MỤC LỤC

BẢNG KÝ HIỆU VIẾT TẮT/GIẢI NGHĨA

VIẾT TẮT/ TÊN RIÊNG	NGHĨA THEO TIẾNG ANH	DỊCH RA TIẾNG VIỆT/GIẢI NGHĨA
ADALINE	Adaptive Linear Element	Phần tử (nơ ron) tuyến tính thích nghi, tên mạng nơ ron do Widrow đề xuất năm 1960
A/D	Analog to Digital Conveter	Bộ chuyển đổi tương tự/số
AI	Artificial Intelligence	Trí tuệ nhân tạo
ANFIS	Adaptive Neuro Fuzzy Inference System	Hệ thống nơ ron-mờ thích nghi
BAM	Bidirectional Associative Memory	Bộ nhớ liên kết hai chiều: tên mạng nơ ron hồi quy hai lớp (Roselblatt)
BMU	Best Matching Unit	Đơn vị (nơ ron) khớp tốt nhất
Boltzmann	Boltzmann	Mạng nơ ron lấy tên Boltzmann
CAM	Content Addressable Memory	Bộ nhớ nội dung được địa chỉ hoá.
CBIS	Computer-Based Information Systems	Hệ thống thông tin dựa trên máy tính
GA	Genetic Algorithm	Giải thuật di truyền
CLIPS	C Language Integrated Production System	Hệ thống sản xuất (nhân quả) tích hợp theo ngôn ngữ C
Hopfield	Hopfield	Tên mạng nơ ron truy hồi (mạng rời rạc, 1982; liên tục, 1984) do Hopfield đề xuất
KBS	Knowledge Base System	Hệ thống dự trên tri thức
LMS	Least Mean Square	Trung bình bình phương nhỏ nhất:
NFS	Neuro-Fuzzy Systems	Các hệ thống nơ ron-mờ
NST	(Chromosome)	Nhiễm sắc thể
MISO	Multi Input Single Output	Hệ thống nhiều đầu vào một đầu ra
OAV	Object Atribute Value	Giá trị thuộc tính đối tượng
Perceptron	Perceptron	Bộ cảm nhận: tên mạng nơ ron truyền thẳng do Rosenblatt đề xuất năm 1960
VLSI	Very Large Scale Integration	Mạch tích hợp mật độ cao.
RBF	Radian Basic Function	Hàm xuyên tâm
SISO	Single Input Single Output	Hệ thống một đầu vào một đầu ra
SVM	Support Vector Machine	Máy vec tơ hỗ trợ

LỜI NÓI ĐẦU

Giáo trình "Các hệ thống dựa trên tri thức" là một trong những hệ thống của chuyên ngành Hệ thống Thông tin. Giáo trình này là những hệ thống ứng dụng cụ thể và mở rộng của lĩnh vực Trí tuệ Nhân tạo. Nói cách khác, các hệ thống dựa trên trí thức được xây dựng dựa trên một nguyên lý nào đó của trí tuệ nhân tạo để xây dựng một hệ thống ứng dụng riêng

Các hệ thống dựa tri thức có nguồn gốc xuất xứ từ một số hệ thống như hệ chuyên gia. Hệ thống sử dụng các tính toán mềm cũng là những hệ gần gũi với các hệ thông dựa trên tri thức chủ yếu gồm hệ mờ, mạng nơ ron, giải thuật di truyền và lập trình tiến hóa, hệ thống dựa theo xác suất. Hệ thống dựa theo trí thức có quy mô rộng hơn miễn là có thể hiện tri thức trong đó.

Giáo trình gồm sáu chương. Chương một mang tính giới thiệu, cho một số khái niệm cơ bản, phân loại các hệ dựa tri thức, một số công cụ hỗ trợ thực hiện hệ thống dựa tri thức. Những khái niệm đã được giới thiệu trong trí tuệ nhân tạo, để tránh trùng lặp, giáo trình không nhắc lại nhiều. Chương hai, giới thiệu thuật toán mang tính xác suất điển hình. Một số hệ thống khác có tính xác suất như hệ mờ, nhưng sử dụng nhiều nguyên tắc khác như tập hợp, logic, tính toán mờ được tách thành một hệ riêng. Chương ba là hệ mờ, chủ yếu trình bày có tính hệ thống và quy trình hướng tới giải bài toán, không quá đi sâu lý thuyết. Chương bốn đề cập tới mạng nơ ron gồm các cấu trúc và luật học và một vài ứng dụng của các mạng nơ ron cụ thể. Chương năm giới thiệu cơ bản về thuyết tiến hóa và giải thuật di truyền. Chương sáu nêu một số hệ lai của hệ mờ với nơ ron, mờ với hệ tiến hóa, hệ tiến hóa với mạng nơ ron. Một số các hệ thống khác của hệ thống dựa theo trí thức không giới thiệu do khuôn khổ giáo trình có hạn.

Những vấn đề của các hệ thống dựa trên trí thức là khá tiên tiến và đang trong tiến trình phát triển, hoàn thiện. Nhiều quan điểm phân loại hay định nghĩa còn đang được bàn luận. Do vậy, giáo trình không tránh khỏi thiếu sót hoặc chưa đủ cập nhật. Mong được đóng góp từ tất cả các bạn đồng nghiệp và độc giả.

CHỦ BIÊN

CHƯƠNG 1:

CƠ BẢN VỀ HỆ THỐNG DỰA TRÊN TRI THỨC

1.1 Tri thức và hệ cơ sở tri thức

1.1.1 Khái niệm về tri thức

Tri thức (*Knowdge)* là sự hiểu biết bằng lý thuyết hay thực tế về một đối tượng, sự việc, hoàn cảnh, sự kiện hay một lĩnh vực nhất định. Tri thức là tổng của tất cả những hiểu biết hiện thời, là một khái niệm trừu tượng trong đời thường. *Chuyên gia (ExpertS)* là những người tập hợp được nhiều tri thức hơn các người bình thường khác. Để có thể đưa tri thức vào máy tính (giống như ta đã mô tả dữ liệu cho máy tính để máy tính có thể giúp ta giải quyết các bài toán), khái niệm tri thức trừu tượng đó càn phải phải được mô tả cụ thể. Trong các cách cụ thể hóa tri thức, người ta thông nhất chia tri thức làm 3 phần, đó là:

i) các sự kiện (*Events* hay *Facts*);

ii) các mối quan hệ, quy tắc, quy luật liên quan giữa các sự kiện hay gọi tắt là luật (*Rules*) giữa các sự kiện đó;

iii) tri thức có tính heuristic. Heuristic xuất phát từ thuật ngữ ơ-ric-ca là một thuật ngữ khó dịch ra tiếng Việt; nó hàm ý được rút ra từ kinh nghiệm, từ suy diễn mang tính may rủi (không hoàn toàn chính xác, nhưng dùng tốt theo một số nghĩa nào đó). Heuristic tạm dịch là tìm ra, phát hiện ra (*to Find hay to Discovery*)

Ví dụ về sự kiện. Giả sử có hai sự kiện "trời mưa" (ký hiệu (hay gán) là biến *A*); sự kiện "đất ướt" (ký hiệu (hay gán) là biến *B*). Những hiện tượng đó, con người khi trưởng thành có thể nhận thức được, gọi là các sự kiện. Các sự kiện tương đương với dữ liệu mà ta đã biết và là dạng đơn giản nhất của trí thức. Nhưng nó chưa hoàn toàn đủ để gọi là là tri thức, nó tương đương với dữ kiện (hay dữ liệu). Ở mức tri thức, con người còn rút ra các mối liên quan giữa các sự kiện qua đúc rút kinh nghiệm, qua thực tế. Giữa các sự kiện đó, con người muốn hiểu sâu hơn, tìm hiểu giữa các sự kiện đó có mối quan hệ nào không?

Mối quan hệ giữa các sự kiện đó có tồn tại không? Gắn hai sự kiên vừa nêu, ta có thể thấy: khi có "trời mưa" dẫn tới (kéo theo) sự kiện "đất ướt", giữa chúng có mối liên hệ, mối liên hệ đó là $A \rightarrow B$. Đây là mối quan hệ mà chúng ta có thể mô tả bằng logic mệnh đề. Ta cũng có thể mô tả $A \rightarrow B$ bằng quy tắc hay là luật *IF…THEN (NẾU…THÌ)* như sau:

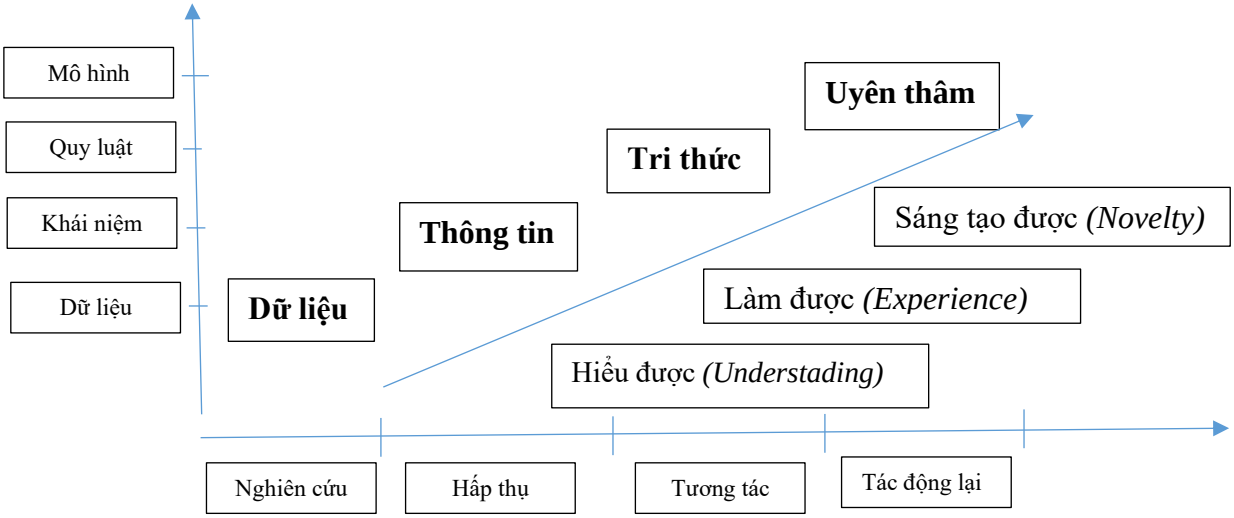
NẾU "trời mưa" THÌ "đất ướt"	NẾU A THÌ B	IF "trời mưa" THEN "đất ướt"	IF A THEN B

Trong ngôn ngữ lập trình, "*IF…THEN"* là một cấu trúc. Trong trí tuệ nhân tạo chúng ta gọi là nó là luật "*IF…THEN"* hay luật nhân quả, hay luật sinh (tiếng Anh: *Production Rule*). Các mối quan hệ này chính là các quy luật (*Rule*) thể hiện mối liên hệ giữa các sự kiện.

