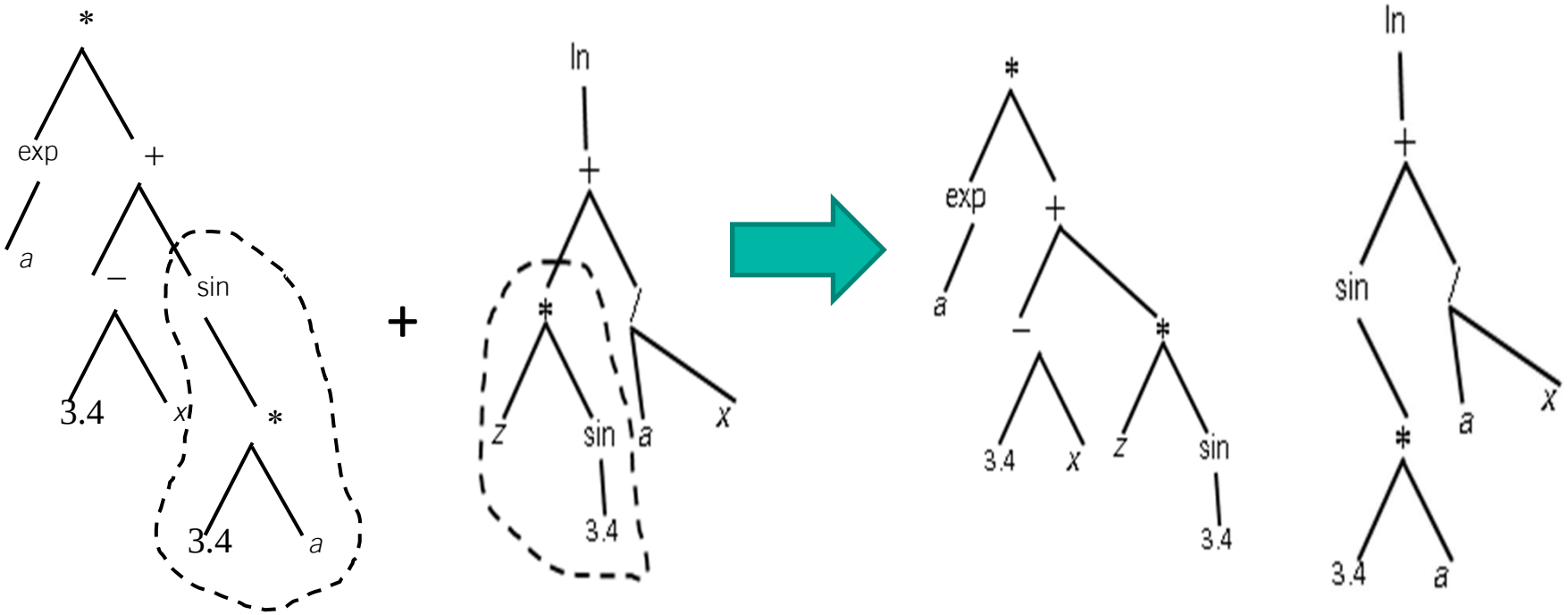


$$y = x * \ln(a) + \sin(z) / \exp(-x) - 3.4$$

Crossover→ 2 parents generating 1 child



Crossover→ 2 parents generating 2 children



- Mutation→choose a random point in a tree and replacing the subtree beneath that point by a randomly generated subtree
- Calculate the fitness of each program → by running it on a set of "fitness case"

Evolutionary programming

- Algorithm

1. *Let current generation be* $t=0$
2. *Initialize all parameters*
3. *Initialize population* $C(0)$ *with* n_s *individual*
4. *For each individual* $\mathbf{x}_i(t) \in C(t)$
   calculate fitness value $f(\mathbf{x}_i(t))$
   End
5. *While no convergence*
   - *For each individual* $\mathbf{x}_i(t) \in C(t)$
     mutate $\mathbf{x}_i(t)$ to produce offspring $\mathbf{x}_i'(t)$
     calculate fitness value $(f(\mathbf{x}_i'(t)))$
     add $\mathbf{x}_i'(t)$ *in offspring set* $C'(t)$
     End
   - *Select new population* $C(t+1)$ *from* $C(t) \cup C'(t)$ *using selection operator*
   - $t = t+1$

   End

Coevolution

- Predator-pray competitive coevolution→
  - Insects that eat plant
    - To survive plant needs to evolve mechanism to defend itself from the insects, and
    - The insects need the plants as food source to survive.
  - Inverse fitness interaction between 2 species→win for 1 species →failure for the other
- Symbiosis
  - Different species cooperate instead of compete→success in 1 species improve the survival strength of the other species→positive feedback among the species that take part in this cooperating process

- Coevolutonary algorithm (CoEA)
  - Competitive coevolution
    - Competition
    - Amensalism
  - Cooperative coevolution
    - Mutualism
    - Commensalism
    - Parasitism
- Competitive coevolution (CCE)
  - Evolve 2 population simultaneously
    - individual in 1 population → solution (evolve to solve as many test cases as possible)→ fitness ∝ the number of test cases solved by the solution
    - individual in another population → test case (evolve to present an incrementally increasing level of difficult to the solution individuals)→fitness is inversely proportional to the number of strategies that solve it

- Relative fitness function→express the performance of individual in one population is comparison with individuals in another population
  - Which individual from the competing population is used
  - Exactly how these competing individuals are used to compute the relative fitness
- Sampling scheme
  - All versub all sampling
  - Random sampling
  - Tournament sampling
  - All versus best sampling
  - Shared sampling
- Relative fitness function → $f(C_1.x_i)$
  - Simple function

  2 population → C_1 and C_2 →aim to calculate the relative fitness of each individual $C_1.x_i$ of C_1

  Sample of individuals is taken from C_2 → S

  sum $C_1.x_i$ = count the number of individuals in S that $C_1.x_i$ win

  $f(C_1.x_i)$ = sum $C_1.x_i$

- Fitness sharing

sim $C_1.x_i$ = similarities of $C_1.x_i$ with all the other individuals in that population
→ e.g., the number of individuals that also beats individual from C_2 samples

$$f\left(C_1.x_i\right) = \frac{sumC_1.x_i}{simC_1.x_i}$$

- Competitive fitness sharing

$$f\left(C_1.x_i\right) = \sum_{l=1}^{C_2.n_s} \frac{1}{C_1.n_l}$$

Where $C_2.x_1$, $C_2.x_2$,…, $C_2.x_{C2.ns}$,→ form C_2 and $C_1.n_l$ = total number of individuals in C_1 that defeat individual $C_2.x_l$

- Tournament fitness
  - Binary tournament determine a relative fitness ranking
  - Tournament fitness results in a tournament tree with root element as the best individual
    - Each level→ 2 opponent are randomly selected from that level and the best of the 2 advances to next level
  - If odd number→ a single individual from the level moves to the next level
  - After tournament → use any selection operator

- Hall of frame
  - At each generation the best individual of a population is stored in that population's hall of frames
  - May have limited size
  - New individual →inserted in hall of frame will replace the worst or the oldest individual
  - Individual from 1 population → complete against a sample of the current opponent population and its hall of frame
  - Prevents overspecialization

- Competitive coevolution with 2 population Algo
  - Initialize C_1 (solution), C_2 (test) population
  - While stopping condition(s) not true do
    - For each $C_1.x_i$ for i=1,…, $C_1.n_s$ do
      - Select a sample of opponents from C_2
      - Evaluate the relative fitness of $C_1.x_i$ w.r.t. this sample
    - End
    - For each $C_2.x_i$ for i=1,…, $C_2.n_s$ do
      - Select a sample of opponents from C_1
      - Evaluate the relative fitness of $C_2.x_i$ w.r.t. this sample
    - End
    - Evolve population C_1 for 1 generation
    - Evolve population C_2 for 1 generation
  - end
  - Select the best individual from C_1 as solution