1.1.2 Tháp dữ liệu và các hệ thống dựa trên máy tính

Hệ thống dựa trên tri thức (Knowledge-Based Systems)

Các hệ thống thông minh nhân tạo sử dụng các kỹ thuật của trí tuệ nhân tạo, thông qua các kỹ thuật đó, hệ thống thông minh có khả năng giải được các bài toán ở các lĩnh vực riêng của mình. Những hệ thống như vậy sử dụng kiến thức của một hoặc nhiều chuyên gia gọi là hệ thống dựa trên tri thức *(Knowledge-Based Systems)* hay hệ chuyên gia *(Expert System)* [1]. Các hệ thống giải bài toán trên máy tính truyền thống từ trước tới nay dựa trên dữ liệu *(Data)* và/hoặc thông tin *(Information)* được gọi là *các hệ thống thông tin dựa trên máy tính (Computer-Based Information Systems: CBIS)*

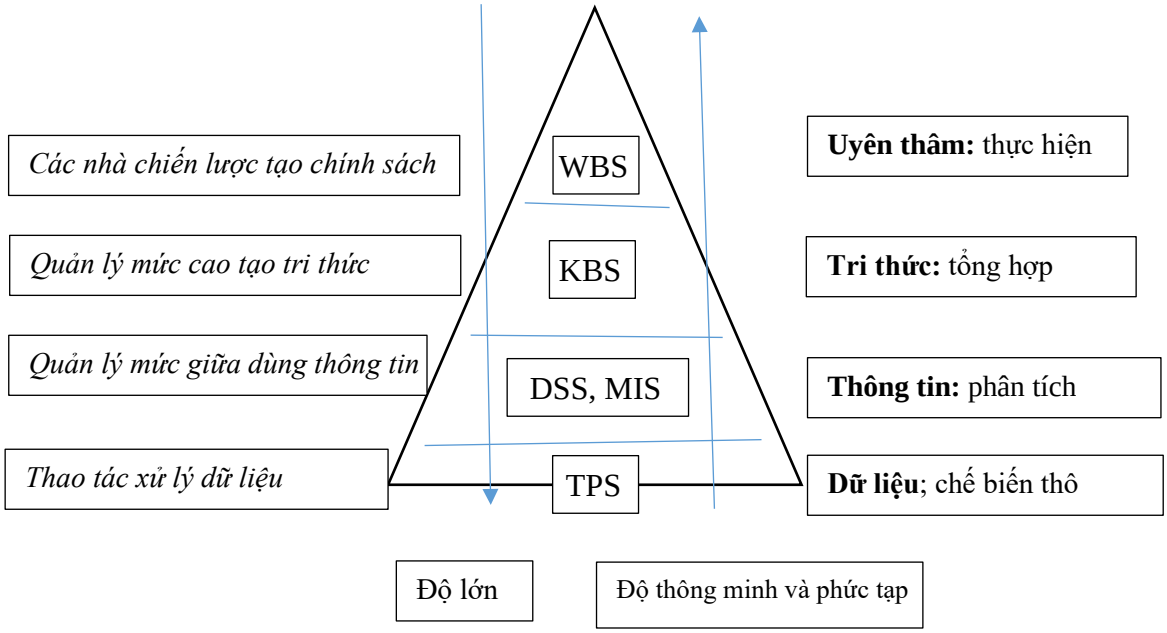


Hình 1.1. Biểu đồ mô tả từ dữ liệu đến trí tuệ

Hình 1.1 mô tả đồ thị phát triển trí tuệ từ dữ liệu, thông tin, tri thức đến thông minh (hay uyên thâm) và mối quan hệ giữ bốn khái niệm này. Khi thực hiện các hoạt động: nghiên cứu, tiếp thu (hấp thụ), tương tác (trao đổi), phản ảnh (tương tác lại) được mô tả

trên trục *x* con người đạt được (kết quả) hiểu biết, thực hành được, tiến tới làm mới và sáng tạo như một sản phẩm của quá trình tư duy. Trục *y* có thể coi là các mức (hội tụ)
mô tả: từ dữ liệu (nguyên liệu thô), được xử lý (xác định được hay không xác định được từ dữ liệu để có thông tin) thành các khái niệm, sau đó rút ra thành quy luật (luật) và tiếp theo là mô hình mô tả.

Hình 1.2 cho thấy sự phát triển của tháp (quản lý) dữ liệu. Mức thấp nhất: mức thao tác dữ liệu hoạt động với môi trường sử dụng các thủ tục (chương trình), ví dụ *hệ thông xử lý giao tác (Transaction Processing System: TPS)* nhằm tạo ra các chương trình con giao tác với các hoạt động (kinh doanh) cơ bản.



Hình 1.2. Tháp quản lý dữ liệu, thông tin, tri thức và trí tuệ (uyên thâm)

Các thông tin từ mức thao tác được phân tích, chế biến, tạo báo cáo và giúp các nhà quản lý ra quyết định *(Decision Support System: DSS)* ở mức thứ hai (mức quản lý trung gian: *Management Information System: MIS*).

Ở mức cao (quản lý), từ các kết quả đã tiến hành qua quyết định ở mức hai, kết hợp với các định mức, luật lệ để khái quát hóa, chuyển thông tin thành trí thức. Các hệ thống thực hiện chức năng này là *các hệ dựa trên tri thức (Knowledge-Based Systems: KBS)* hoặc *các hệ dựa trên kiến thức uyên thâm (Wisdom-Based Systems).*

1.3 Hệ cơ sở tri thức là gì?

Hệ CSTT là hệ thống dựa trên tri thức (một tập hợp các tri thức và tập các quan hệ), cho phép mô hình hóa các tri thức của chuyên gia, dùng tri thức này để giải quyết vấn đề phức tạp cùng lĩnh vực.

Hai yếu tố quan trọng trong hệ cơ sở tri thức là: sự kiện và lập luận hay suy diễn)

Sự kiện	*Lập luận (suy diễn)*
Sự kiện 1	Lập luận 1
Sự kiện 2	Lập luận 2
……	.....
Sự kiện n	Lập luận m

1.2 Biểu diễn tri thức

Tri thức có thể phân làm hai nhóm chính:

- Mô tả tri thức theo sự kiện (*Factual Knowledge Representation*)
  - Hằng (*Constant)*
  - Biến (*Variables)*
  - Hàm (*Functions*)
  - Vị từ (*Predicates*)
  - Các công thức (*Well-Formed Formulas*)
  - Logic vị từ cấp 1 (*First Order Logic*)
- Mô tả tri thức theo thủ tục (*Procedural Knowledge Representation*)

Trong chương trình trí tuệ nhân tạo, ta đã biết một số phương pháp mô tả tri thức theo sự kiện như:

- Phương pháp kinh điển: mô tả tri thức bằng logic hình thức:

  Logic mệnh đề. Ví dụ: A→ B; Logic vị từ (xem giáo trình trí tuệ nhân tạo).

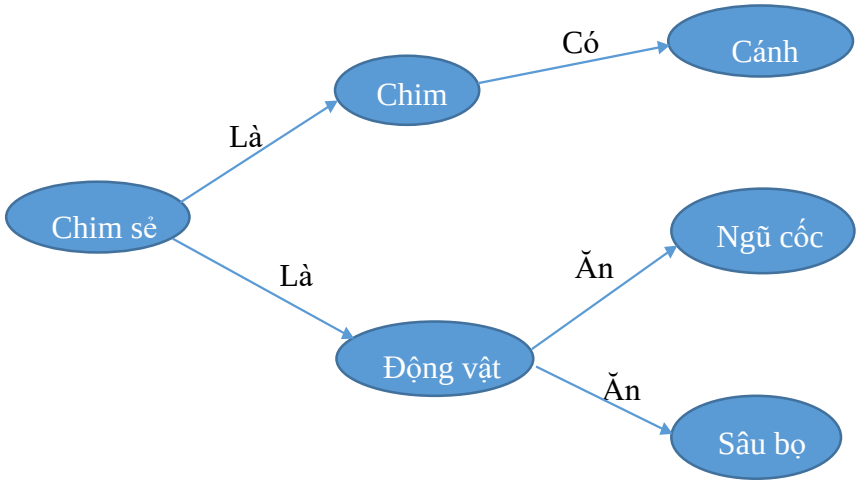
- Phương pháp mô tả bằng luật IF…THEN hay luật nhân quả
- Mô tả tri thức bằng cặp ba: OAV (*Object Atribute Value*);
- Mô tả tri thức băng khung (*Frame*)
- Mô tả tri thức bằng mạng ngữ nghĩa.

Đây là một phương pháp mô tả có nhiều ứng dụng và thành công; biến thể của nó là các mạng tính toán, mạng Bayes, mạng nơ-ron nhân tạo… Bởi vậy, chúng ta sẽ tìm hiểu về cách mô tả này (như là mở rộng của giáo trình trí tuệ nhân tạo). Ở đây, phương pháp mô tả dùng mạng ngữ nghĩa có nhiều liên quan đến các phần sau.

Mô tả tri thức bằng mạng ngữ nghĩa

Mạng ngữ nghĩa có liên quan đến các vấn đề của hệ dựa trí thức như mạng tính toán, mạng nơ-ron… Những mạng đó có thể coi là trường hợp riêng của mạng ngữ nghĩa.

Định nghĩa 1: Mạng ngữ nghĩa là sự mở rộng và phát triển từ mô tả bộ ba OAV. Mạng ngữ nghĩa là mạng (gồm nút và cung *G={V, U}*, trong đó nút *V* được gán một ngữ nghĩa nhất định, *U* là mối liên hệ giữa các nút. Ví dụ đơn giản về một mạng ngữ nghĩa (hình 1.3):



Hình 1.3. Mô tả mạng ngữ nghĩa (Sematic Net)

Mạng ngữ nghĩa có khả năng mở rộng và phát triển (suy rộng ra nó có khả năng suy diễn và phát triển tri thức). Mặt khác, mạng ngữ nghĩa cũng có những ngoại lệ. Ví dụ về ngoại lệ như "chim biết bay", nhưng chim đà điểu, chim cánh cụt không không biết bay. Mặt khác, chim đà điểu, chim cánh cụt vẫn thuộc họ chim. Mặt trái của vấn đề mở rộng của mạng ngữ nghĩa nói chung hay suy diễn nói riêng là không hoàn toàn chính xác (nói cách khác, nó có tính xác suất hay có độ chắc chắn mà ta sẽ đề cập ở các phần sau).

Khái niệm mạng tính toán

Định nghĩa 1: Mạng tính toán là trường hợp riêng của mạng ngữ nghĩa. Như ta biết, mạng (*ký hiệu G)* là tập hợp của tập các Nút (*ký hiệu V)* và tập các cung (*ký hiệu U)*. Ở đây cần phân biệt: trong mạng máy tính (*Computer Net*) nút của nó là máy tính. Mạng tính toán (Computing Net): nút của nó là hàm và biến, trong đó để phân biệt, người ta thường dùng nút dạng chữ nhật để ký hiệu hàm; nút tròn mô tả biến. Có nhiều định nghĩa khác nhau về mạng tính toán tùy theo loại hình mô tả.

Định nghĩa 2: Mạng tính toán là một dạng đặc biệt của mạng ngữ nghĩa, trong đó các nút được mô tả bởi: i) Hàm: Ký hiệu nút bằng một hình dạng (ví dụ dạng hình chữ nhật); ii)

Biến: ký hiệu nút bằng hình dạng khác (ví dụ dạng hình tròn); cung mô tả mối liên hệ giữa các nút hàm và các nút biến.

Ví dụ: Cho tam giác ABC với tập các biến $M=\{a, b, c, \alpha, \beta, \gamma, h_a, h_b, h_c, p, S, r, R...\}$, gồm các tham số cơ bản của tam giác và tập các hàm $F=\{f_1, f_2, f_3, \ldots, f_m\}$ mô tả mối quan hệ giữa các biến trong tam giác. Ta có một số định nghĩa sau.

Định nghĩa 3: Mạng tính toán là 1 tập *{M, F}*. Trong trường hợp tổng quát, có thể viết:

$$M = \{x_1, x_2, \ldots, x_n\}, F = \{f_1, f_2, \ldots, f_m\}.$$

trong đó, x_i là hàm thứ *i i=1..n;* f_j là hàm thứ *j, j= 1..m.*

Bài toán A → B: Cho mạng tính toán *{M, F}, A, B* ⊆ *M*; Cho *A = {a, b, α}; B={p, S}*. Tìm lời giải $D = \{f_1, f_2, f_3, \ldots, f_k\}$ để có thể tìm được *B* khi cho *A*.

Với mỗi $f \in F$, ta kí hiệu *M(f)* là tập các biến có liên hệ trong quan hệ *f*. Dĩ nhiên, *M(f)* là một tập con của *M: M(f)* ⊆ *M*.

1.2.1 Các vấn đề trên mạng tính toán

Cho một mạng tính toán *(M, F), M* là tập các biến và *F* là tập các quan hệ. Giả sử có một tập biến $A \subseteq M$ được xác định (tức là tập gồm các biến đã biết trước giá trị) và *B* là một tập biến bất kì trong *M*. Khi đó, *A* được gọi là giả thiết, *B* được gọi là mục tiêu tính toán (hay tập biến cần tính) của bài toán. Trường hợp tập *B* chỉ gồm một phần tử *b*, ta viết tắt bài toán trên là A→b.

Định nghĩa 4: Bài toán *A→B* được gọi là giải được khi có thể tính được giá trị các biến thuộc *B* xuất phát từ giả thiết *A*. Ta nói rằng một dãy quan hệ $\{f_1, f_2, \ldots, f_k\} \subseteq F$ là một lời giải của bài toán *A→B*.

Lời giải $\{f_1, f_2, \ldots, f_k\}$ được gọi là lời giải tốt nếu không thể bỏ bớt một số bước tính toán trong quá trình giải, tức là không thể bỏ bớt một số quan hệ trong lời giải.

Lời giải được gọi là lời giải tối ưu khi nó có một số bước tính toán ít nhất trong số các lời giải tốt.

1.2.2 Ví dụ minh họa mạng tính toán. Thuật toán vết dầu loang

Bài toán: Cho ΔABC, tập *{M, F}, tập A={a, b, α}*. Tìm tập *B={p, S}*

Bước 1: Xây dựng mạng tính toán.

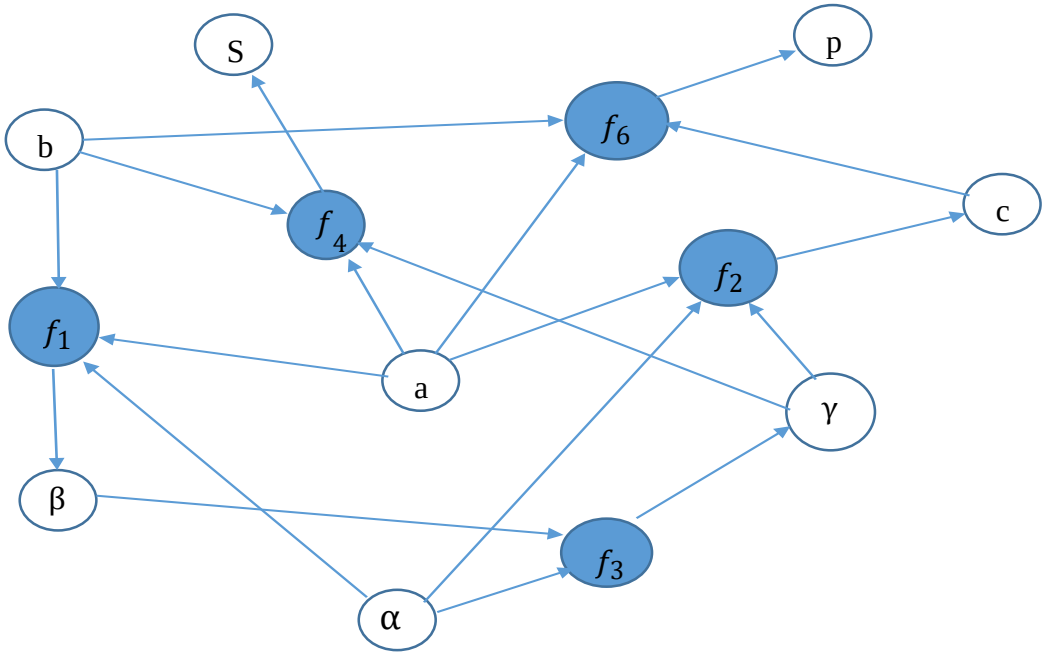
1. Tập biến $M = \{a,b,c, \alpha, \beta, \gamma, h_a, h_b, h_c, p, S, r, R,...\}$, trong đó *a, b, c* là 3 cạnh; α, β, γ là 3 góc ứng với 3 cạnh; h_a, h_b, h_c là các đường cao tương ứng với ba cạnh; *S* là diện tích; *P* là chu vi; *r, R* là bán kính đường tròn nội tiếp và ngoại tiếp của tam giác *ABC...*

2. Các quan hệ *F* gồm:

f1: $\frac{a}{\sin\alpha} = \frac{b}{\sin\beta}$; *f2:* $\frac{a}{\sin\alpha} = \frac{c}{\sin\gamma}$; *f3:* $\alpha + \beta + \gamma = 180^o$; *f4:* $S = \frac{1}{2}ab\sin\alpha$; *f5:* $S = \frac{1}{2}ah_a$:

f6: $S = (p(p-a)(p-b)(p-c))^{0.5}$

f7: $p = (a + b + c)/2$



Hình 1.4. Sơ đồ thể hiện một mạng tính toán

Bước2:

Chuyển từ cách mô tả bằng mạng ngữ nghĩa (mô hình hình học, hình 1.4) sang mô tả bằng ma trận (mô hình toán học). Để tạo ma trận, chọn các cột là hàm từ f_1 đến f_7; các biến là các hàm; các liên kết giữa biến và hàm nếu tồn tại nhận giá trị *-1*; giữa biến và hàm không có liên kết nhận giá trị *0* như bảng dưới đây.

Biến\hàm	*f1*	*f2*	*f3*	*f4*	*f5*	*f6*	*f7*
a	*-1*	*-1*	*0*	*-1*	*-1*	*-1*	*-1*
b	*-1*	*0*	*0*	*-1*	*0*	*-1*	*-1*
c	*0*	*-1*	*0*	*0*	*0*	*-1*	*-1*
α	*-1*	*-1*	*-1*	*0*	*0*	*0*	*0*
β	*-1*	*0*	*-1*	*0*	*0*	*0*	*0*
γ	*0*	*-1*	*-1*	*-1*	*0*	*0*	*0*
h_a	*0*	*0*	*0*	*0*	*-1*	*0*	*0*
P	*0*	*0*	*0*	*0*	*0*	*-1*	*-1*
S	*0*	*0*	*0*	*-1*	*-1*	*-1*	*0*

Bước 3: Kích hoạt các biến đã cho (bằng cách đổi -1 thành +1) như bảng dưới đây

Biến\hàm	*f1*	*f2*	*f3*	*f4*	*f5*	*f6*	*f7*

a	*+1*	*+1*	*0*	*+1*	*+1*	*+1*	*+1*
b	*+1*	*0*	*0*	*+1*	*0*	*+1*	*+1*
c	*0*	*-1*	*0*	*0*	*0*	*-1*	*-1*
α	*+1*	*+1*	*+1*	*0*	*0*	*0*	*0*
β	*-1*	*0*	*-1*	*0*	*0*	*0*	*0*
γ	*0*	*-1*	*-1*	*-1*	*0*	*0*	*0*
h_a	*0*	*0*	*0*	*0*	*-1*	*0*	*0*
P	*0*	*0*	*0*	*0*	*0*	*-1*	*-1*
S	*0*	*0*	*0*	*-1*	*-1*	*-1*	*0*

Bước 4: Từ bước một, ta nhận thấy trong công thức f_1 biến β có có thể tính được do đã biết *a, b,* α Một cách tổng quát có thể phát biểu quy tắc "trong một hàm có *n* biến; nếu cho biết *n-1* biến; biến còn lại hoàn toàn có thể tinh được". Đối chiếu quy tắc đó vào bảng ở bước 3 ta quan sát cột có biến f_1 Cột này có ba dấu (+) ứng với các biến đã cho biết và chỉ có một biến có dấu (-) cho nên có thể tính được biến có dấu trừ này. (biến β). Từ đó, rút ra quy tắc cho bước 4 "Cột nào chỉ có một và chỉ một dấu -1 thì đổi thành +1). Ta có bảng kết quả như dưới đây. Trong bảng, ta ký hiệu tập đã cho các giá trị là A_0. Tập dùng hàm f_1 để tính là tập A_1

Biến\hàm	f1	f2	f3	f4	f5	f6	f7
a	+1(A_0)	+1(A_0)	0	+1(A_0)	+1(A_0)	+1(A_0)	+1(A_0)
b	+1(A_0)	0	0	+1(A_0)	0	+1(A_0)	+1(A_0)
c	0	+1(A_3*)	0	0	0	+1(A_3)	+1(A_3)
α	+1(A_0)	+1(A_0)	+1(A_0)	0	0	0	0
β	+1(A_1*)	0	+1(A_1)	0	0	0	0
γ	0	+1(A_2)	+1(A_2*)	+1(A_2)	0	0	0
h_a	0	0	0	0	+1(A_5*)	0	0
P	0	0	0	0	0	+1(A_6*)	+1(A_6)
S	0	0	0	+1(A_4*)	+1(A_4)	+1(A_4)	0

Bước 5. Lặp lại bước 4 một cách tương tự, ta có sơ đồ lời giải sau. Lời giải của bài toán:

A_0=A={a,b, α}$\xrightarrow{f_1}$ A_1={a,b, α, β}$\xrightarrow{f_3}$ A_2={a, b, α, β, γ}$\xrightarrow{f_2}$ A_3=$\xrightarrow{f_5}$ A_5 {a,$b\alpha$, β, γ, c}$\xrightarrow{f_4}$ A_4 ={ a, b , α, β, γ, c, S}= { a, b , α, β, γ, c, S, h_a}$\xrightarrow{f_6}$ A_6 = { a, b , α, β, γ, c, S, h_a, P}.