- Competitive coevolution with 1 population Algo
  - Initialize 1 population C
  - While stopping condition(s) not true do
    - For each $C.x_i$ for i=1,…, $C.n_s$ do
      - Select a sample of opponents from C (exclude $C.x_i$)
      - Evaluate the relative fitness of $C.x_i$ w.r.t. the sample
    - End
    - Evolve population C for 1 generation
  - end
  - Select the best individual from C as solution

- Cooperative coevolution
  - Similar to divide and conquer
  - Only mutualism here
  - Individual from different species or subpopulation have to cooperate in some way to solve a global task
  - Fitness depends on that individual's ability to collaborate with individual from other species
  - Ex. For an n_x dimensional problem→n_x subpopulation (1 for each)
    - each subpopulation → responsible for optimizing one of the parameters of problem and no subpopulation can form a complete solution by itself
    - Merge representative from each subpopulation
    - Consider j subpopulation (C_j)→ each $C_j.x_i$ perform a single collaboration with the best individual from each other subpopulation → complete solution
    - Credit assign to $C_j.x_i$ → fitness of the complete solution
    - Evolve independently from one another→ separate population is used to evolve each subcomponent using some EA

Sansanee Auephanwiriyakul, *Senior Member IEEE*

Introduction to Swarm Intelligence

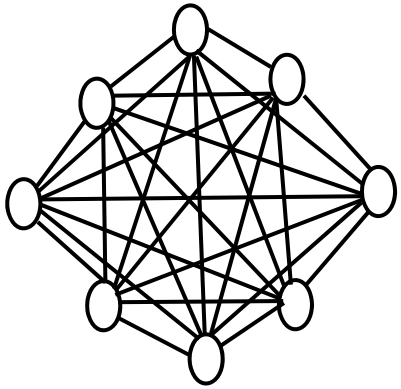
Swarm Intelligence (SI)

- Structured collection of interacting organisms (or agents)
- Individual organism
  - Simple in structure but their collective behavior can become complex
- Tight coupling between individual behavior and the behabior of the entire swarm
- Social interaction→direct (through visual audio, chemical context), or indirect (occur when one changes the environment and the other individuals respond to the new environment)
- 2 concepts
  - Self-organization
    - Positive feedback

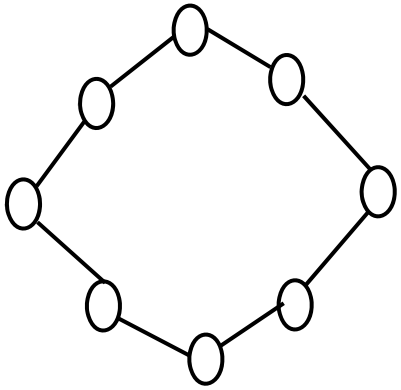
Swarm Intelligence

- 2 concepts
  - Self-organization
    - Positive feedback
    - Negative feedback
    - Fluctuation
    - Multiple interaction
  - Division of labor
- SI→no one is in charge, no one is giving orders, which the swarm individuals should obey or follow
- Social networkstructureof swarm→form an integral part of the existence of that swarm→provide the communication channel, able to self-organize to form the optimal nest structures

- Partical swarm optimization (PSO)
  - Simulation of social behavior of birds within flock
  - Particles are “flown” through hyperdimensional search space
  - Neighborhood topology
    - Star topology→first version PSO →gbest

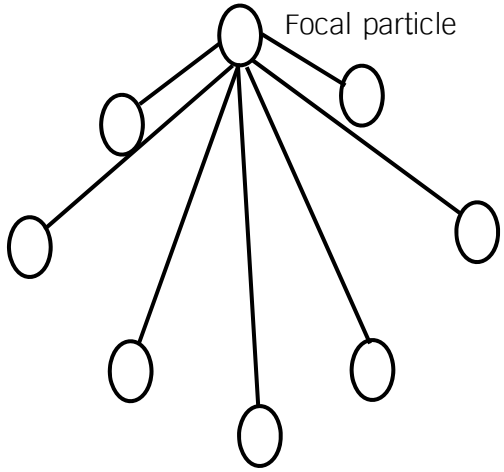


- Ring topology



n=2

- Wheel topology



- Individual best (pbest)
  - Initialize the swarm $P(t)$ at t=0 of particles such that the position ($\mathbf{x}_i(t)$) of i^{th} particle ($P_i \in P(t)$) is random within the hyperspace
  - Evaluate the performance F of each particle using $\mathbf{x}_i(t)$
  - Compare the performance of each individual to its best performance
    - *If* $F(\mathbf{x}_i(t)) < pbest_i$