Từ đó, lời giải sẽ là: $D_1 = \{f_1, f_3, f_2, f_4, f_5, f_6\}$.

Có thể nhận thấy , lời giải này không phải lời giải tốt vì có bước tính toán thừa là f_5.

Bỏ f_5, ta được lời giải tốt là: $D_2 = \{f_1, f_3, f_2, f_4, f_6\}$. Và sơ đồ lời giải tốt như sau:

$A_0 = A = \{a, b, \alpha\} \rightarrow A_1 = \{a, b, \alpha, \beta\} \rightarrow A = \{a, b, \alpha, \beta, \gamma\} \rightarrow A_3 = \{a, b, \alpha, \beta, \gamma, c\} \rightarrow$
$\rightarrow A_4 = \{a, b, \alpha, \beta, \gamma, c, S\} \rightarrow A_5 = \{a, b, \alpha, \beta, \gamma, c, S, P\}$.

Lời giải tối ưu của bài toán

Định nghĩa 6: Lời giải tối ưu là lời giải ngắn nhất trong tất cả các lời giải tốt (số hàm để tính toán là ít nhất).

Mệnh đề 1. Nếu bài toán $A \rightarrow B$ là giải được thì sẽ tồn tại lời giải tối ưu cho bài toán.

Ngoài ra ta có thể áp dụng thuật toán A^*(thuật toán heuristic) để tìm lời giải tối ưu trong trường hợp bài toán giải được.

Kiểm định giả thuyết cho bài toán

Xét bài toán $A \rightarrow B$ trên mạng tính toán *(M, F).* Xét giả thiết *A* của bài toán xem thừa hay thiếu và tìm cách điều chỉnh giả thiết *A*.

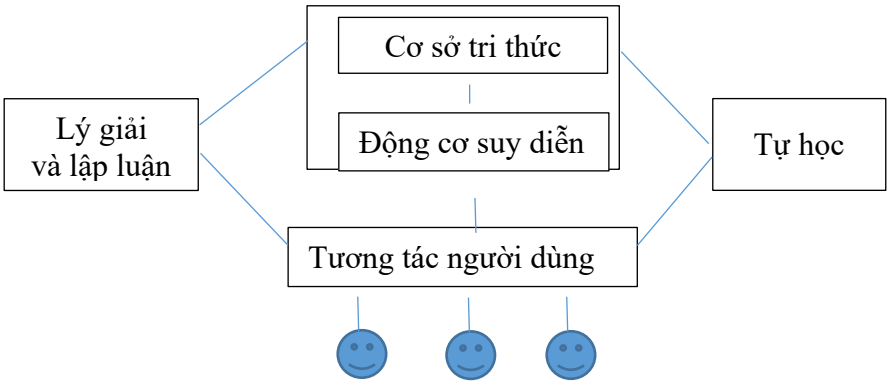
Trước hết ta cần xét xem bài toán có giải được hay không. Nếu bài toán giải được thì giả thiết cho là đủ. Tuy nhiên, có thể xảy ra tình trạng thừa giả thiết. Ta dựa vào thuật toán để thu gọn giả thiết từ kết quả của lời giải.

1.3 Mục đích xây dựng các hệ thống dựa trên tri thức

Các hệ thống dựa trên tri thức với các mục đích chính sau:

- Cung cấp các hệ thống với mức thông minh cao
- Hỗ trợ con người trong khám phá và phát triển các lĩnh vực chưa được biết tới
- Cung cấp lượng lớn tri thức trong các lính vực khác nhau
- Hỗ trợ quản lý tri thức trong các cơ sở tri thức
- Giải quyết các vấn đề một cách tốt hơn so với các hệ thống thông tin truyền thống
- Thu thập các nhận thức mới bằng mô phỏng các tình huống chưa được biết tới
- Hỗ trợ, cải thiện đáng kể hiệu suất phần mềm
- Giảm đáng kể thời gian và chi phí phát triển các hệ thống điện toán

1.4 Các thành phần của hệ thống dựa trên tri thức



Hình 1.5. Các thành phần của hệ thống dựa trên tri thức

Các hệ dựa theo tri thức gồm hai phần cơ bản : cơ sở tri thức (KBS) và chương trình tìm kiếm (Search Program) được gọi là động cơ suy diễn (Inference Engine) [1]. Động cơ suy diễn là một chương trình phần mềm có khả năng suy diễn từ tri thức thành cơ sở tri thức. Cơ sở tri thức có thể được sử dụng như kho chứa các dạng tri thức khác nhau. Do tiềm năng của các chuyên gia nằm ở khả năng lý giải và lập luận nên hiệu năng của các hệ chuyên gia phụ thuộc vào việc quyết định hay đề xuất nào được sử dụng để lý giải hay lập luận. Con người có thể học những việc mới, song đôi khi có thể quên kiến thức đã biết. Mô phỏng việc học như vậy của con người chính là nhiệm vụ của các hệ dựa theo tri thức. Quy mô của các hệ dựa tri thức có thể khác nhau tùy thuộc vào cách mô phỏng. Mô hình dựa tri thức có thể cập nhật theo thói quen mang tính cơ học hoặc cập nhật tự động bằng máy móc (hay chính là học máy). Ngoài ra, hệ thống dựa theo tri thức cần có mối tương tác với người dùng được trang bị các phương tiện xử lý ngôn ngữ tự nhiên (hình 1.5).

1.5 Phân loại các hệ thống dựa trên tri thức

Theo một số các tác giả [1], các hệ dựa tri thức có thể chia thành 5 nhóm như sau:

1.5.1. Hệ chuyên gia

Hệ chuyên gia là sơ khai của các hệ dựa tri thức và là hệ thống thông dụng nhất. Nó có thể thay thế một hoặc nhiều chuyên gia để giải quyết các vấn đề (hay bài toán). Nó được dùng cho nhiều tình huống hơn hệ thống thông tin dựa trên máy tính truyền thống. Các hệ chuyên gia kinh điển điển hình là hệ *MYCIN:* hệ chẩn đoán huyết học ttiên rong y tế, là hệ dựa theo luật. Hệ chuyên gia *PROCPECTOR* là hệ chuyên gia dùng đầu trong tìm kiếm các mỏ đá đỏ dựa trên lý thuyết Bayes. Các hệ chuyên gia tiên tiến, người đọc có thể tham khảo ở [2, 15, 22].

`

1.5.2. Các hệ thống liên kết

Các hệ được gọi là các hệ thống liên kết gồm các hệ siêu đa phương tiện, hệ siêu văn bản, hệ siêu âm thanh, hệ siêu ảnh động. Các hệ liên kết được hiểu theo nghĩa có chất lượng tốt và thể hiện sự thông minh. Các hệ thống liên kết đa phương tiện như Internet ngày nay đã trở nên phổ cập và thông dụng.

1.5.3. Các hệ quản trị cơ sở dữ liệu liên kết, tương tác người dùng thông minh

Ngày nay tri thức suy diễn của người dùng có thể được cất giữ trong các cơ sở dữ liệu để dùng cho các ứng dụng trong những môi trường gần giống nhau.

1.5.4. Các hệ dựa tri thức cho Công nghệ Phần mềm

Đây là một trong các dạng của các hệ cơ sở tri thức. Các hệ dựa tri thức cho Công nghệ Phần mềm chỉ dẫn cách phát triển các hệ thống thông tin hay hệ thống thông minh nhằm nâng cao hiệu quả và chất lượng phần mềm.

1.5.5. Các hệ thống dựa theo tri thức cho đào tạo thông minh

Các hệ thống đó giúp giảng dạy, hướng dẫn học tập và thực hành trong các lĩnh vực nghề nghiệp, kỹ thuật, văn hóa khác nhau. Ngoài việc cung cấp tư liệu học tập, các hệ thống này có khả năng đánh giá trình độ, kỹ năng học viên khối kỹ thuật hoặc phi kỹ thuật; soạn giáo trình bài giảng và ngân hàng đề thi, ngân hàng câu hỏi. Một trong những nhánh nối tiếng của hệ thống này là hệ đào tạo dựa trên đối thoại.

1.6 Các khó khăn trong xây dựng các hệ thống dựa trên tri thức

1.6.1 Xây dựng hệ dựa tri thức

Phần lớn các hệ đều bị giới hạn bởi các tri thức cho bài toán cần giải và rất ít tri thức khác được sử dụng. Ví dụ:

NẾU ô tô không khởi động được THÌ kiểm tra ac-quy

Trong ví dụ này, hệ thống không có thông tin về quan hệ giữa ắc quy và khả năng hoạt động của xe. Nó chỉ có thể là hàm heuristic (kinh nghiệm thực tế) để kiểm tra ac-quy trong tình huống này.

1.6.2 Đặc tính của tri thức

Vì tri thức đóng vai trò then chốt trong tìm kiếm lời giải và mô hình hóa trí thông minh, do đó, hệ cơ sở tri thức là thành phần cốt lõi của các hệ dựa theo tri thức. Để giải quyết chỉ 1 vấn đề đơn giản trong thực tế, đã phải có một lượng các kiến thức đủ lớn. Mặt khác, tri thức luôn thay đổi. Điều đó làm khó cho việc phát triển của các hệ thống dựa theo tri thức.

1.6.3 Độ lớn của cơ sở tri thức

Như đã nói ở trên, để giải quyết 1 vấn đề cho dù cực kỳ đơn giản cũng đòi hỏi một lượng tri thức rất lớn. Trong kho cơ sở dữ liệu chứa một số "khúc" tri thức được mô tả bằng kỹ thuật khác biệt. Tri thức được cất giữ ở các kho khác loại tạo nên sự phức tập thiếu tính cấu trúc. Tri thức không được cất giữ theo tiến trình hoặc tức thời, trừ các tri thức suy diễn.