      $pbest_i = F(\mathbf{x}_i(t))$

      $x_{pbest_i}(t) = x_i(t)$

    - End
  - Change the velocity vector of each particle

    $v_i(t) = v_i(t-1) + \rho\left(x_{pbest_i} - x_i(t)\right)$ where ρ is positive random number

  - Move each particle to a new position $x_i(t) = x_i(t-1) + v_i(t)$
  - Set t=t+1
  - Repeat until converge (go to step 2)

- Global best (gbest)
  - Initialize the swarm $P(t)$ at t=0 of particles such that the position ($\mathbf{x}_i(t)$) of i^{th} particle ($P_i \in P(t)$) is random within the hyperspace
  - Evaluate the performance F of each particle using $\mathbf{x}_i(t)$
  - Compare the performance of each individual to its best performance
    - *If* $F(\mathbf{x}_i(t)) < pbest_i$

      $pbest_i = F(\mathbf{x}_i(t))$

      $\mathbf{x}_{pbest_i}(t) = \mathbf{x}_i(t)$

    - End
  - Compare the performance of each individual to its best performance
    - *If* $F(\mathbf{x}_i(t)) < gbest$

      $gbest = F(\mathbf{x}_i(t))$

      $\mathbf{x}_{gbest}(t) = \mathbf{x}_i(t)$

      Where ρ_1 and ρ_2 are random number → $\rho_1 = r_1 c_1$ and $\rho_2 = r_2 c_2$ with r_1 and $r_2 \sim U(0,1)$, $c_1 + c_2 \leq 4$

- Change the velocity vector of each particle

$$v_i(t) = v_i(t-1) + \rho_1\left(x_{pbest_i} - x_i(t)\right) + \rho_2\left(x_{gbest} - x_i(t)\right)$$

- Move each particle to a new position

$$x_i(t) = x_i(t-1) + v_i(t)$$

- Set $t=t+1$
- Repeat until converge (go to step 2)

- Local best (lbest)
  - Initialize the swarm $P(t)$ at t=0 of particles such that the position ($\mathbf{x}_i(t)$) of i^{th} particle ($P_i \in P(t)$) is random within the hyperspace
  - Evaluate the performance F of each particle using $\mathbf{x}_i(t)$
  - Compare the performance of each individual to its best performance
    - *If* $F(\mathbf{x}_i(t)) < pbest_i$

      $pbest_i = F(\mathbf{x}_i(t))$

      $$\mathbf{x}_{pbest_i}(t) = \mathbf{x}_i(t)$$

    - End
  - Compare the performance of each individual to its best performance
    - *If* $F(\mathbf{x}_i(t)) < lbest$

      $lbest = F(\mathbf{x}_i(t))$

      $$\mathbf{x}_{lbest}(t) = \mathbf{x}_i(t)$$

      Where ρ_1 and ρ_2 are random number → $\rho_1 = r_1 c_1$ and $\rho_2 = r_2 c_2$ with r_1 and $r_2 \sim U(0,1)$, $c_1 + c_2 \leq 4$

- Change the velocity vector of each particle

$$v_i(t) = v_i(t-1) + \rho_1\left(x_{pbest_i} - x_i(t)\right) + \rho_2\left(x_{lbest} - x_i(t)\right)$$

- Move each particle to a new position

$$x_i(t) = x_i(t-1) + v_i(t)$$

- Set $t=t+1$
- Repeat until converge (go to step 2)