1.6.4 Thu thập tri thức

Thu thập tri thức qua một hoặc nhiều chuyên gia rất khó khăn. Các kỹ sư tri thức cần "biết" cách trình bày yêu cầu với các chuyên gia để giúp hình thành và giải quyết các bài toán thực tế và mô tả trí thức đó cho hệ thống. Hiện nay chưa có một thủ tục được định trước cho việc thu thập và mô tả tri thức.

1.6.5 Học chậm và phân tích

Khi được cài đặt, mô hình KBS thường chậm và không thể sử dụng với một lượng lớn tri thức. Khi được cài đặt nó có thể khó bảo trì. Giải quyết một vấn đề có thể phải áp dụng nhiều tri thức, kỹ thuật và công cụ, các tiến trình của KBS và môi trường áp dụng, phát triển đã tạo nên sự liên kết giữa KBS và cơ sở dữ liệu.

Trên tất cả, điều khó khăn để nghiên cứu chính xác và xây dựng một mô hình ứng dụng AI/KBS đã mở ra điều kiện phát triển cho ngành học máy, khám phá ra ảnh hưởng của tri thức đối với việc đưa ra phán đoán và kỹ năng xử lý một lương lớn các vấn đề.

1.7 Lập trình thông minh

Ta đã biết, trong tính toán truyền thống:

PROGRAM = DATA + ALGORITHM

Vậy đối với hệ tri thức có thể suy diễn tương tự

INTELLIGENCE.PROGRAM = KNOWLEDGE + INFERENCE

Sự hiểu biết chứa các kiến thức chuyên sâu về một lĩnh vực nào đó.

Luật suy diễn là lập luận mà trong đó kết luận được rút ra từ các sự kiện được biết trước theo kiểu: nếu các tiền đề là đúng thì kết luận phải đúng. Nghĩa là các sự kiện cho trước đòi hỏi rằng kết luận là đúng.

1.8 Các ngôn ngữ, công cụ sử dụng cho hệ cơ sở tri thức

Các công cụ truyền thống cơ bản gồm:

- PROLOG (*Programing Logic*)
- LISP (*List Processing*)

Các công cụ tiên tiến điển hình cho hệ cơ sở dựa trí thức:

- AIML (*Artificial Intelligence Modeling Language*)

- MATLAB
- JavaNNS (*Java Nơ ron Networks Simulator*)
- CLIPS (*C Language Integrated Production System*)

CÂU HỎI VÀ BÀI TẬP

1. Thế nào là tri thức, hệ cơ sở tri thức?
2. Nêu các phương pháp mô tả tri thức mà các bạn đã biết.
3. Bạn hãy trình bày biểu đồ mô tả từ dữ liệu đến trí tuệ.
4. Bạn hãy trình bày tháp quản lý dữ liệu, thông tin, tri thức và trí tuệ (uyên thâm); Nêu các thành phần và ý nghĩa của các mức trong tháp.
5. Cho tam giác ABC, mạng tính toán *{M, F}* trong đó, $M=\{a, b, c, \alpha, \beta, \gamma, h_a, h_b, h_c, p, S, r, R...\}$ là tập các biến của tam giác; tập hàm *F={f1, f2, f3, f4, f5, f6};* trong đó: $f1:(a/\sin\alpha=b/\sin\beta)$; $f2:(c/\sin\gamma=b/\sin\beta)$; $f3:(\alpha+\beta+\gamma=180^o)$; $f4:(2p=a+b+c)$; $f5:(S=1/2.c.h_c)$; $f6: S=[p(p-a)(p-b)(p-c)]^{1/2}$; $A=\{a, b, \alpha\}$; $B=\{p, S\}$.
   a) Tìm lời giải của bài toán $A \rightarrow B$? Sử dụng thuật toán vết dầu loang.
   b) Tìm lời giải tốt? lời giải tối ưu?

CHƯƠNG 2:

CÁC HỆ THỐNG TRI THỨC DỰA TRÊN XÁC SUẤT

Trong chương "Học máy" của trí tuệ nhân tạo, ta đã tìm hiểu thuật toán cây quyết định ID3, mạng Bayes, thuật toán SVM (Support Vectơr Machine). Chương hai nêu hai thuật toán học liên quan tới xác suất: một trong các thành phần của các hệ cơ sở tri thức. Hệ mờ cũng liên qua nhiều tới xác suấtt, chúng ta dành một chương riêng để nghiên cứu.

Chương trước ta đã biết về biểu diễn tri thức và các kỹ thuật suy diễn trong trường hợp giả định có sẵn tri thức và có thể biểu diễn tường minh tri thức. Tuy nhiên, trong nhiều tình huống, sẽ không có sẵn tri thức như:

- Kỹ sư phần mềm cần thu nhận tri thức từ chuyên gia lĩnh vực.
- Cần biết các luật mô tả lĩnh vực cụ thể
- Bài toán không được biểu diễn tường minh theo luật, sự kiện hay các quan hệ.

Do vậy, cần phát triển các hệ thống và học. Học là xác định vấn đế chưa biết. Trong các hệ học, giả sử các sự kiện của giả thiết và sự kiện kết luận đã cho, điều cần học (đơn giản là xác định) ở đây cần biết là mối quan hệ (hay quy tắc, hay luật) giữa giả thiết và kết luận. Có hai cách tiếp cận cho hệ thống học là: Học từ ký hiệu và học từ dữ liệu. Học từ ký hiệu bao gồm việc hình thức hóa, sửa chữa các luật tường minh, sự kiện và các quan hệ; học từ dữ liệu được áp dụng cho những hệ thống được mô hình hóa dưới dạng số liên quan đến các kỹ thuật tối ưu các tham số. Học theo dạng số bao gồm mạng Nơ-ron nhân tạo, thuật giải di truyền, bài toán tối ưu truyền thống.

Dưới đây giới thiệu một số thuật toán học sử dụng phổ biến trong các hệ cơ sở tri thức.

2.1 Thuật toán độ hỗn loạn

Thuật toán độ lộn xộn sử dụng công thức Entropy (dựa trên xác suất để làm tiêu chí tìm quy luật cho bài toán học).

2.1.1 Bài toán

Cho tập hợp dữ liệu học (Bảng 5.1) gồm các đặc trưng đầu vào: i) xem trời (Outlook), ii) nhiệt độ (Temperature), iii) độ ẩm (Humidity), iv) gió (Windy) với 14 mẫu thời tiết. Đầu ra là quyết định chơi Tennis với giá trị (Yes, No). Dùng thuật toán độ lộn xộn tìm quy luật cho quyết định đi chơi (Play) Tennis hay không?

Bảng 2.1. Tập dữ liệu thời tiết

TT	Outlook	Temperature	Humidity	Windy	Play
1	Sunny	Hot	High	False	No
2	Sunny	Hot	High	True	No
3	Overcast	Hot	High	False	Yes
4	Rainy	Mild	High	False	Yes
5	Rainy	Cool	Normal	False	Yes
6	Rainy	Cool	Normal	True	No
7	Overcast	Cool	Normal	True	Yes
8	Sunny	Mild	High	False	No
9	Sunny	Cool	Normal	False	Yes
10	Rainy	Mild	Normal	False	Yes
11	Sunny	Mild	Normal	True	Yes
12	Overcast	Mild	High	True	Yes
13	Overcast	Hot	Normal	False	Yes
14	Rainy	Mild	High	True	No

2.1.2 Thuật toán độ lộn xộn

Lý thuyết thông tin cho công thức xác định độ lộn xộn:

$$E_{A_i}(b) = \sum_b \frac{n_b}{n_t}\left[-\sum_c \frac{n_{bc}}{n_b}\log_2\frac{n_{bc}}{n_b}\right]$$

trong đó:

n_b: Số mẫu trong nhánh b

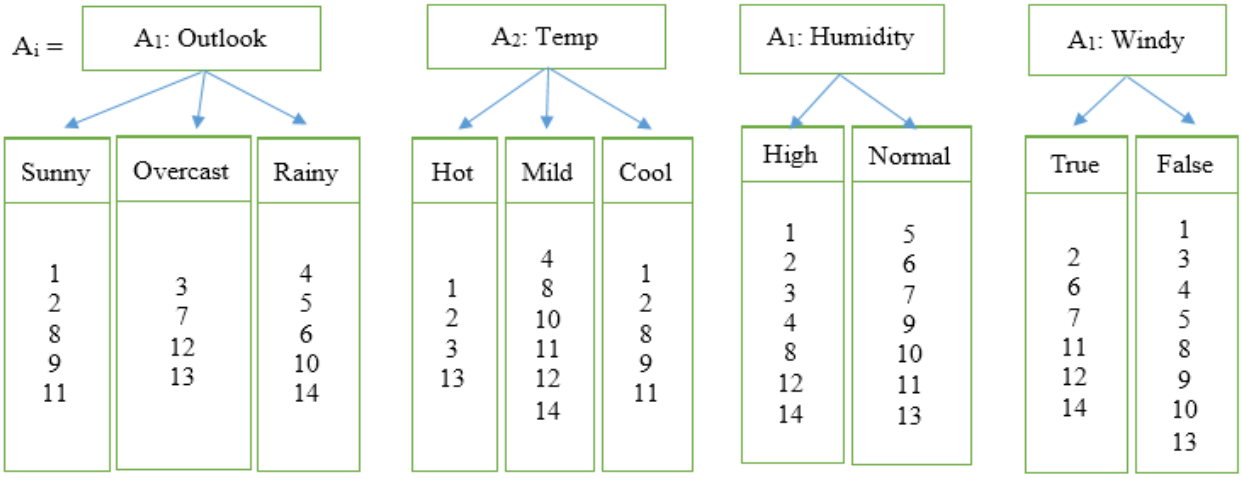
n_t: Tổng số mẫu trong tất cả các nhánh

n_{bc}: Tổng số mẫu trong nhánh b thuộc lớp đầu ra (Play) c (c có giá trị: Yes hoặc No)

Thuật toán độ lộn xộn hay hỗn loạn dựa trên công thức trên theo các bước sau:

Bước 1: Phân hoạch (hay đơn giản là chia) toàn bộ cơ sở dữ liệu theo đặc trưng đầu vào

Chọn 4 đặc trưng đầu vào i) xem trời (Outlook), ii) nhiệt độ (Temperature), iii) độ ẩm (Humidity), iv) gió (Windy) làm bốn gốc (Bảng 5.1); mỗi gốc chia thành các cành theo các giá trị *b* mà đặc trưng đó thể hiện; Mỗi cành chia tiếp thành các lá *c* có giá trị đầu ra (Play) là *c=Yes* hoặc *c=No*.



Hình 2.1. Các giá trị xác suất của các sự kiện

Bước 2: Tính độ lộn xộn

$E_{A1}(b)= \frac{5}{14}[-\frac{2}{5}log_2\frac{2}{5}-\frac{3}{5}log_2\frac{3}{5}]+\frac{4}{14}[-\frac{4}{4}log_2\frac{4}{4}-\frac{0}{4}log_2\frac{0}{4}]+\frac{5}{14}[-\frac{3}{5}log_2\frac{3}{5}-\frac{2}{5}log_2\frac{2}{5}]=0,69$

$E_{A2}(b)= \frac{4}{14}[-\frac{2}{4}log_2\frac{2}{4}-\frac{2}{4}log_2\frac{2}{4}]+\frac{6}{14}[-\frac{4}{6}log_2\frac{4}{6}-\frac{2}{6}log_2\frac{2}{6}]+\frac{4}{14}[-\frac{3}{4}log_2\frac{3}{4}-\frac{1}{4}log_2\frac{1}{4}]=0,91$

$E_{A3}(b)= \frac{7}{14}[-\frac{2}{7}log_2\frac{2}{7}-\frac{5}{7}log_2\frac{5}{7}]+\frac{7}{14}[-\frac{6}{7}log_2\frac{6}{7}-\frac{1}{7}log_2\frac{1}{7}] =0,73$

$E_{A4}(b)= \frac{6}{14}[-\frac{3}{6}log_2\frac{3}{6}-\frac{3}{6}log_2\frac{3}{6}]+\frac{8}{14}[-\frac{6}{8}log_2\frac{6}{8}-\frac{2}{8}log_2\frac{2}{8}] =0,89$

Bước 3: Chọn tiêu chí gốc có độ lộn xộn nhỏ nhất: $min(E_{Ai}(b))= E_{A1}(b)=0,69$

Bước 4: Dựa vào số hạng Entropy trong tiêu chí A_1 ta có luật sau:

Luật 1: IF "Outlook" là "Overcast" THEN "Play" là "Yes"

Bước 5: Tổ hợp chập 2 thuộc tính

$E_{(A1 \text{ là Sunny})}\wedge A_2=\frac{2}{5}[-\frac{0}{2}log_2\frac{0}{2}-\frac{2}{2}log_2\frac{2}{2}]+\frac{2}{5}[-\frac{1}{2}log_2\frac{1}{2}-\frac{1}{2}log_2\frac{1}{2}]+\frac{1}{5}[-\frac{1}{1}log_2\frac{1}{1}-\frac{0}{1}log_2\frac{0}{1}]=0,4$

$E_{(A1 \text{ là Sunny})}\wedge A_3=\frac{3}{5}[-\frac{0}{3}log_2\frac{0}{3}-\frac{3}{3}log_2\frac{3}{3}]+\frac{2}{5}[-\frac{2}{2}log_2\frac{2}{2}-\frac{0}{2}log_2\frac{0}{2}]=0$

$E_{(A1 \text{ là Sunny})}\wedge A_4=\frac{2}{5}[-\frac{1}{2}log_2\frac{1}{2}-\frac{1}{2}log_2\frac{1}{2}]+\frac{3}{5}[-\frac{1}{3}log_2\frac{1}{3}-\frac{2}{3}log_2\frac{2}{3}]=0,95$

Chọn tiêu chí gốc có Entropy min

Dựa vào số hạng Entropy trong tiêu chí A_1 ta có luật sau:

Luật 2: IF "Outlook" là "Sunny" and "Humidity" là "High" THEN "Play" là "No"

Luật 3: IF "Outlook" là "Sunny" and "Humidity" là "Normal" THEN "Play" là "Yes"

$E_{(A1 \text{ là Rainy})} \wedge A_2 = \frac{0}{5}[-\frac{0}{0}\log_2\frac{0}{0} - \frac{0}{0}\log_2\frac{0}{0}] + \frac{3}{5}[-\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3}] + \frac{2}{5}[-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}] = 0{,}95$

$E_{(A1 \text{ là Rainy})} \wedge A_3 = \frac{2}{5}[-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}] + \frac{3}{5}[-\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3}] = 0{,}95$

$E_{(A1 \text{ là Rainy})} \wedge A_4 = \frac{2}{5}[-\frac{0}{2}\log_2\frac{0}{2} - \frac{2}{2}\log_2\frac{2}{2}] + \frac{3}{5}[-\frac{3}{3}\log_2\frac{3}{3} - \frac{0}{3}\log_2\frac{0}{3}] = 0$

Chọn tiêu chí gốc có Entropy min

Dựa vào số hạng Entropy trong tiêu chí A_1 ta có luật sau:

Luật 4: IF "Outlook" là "Rainy" and "Windy" là "True" THEN "Play" là "No"

Luật 5: IF "Outlook" là "Rainy" and "Windy" là "False" THEN "Play" là "Yes"

2.2 Thuật toán Bayes

Thuật toán sử dụng khá phổ biến trong thực tế, vì nó cho phép tính xác suất điều kiện đơn giản, nhanh chóng và kết quả tốt.

2.2.1 Định lý Bayes

Phương pháp Bayes cho phép tính xác suất xảy ra của một sự kiện ngẫu nhiên *X* khi biết sự kiện liên quan *Y*. Đại lượng này được gọi là xác suất có điều kiện hay xác suất hậu nghiệm vì nó được rút ra từ giá trị được cho của *Y* hoặc phụ thuộc vào giá trị đó.

Theo định lý Bayes, sác xuất xảy ra phụ thuộc vào các yếu tố:

- Xác suất xảy ra *X* của riêng nó, không liên quan đến yếu tố khác. Đây được gọi là xác suất tiên nghiệm *(ký hiệu P(X))*
- Xác suất xảy ra *Y* không liên quan đến yếu tố khác. Đại lượng này được gọi là hằng số chuẩn hóa, vì nó luôn giống nhau, không phụ thuộc vào sự kiện đang muốn biết *(ký hiệu P(Y))*
- Xác suất xảy ra *Y* khi biết *X*. Đại lượng này gọi là khả năng xảy ra *Y* khi biết *X* đã xảy ra *(ký hiệu P(Y/X))*

Để xác định xác suất giả thuyết *Y* khi xảy ra sự kiện ngẫu nhiêu *X* ta có công thức tính xác suất theo định lý Bayes như sau:

$$P(X/Y) = \frac{P(Y/X).P(X)}{P(Y)}$$

Từ kết quả tính được ta có thể đánh giá được xác suất của sự kiện ngẫu nhiên *X* là đúng hay sai hay có xảy ra hay không?

2.2.2. Bài toán và thuật toán Bayes đơn giản

Cho tập dữ liệu dự báo như Bảng 5.1. Giả sử có tình huống thời tiết xay ra, cần quyết định có chơi Tennis không dùng thuật giải Bayes cho hai trường hợp thời tiết như sau:

a) Dữ liệu của mẫu tin 1 cần dự báo (giống mẫu 1 của tập dữ liệu đã được học)

Outlook	Temp	Humidity	Windy	Play
Sunny	Cool	High	True	?

b) Dữ liệu của mẫu tin 2 cần dự báo (không giống mẫu nào đã học, cần suy diễn)

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	?

Để hiểu thuật toán, thực hiện các bước của thuật toán trên các bài toán đã nêu.

Trường hợp 1:

Bước 1: Phân hoạch dữ liệu theo đặc trưng đầu vào

Outlook			**Temp**			**Humidity**			**Windy**			**Play**	
	Yes	*No*		*Yes*	*No*		*Yes*	*No*		*Yes*	*No*	*Yes*	*No*
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Wild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								

Bước 2: Tính toán theo tiêu chí theo định lý Bayes

Áp dụng định lý Bayes ta có:

$$P(X/Y) = \frac{P(Y/X).P(X)}{P(Y)}$$

Theo bài trong mẫu tin $Y = (Y_1, Y_2, \dots, Y_n)$ có n giá trị thuộc tính được biết. Ta có :

$$P(X/Y) = \frac{P(Y1/X).P(Y2/X)....P(Yn/X).P(X)}{P(Y)}$$

$$P_{(Yes/X)} = \frac{P_{(Outlook\ =Sunny/Yes)} \cdot P_{(Temp\ =\ Cool/Yes)} \cdot P_{(Humidity\ =\ High/Yes)} \cdot P_{(Windy\ =\ True/Yes)} \cdot P_{(Yes)}}{P_{(X)}}$$

$$= \frac{(2/9 \,.\, 3/9 \,.\, 3/9 \,.\, 3/9) \,.\, 9/14}{P(X)} = \frac{0{,}007055}{P(X)}$$

$$P_{(No/X)} = \frac{P_{(Outlook\ =Sunny/No)} \cdot P_{(Temp\ =\ Cool/No)} \cdot P_{(Humidity\ =\ High/No)} \cdot P_{(Windy\ =\ True/No)} \cdot P_{(No)}}{P_{(X)}}$$

$$= \frac{(3/5 \,.\, 1/5 \,.\, 4/5 \,.\, 3/5) \,.\, 5/14}{P(X)} = \frac{0{,}027429}{P(X)}$$

Bước 3: Kết luận.

Từ kết quả trên, ta thấy, ước lượng xác suất dự báo mẫu tin *X* cho lớp "Play" là "Yes" nhỏ hơn ước lượng xác suất lớp "Play" là "No", Bayes đơn giản gán nhãn *X* cho lớp "Play" là "No".

Trường hợp 2:

Bước 1: Phân hoạch theo đặc trưng đầu vào

Outlook			**Temp**			**Humidity**			**Windy**			**Play**	
	Yes	*No*		*Yes*	*No*		*Yes*	*No*		*Yes*	*No*	*Yes*	*No*
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Wild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								

Bước 2: Tính toán theo tiêu chí theo định lý Bayes

Áp dụng định lý Bayes ta có:

$$P(X/Y) = \frac{P(Y/X).P(X)}{P(Y)}$$

Theo bài trong mẫu tin $Y = (Y_1, Y_2, \dots, Y_n)$. Giả sử các $Y_1, Y_2, \dots, Y_n$ là độc lập, ta có :

$$P(X/Y) = \frac{P(Y1/X).P(Y2/X)....P(Yn/X).P(X)}{P(Y)}$$

$$P_{(Yes/X)} = \frac{P_{(Outlook\ =Sunny/Yes)} \cdot P_{(Temp\ =\ Hot/Yes)} \cdot P_{(Humidity\ =\ High/Yes)} \cdot P_{(Windy\ =\ False/Yes)} \cdot P_{(Yes)}}{P_{(X)}}$$

$$= \frac{(2/9\ .\ 2/9\ .\ 3/9\ .\ 6/9)\ .\ 9/14}{P(X)} = \frac{0{,}007055}{P(X)}$$

$$P_{(No/X)} = \frac{P_{(Outlook\ =Sunny/No)} \cdot P_{(Temp\ =\ Hot/No)} \cdot P_{(Humidity\ =\ High/No)} \cdot P_{(Windy\ =\ False/No)} \cdot P_{(No)}}{P_{(X)}}$$

$$= \frac{(3/5\ .\ 2/5\ .\ 4/5\ .\ 2/5)\ .\ 5/14}{P(X)} = \frac{0{,}027429}{P(X)}$$

Bước 3: Kết luận

Từ kết quả, ta thấy ước lượng xác suất dự báo cho mẫu tin *X* cho lớp "Play" là "Yes" nhỏ hơn ước lượng xác suất lớp "Play" là "No", Bayes thơ gây gán nhãn *X* cho lớp "Play" là "No".