- Fitness calculation→can be objective function→ same idea as in GA
- Convergence→executed for a fixed number of iteration or velocity changes are close to 0 for all particles
- PSO system parameter
  - Dimension of problem
  - Number of individual
  - Upper limit of ρ
  - Upper limit on the maximum velocity (v_{max})

    If $v_{ij}(t) > v_{max}$ then $v_{ij}(t) = v_{max}$ and If $v_{ij}(t) < -v_{max}$ then $v_{ij}(t) = -v_{max}$

    where $v_{ij}(t)$ is a velocity of ith particle at jth dimension at time t

    OR if do not want to use v_{max}

$$v_i(t) = \kappa\left(v_i(t-1) + \rho_1\left(x_{pbest_i} - x_i(t)\right) + \rho_2\left(x_{gbest} - x_i(t)\right)\right)$$

Where

$$\kappa = 1 - \frac{1}{\rho} + \frac{\sqrt{\left|\rho^2 - 4\rho\right|}}{2} \qquad \text{and } \rho = \rho_1 + \rho_2 > 4.0$$

- Neighborhood size
  - gbest →simply lbest with the entire swarm as the neighborhood
  - gbest →more susceptible to local minima
- Inertia weight

$$v_i(t) = \phi v_i(t-1) + \rho_1\left(x_{pbest_i} - x_i(t)\right) + \rho_2\left(x_{gbest} - x_i(t)\right)$$

Where ϕ is inertia weight ($\phi \leq 1$) and

$$\phi > \frac{1}{2}\left(c_1 + c_2\right) - 1$$

– Modified PSO

- Using selection→ perform selection before velocity update

Algo

- for each particle in the swarm, score the performance of that particle w.r.t a random selected group of *k* particles
- Rank the particles according to these performance scores
- Select the top half of the particle and copy their current position onto that of the bottom half of the swarm without changing the personal best values of the bottom half of the swarm

- Breeding PSO

Algorithm

- Calculate the particle velocities and new positions
- For each particle assign a breeding probability p_b
- Select 2 particles assume P_a and P_b and produce 2 offsprings using

$$\mathbf{x}_a(t+1) = r_1\mathbf{x}_a(t) + (1-r_1)\mathbf{x}_b(t)$$

$$\mathbf{x}_b(t+1) = r_2\mathbf{x}_b(t) + (1-r_2)\mathbf{x}_a(t)$$

$$\mathbf{v}_a(t+1) = \frac{\mathbf{v}_a(t)+\mathbf{v}_b(t)}{\left\|\mathbf{v}_a(t)+\mathbf{v}_b(t)\right\|}\left\|\mathbf{v}_a(t)\right\|$$

$$\mathbf{v}_b(t+1) = \frac{\mathbf{v}_a(t)+\mathbf{v}_b(t)}{\left\|\mathbf{v}_a(t)+\mathbf{v}_b(t)\right\|}\left\|\mathbf{v}_b(t)\right\|$$

Where r_1, $r_2 \sim U(0,1)$

- Set personal best position of each particle involved in the breeding process to its current position

- Neighborhood topology
  - Use of indices
  - Use spatial neighbor

    P_b is said to be in the neighborhood of P_a if

$$\frac{\|\mathbf{x}_a - \mathbf{x}_b\|}{d_{max}} < \xi$$

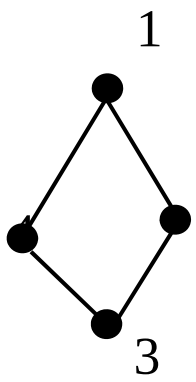
where d_{max} is the largest distance between any 2 particles and

$$\xi = \frac{3t + 0.6t_{max}}{t_{max}}$$

- Ant Colony Optimization
  - tasks
    - Reproduction
    - Defense
    - Foor collection
    - Brood care
    - Nest brooming
    - Nest building
  - Natural stigmergy
    - Lack of central coordination
    - Communication and coordination among individuals in a colony are based on local modifications of the environment
    - Positive feedback

- Shortest path problem →traveling salesman problem (TSP)
- Virtual pheromone
- Negative feedback→avoid premature feedback
- Time scale→not too large and not too short
- Ant System (AS)
  - Find a closed tour of minimal length connected *n* given cities

$$d_{ij} = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2}$$ distance between city *i* and city *j*