Thuật toán này gọi là Thuật toán Bayes đơn giản do các đặc trưng đầu vào $Y_1, Y_2, \ldots, Y_n$ giả thiết là độc lập. Trong trường hợp phụ thuộc, các xác suất có điều kiện liên quan cần phải cho, điều đó không phải lúc nào cũng thống kê được.

CÂU HỎI VÀ BÀI TẬP

1. Công thức tính độ hỗn loạn dựa vào cơ sở lý thuyết hoặc định lý nào?
2. Cho bảng dữ liệu huấn luyện dưới đây, trong đó các dòng A, B, C là thuộc tính, D là nhãn phân loại.

A	0	0	1	1	0	1
B	0	0	1	0	1	0
C	1	0	1	1	0	0
D	-	+	+	+	-	-

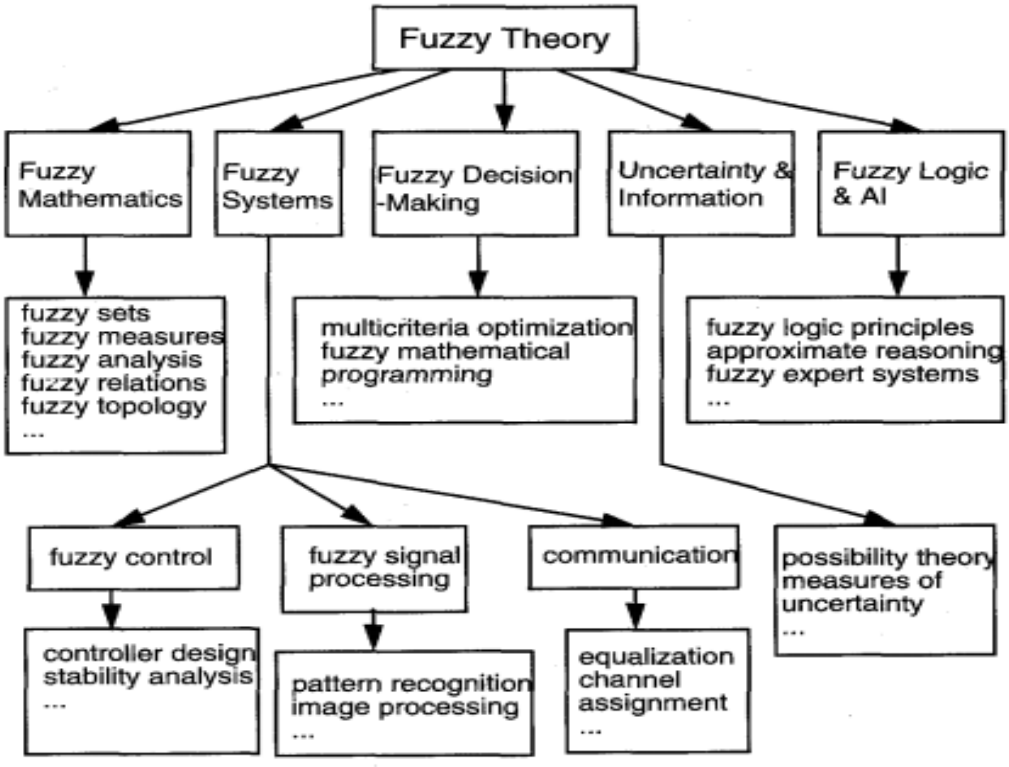
a) Sử dụng thuật toán độ hỗn loạn rút ra các luật
b) Sử dụng thuật toán Bayes kiểm tra trường hợp B

CHƯƠNG 3:

HỆ MỜ

Lý thuyết mờ được khởi xướng bởi Lotfi A. Zadeh vào năm 1965. Ngay từ năm 1962, ông đã cho biết, để xử lý các hệ thống sinh học "chúng ta cần nhiều loại toán học khác nhau, như toán học mờ, vì các loại toán kinh điển không thể diễn tả đầy đủ được về phân bố xác suất". Sau đó, ông chính thức hóa các ý tưởng của mình vào công bố bài báo "tập mờ: Fuzzy Sets" [8]. Cuối năm 1970, nhiều phương pháp mờ như thuật toán mờ, ra quyết định mờ...đã được đề xuất và công bố. Đến nay, lý thuyết và ứng dụng mờ, các hệ lai mờ đã phat triển.

Các lĩnh vực nghiên cứu trong lý thuyết mờ



Hình 3.1. Phân loại lý thuyết mờ

Về lý thuyết mờ, chúng ta sử dụng các khái niệm cơ bản của tập mờ hoặc hàm thành viên. Lý thuyết mờ có thể được phân loại theo hình 3.1 [4] với năm nhánh chính:

(i) toán học mờ (Fuzzy Mathematics), cung cấp các khái niệm toán học và được mở rộng bằng cách thay thế một số khái niệm cổ điển bằng lý thuyết tập mờ;

(ii) logic mờ và trí tuệ nhân tạo (Fuzzy Logic and AI) trong đó, gần giống với logic kinh điển được giới thiệu và hệ thống chuyên gia được phát triển dựa trên thông tin mờ và lập luận gần đúng;

(iii) hệ thống mờ (Fuzzy Systems), trong đó gồm, điều khiển mờ (Fuzzy Control), phương pháp tiếp cận trong xử lý tín hiệu mờ (Fuzzy Signal Processing) và thông tin liên lạc hay truyền thông (Communication);

(iv) lý thuyết không chắc chắn và thông tin (Uncertainty and Information), nơi các dạng không chắc chắn và quan niệm về thông tin được phân tích;

(v) quyết định mờ (Fuzzy Decision Making) trong đó, xem xét và các giải bài toán tối ưu với các ràng buộc mềm.

Tất nhiên, năm nhánh này không phải là độc lập mà có mối liên hệ mạnh mẽ giữa chúng. Ví dụ, điều khiển mờ sử dụng các khái niệm toán học và logic mờ. Từ quan điểm thực tế, phần lớn các ứng dụng của lý thuyết mờ thường tập trung vào các hệ thống mờ, đặc biệt là điều khiển mờ như chúng ta có thể thấy từ một số hệ chuyên gia mờ thực hiện các chẩn đoán y học và hỗ trợ quyết định (Terano, Asai và Sugeno [1994]). Vì lý thuyết mờ vẫn trong giai đoạn phát triển của cả lý thuyết và thực tế, chúng ta hy vọng các ứng dụng thực tế vững chắc sẽ xuất hiện nhiều hơn trong lĩnh vực này.

Từ hình. 3.1 chúng ta thấy lý thuyết mờ là một lĩnh vực rất rộng mà bao gồm nhiều đề tài nghiên cứu. Trong giáo trình này, chúng ta tập trung vào một vài mục cơ bản liên quan tới toán học mờ (tập mờ), logic mờ, hệ mờ và ví dụ một bài toán xuyên suốt từ mờ hóa, xử lý mờ đến giải mờ để người học có thể nắm được bản chất của phương pháp và suy diễn mờ.

3.1 Tập mờ

Chúng tôi giới thiệu một số khái niệm cơ bản và thuật ngữ liên quan tới tập mờ. Nhiều trong số đó là phần mở rộng của các khái niệm cơ bản của một tập kinh điển (rõ), nhưng một số chỉ đối với tập mờ.

- Tập tỏ (Crisp Set): Cho tập A (tỏ). Một phần tử của tập là x, $x \in A$. Nếu c không phải là phần tử của A thì x không thuộc A, tức là:

$$\mu_A(x) = \begin{cases} 1 & if \quad x \in A \\ 0 & if \quad x \notin A \end{cases} \tag{3.1}$$

- Tập mờ (Fuzzy Set) *A* tồn tại hàm $\mu_A(x)$: $0 \leq \mu_A(x) \leq 1$

Định nghĩa 3.1:

Một tập mờ trong tập vũ trụ U được đặc trưng bởi một hàm thành viên (3.1) có giá trị trong khoảng [0, 1].

Do đó, một tập mờ là một sự tổng quát của một tập cổ điển bằng cách cho phép các hàm thành viên để có bất kỳ giá trị trong khoảng [0, 1]. Nói cách khác, các hàm thành viên của một tập cổ điển chỉ có thể lấy hai giá trị: 0 và 1, trong khi hàm thành viên của một tập mờ là một hàm liên tục với phạm vi [0, 1].

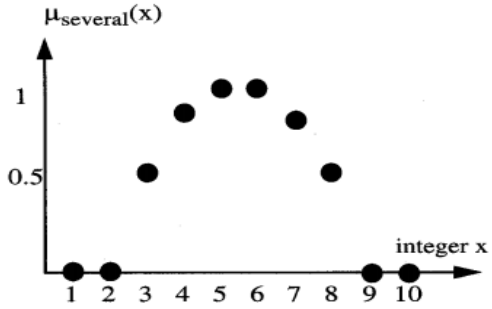
3.2 Các khái niệm cơ bản liên quan đến tập mờ

Các khái niệm về hỗ trợ, mờ Singleton, trung tâm, điểm giao nhau, chiều cao, tập mờ bình thường, cắt α, tập mờ lồi, và các phép chiếu được xác định như sau:

Độ hỗ trợ của một tập mờ *A* trong vũ trụ của *U* là một tập rõ chứa tất cả các phần tử của *U* mà có các giá trị thành viên trong *A* khác *0*, tức là,

$$supp(A) = \{x \in U | \mu_A(x) > 0\} \tag{3.2}$$

Trong đó *supp*(*A*) biểu thị sự hỗ trợ của tập mờ *A*. Ví dụ, sự hỗ trợ của tập mờ "several" trong *hình 3.1* là tập hợp các số nguyên *{3, 4, 5, 6, 7, 8)*. Nếu sự hỗ trợ của một tập mờ là rỗng, nó được gọi là một tập mờ rỗng. Một tập mờ đơn là tập mờ mà sự hỗ trợ của nó là một điểm duy nhất trong *U*.