Graph (*N*,*E*) With 4 cities and they are all connected

- Tabu list and J_i^k set of all cities that ant *k* is has to visit when it is on city *i*

– AS algorithm

For every edge (i,j)

$\tau_{ij}(0) = \tau_0$ *#initialize pheromone*

End For

For $k = 1$ *to* m *#for each ant k*

Place ant k on a randomly chosen city

End For

Set T^+ *and* L^+

For $t = 1$ *to* t_{max}

For $k = 1$ *to* m

build tour $T^k(t)$ ***by applying*** $n-1$ *times the following steps: choose the next city* j when the current city is city i using the transition rule

$$p_{ij}^k(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in J_i^k}[\tau_{il}(t)]^{\alpha}[\eta_{il}]^{\beta}} & \text{if } j \in J_i^k \\ 0 & \text{else} \end{cases}$$

Where α *and* β *are adjustable and visibility* $\eta_{ij} = \dfrac{1}{d_{ij}}$

End For

For $k = 1$ *to* m

compute length $L^k(t)$ *of tour* $T^k(t)$ *produced by ant* k

End For

If an improved tour is found then

update T^+ *and* L^+

End If

For every edge (i,j)

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) + e\Delta\tau_{ij}^{e}(t)$$

where

$$\Delta\tau_{ij}^{e}(t) = \begin{cases} \dfrac{Q}{L^+} & \text{if } (i,j) \in T^+ \\ 0 & \text{if } (i,j) \notin T^+ \end{cases}$$

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t)$$

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{Q}{L^k(t)} & \text{if } (i,j) \in T^k(t) \\ 0 & \text{if } (i,j) \notin T^k(t) \end{cases}$$

End For

End For

- Ant colony system (ACS)
  - Different transition rule
  - Different pheromone trail update rule
  - Use of local updates of pheromone trail to favor exploration
  - Use of candidate list→ cl closest cities ordered by increasing distance→ ant first restricts the choice of the next city to those in the list, and consider other cities if all in the list have been visited

For every edge (*i*,*j*)

$\tau_{ij}(0) = \tau_0$ *#initialize pheromone*

End For

For $k = 1$ *to m #for each ant k*

Place ant k on a randomly chosen city

End For

Set T^+ *and* L^+

For $t = 1$ *to* t_{max}

For $k = 1$ *to m*

build tour $T^k(t)$ *by applying* $n-1$ *times the following steps:*

if exits at least one city $j \in$ *candidate list then choose the next cityt* $j \in J_i^k$ *among cI citiy in the candicate list as*

$$j = \begin{cases} \underset{u \in J_i^k}{\arg\max}\left\{[\tau_{iu}(t)][\eta_{iu}]^\beta\right\} & \text{if } q \le q_0 \\ J & \text{if } q > q_0 \end{cases} \quad \text{where } q \sim U(0,1),\ 0 \le q_0 \le 1$$

where $J \in J_i^k$ is chosen according to $p_{iJ}^k(t) = \dfrac{[\tau_{iJ}(t)][\eta_{iJ}]^\beta}{\sum_{l \in J_i^k}[\tau_{il}(t)][\eta_{il}]^\beta}$

else choose the closest city $j \in J_i^k$

end if

after each transition ant k *applies the local update rule* $\tau_{ij}(t) \leftarrow (1-\rho)\tau_{ij}(t) + \rho\tau_0$ *where* $\tau_0 = (nL_{nn})^{-1}$

End For

For k= 1 to m

compute length $L^k(t)$ *of tour* $T^k(t)$ *produced by ant k*

End For

If an improved tour is found then

update T^+ *and* L^+

End If

For every edge (i,j)

$$\tau_{ij}(t) \leftarrow (1-\rho)\tau_{ij}(t) + \rho\Delta\tau_{ij}(t)$$

where

$$\Delta\tau_{ij}(t) = \frac{1}{L^+}$$

End For

End For