Hình 3.1 Hàm thành viên cho tập mờ "Several"

Trung tâm của một tập mờ được định nghĩa như sau: Nếu giá trị trung bình của tất cả các điểm mà tại đó hàm thành viên của tập mờ đạt được giá trị tối đa của nó là hữu hạn, thì định nghĩa giá trị trung bình này như trung tâm của tập mờ, nếu giá trị trung bình này bằng dương vô cùng hoặc âm vô cùng thì giá trị trung tâm được định nghĩa là nhỏ nhất (lớn nhất) trong số tất cả các điểm mà đạt được giá trị thành viên tối đa. *Hình 3.2* cho thấy các giá trị trung tâm của

một số tập mờ điển hình. Các điểm giao nhau của một tập mờ là điểm trong *U* mà giá trị thành viên trong *A* bằng 0.5.

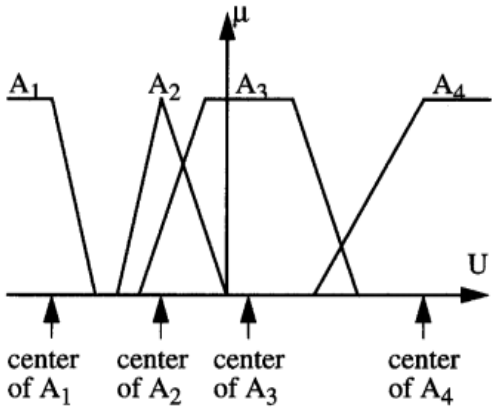
Chiều cao của một tập mờ là giá trị thành viên lớn nhất đạt được bằng bất kỳ điểm nào.

Ví dụ: Chiều cao của tất cả các tập mờ trong hình 3.2 bằng một. Nếu chiều cao của một tập mờ tương đương với một, nó được gọi là một tập mờ bình thường. Do đó, tất cả các tập mờ trong *hình 3.2* là bình thường.

Một giao điểm α của tập mờ *A* là một tập *A*, có chứa mờ
tất cả các yếu tố trong *U* có giá trị thành viên trong *A* lớn hơn tập hoặc bằng *α*:

$$A_\alpha = \{x \in U | \mu_A(x) \geq \alpha\}$$

Ví dụ: Đối với *α=0.3,* điểm cắt *α* của tập mờ *(Hình 3.2)* là tập hợp *[0.7, 0.7],* và *α=0,9,* nó là *[-0.1, 0.1].*



Hình 3.2. Trung tâm của một số tập mờ điển hình

Khi *U* là không gian Euclide R^n, *n* chiều, các khái niệm về tập hợp lồi có thể được tạo ra từ tập mờ. Một tập mờ *A* được gọi là lồi khi và chỉ khi điểm cắt *α* của nó A_0 là một tập lồi cho bất kỳ *α* trong khoảng *(0, 1].* Mệnh đề sau đây đưa ra một định nghĩa tương đương với một tập mờ lồi.

Mệnh đề 3.1. Một tập mờ *A* trong R^n là lồi khi và chỉ khi:

$$\mu_A[\lambda x_1 + (1-\lambda)x_2] \geq min[\mu_A(x_1), \mu_A(x_2)] \quad (3.3)$$

cho tất cả *x1, x2 R và λ [0, 1].*

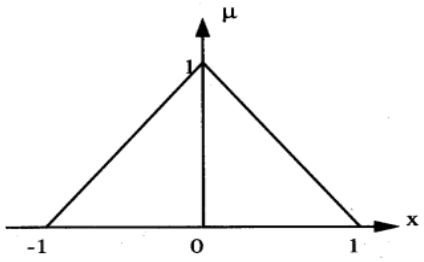
3.3 Hàm thuộc về (hay hàm thành viên)

Làm thế nào để xác định hàm thành viên?

Có hai cách tiếp cận để xác định một hàm thành viên.

+ Phương pháp tiếp cận đầu tiên là sử dụng kiến thức của các chuyên gia của con người, đó là yêu cầu các chuyên gia tên miền để xác định các hàm thành viên. Thông thường, phương pháp này chỉ có thể cung cấp cho một công thức của hàm thành viên; tinh chỉnh là cần thiết

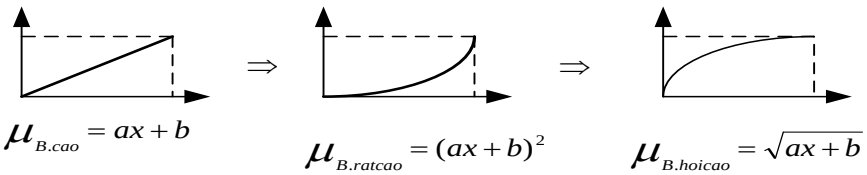
+ Phương pháp thứ hai, sử dụng dữ liệu thu thập từ các cảm biến khác nhau để xác định các hàm thành viên. Trước tiên, xác định cấu trúc của các hàm thành viên và sau đó tinh chỉnh các thông số của các hàm thành viên dựa trên dữ liệu.



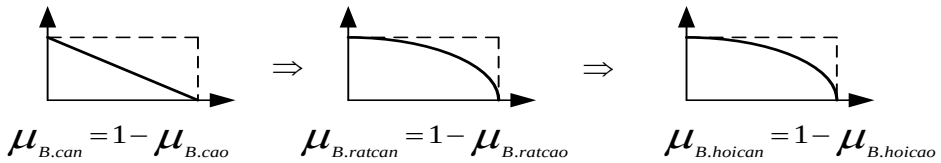
Hình 3.3. Một hàm thành viên có thể mô tả "các con số gần bằng không".

<u>*Đặc điểm*</u>: Các hàm thành viên có 3 dạng:

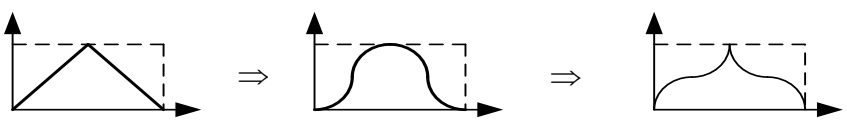
+ Dạng tăng:



+ Dạng giảm:



+ Dạng vừa tăng vừa giảm



Hình 3.4. Một số dạng hàm thành viên

3.4 Hệ mờ là gì?

5.4.1. Tại sao lại gọi là hệ mờ?

Theo từ điển tiếng Anh Oxford, từ "mờ" được định nghĩa là "không rõ ràng, định nghĩa không chính xác, mơ hồ". Chúng ta yêu cầu người đọc bỏ qua định nghĩa này và xem từ "mờ" như một tính từ có tính kỹ thuật. Cụ thể, hệ thống mờ là những hệ thống được xác định một cách chính xác. Chúng ta muốn nhấn mạnh là, mặc dù, hiện tượng mà lý thuyết mờ đặc trưng có thể mờ, nhưng chính bản thân lý thuyết đó là chính xác.

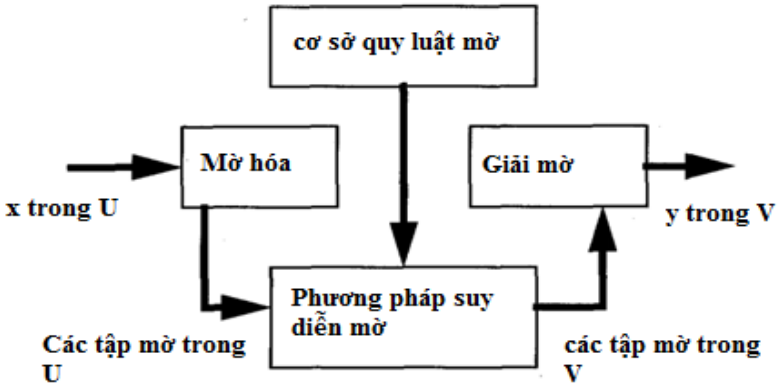
Trong xã hội học, có hai loại biện minh cho lý thuyết mờ. Một là, thế giới thực quá phức tạp. Để mô tả nó chính xác cần phải có một lý thuyết, do đó xấp xỉ (hoặc xâp xỉ mờ hay lý thuyết mờ) phải được giới thiệu để có được một phương pháp hợp lý, dễ hiểu, mô hình hóa được thế giới thực. Hai là, khi bước vào thời đại thông tin, tri thức của con người ngày càng trở nên quan trọng. Chúng ta cần một lý thuyết xây dựng tri thức nhân loại trong một mô hình hệ thống và đặt nó vào hệ thống kỹ thuật, cùng với các thông tin khác như mô hình toán học và các phép đo, cảm biến.

Sự biện minh đầu tiên là chính xác, nhưng không đặc trưng cho tính chất độc đáo của lý thuyết mờ. Trong thực tế, hầu hết các lý thuyết áp dụng cho kỹ thuật đều mô tả thế giới thực một cách gần đúng. Ví dụ, hệ thống thực là phi tuyến, nhưng chúng ta đặt rất nhiều nỗ lực trong việc nghiên cứu các hệ thống tuyến tính. Một nguyên lý kỹ thuật tốt thì có thể chính xác đến mức mà nó đặc trưng cho tính năng chính của thế giới thực, đồng thời dễ theo dõi để phân tích toán học. Trong khía cạnh này, lý thuyết mờ không khác với lý thuyết khác cho các lĩnh vực kỹ thuật.

Sự biện minh thứ hai đặc trưng cho tính độc đáo của lý thuyết mờ và chứng minh sự tồn tại của lý thuyết mờ như một nhánh độc lập trong kỹ thuật. Theo nguyên tắc chung, một lý thuyết cho các hệ kỹ thuật tốt cần có khả năng sử dụng tất cả các thông tin sẵn có một cách hiệu quả. Đối với nhiều hệ thống thực, thông tin quan trọng đến từ hai nguồn: một nguồn từ các chuyên gia, mô tả kiến thức của họ về hệ thống bằng ngôn ngữ tự nhiên; nguồn khác là phép đo hay cảm biến và các mô hình toán học có nguồn gốc dựa trên các định luật vật lý. Do đó, nhiệm vụ quan trọng là kết hợp hai loại thông tin này để đưa vào thiết kế hệ thống. Để đạt được sự kết hợp này, câu hỏi quan trọng là làm thế nào để xây dựng tri thức của người vào một khuôn dạng thống nhất sử dụng để xây dựng các phép đo và mô hình toán học. Nói cách khác, làm thế nào để mô phỏng tri thức của người bằng một công thức toán học. Về cơ bản, những gì một hệ mờ cần có là thực hiện phép chuyển đổi này. Để hiểu phép chuyển đổi này được thực hiện như thế nào, đầu tiên chúng ta phải biết hệ mờ là gì.

Hệ thống mờ là những hệ thống dựa trên tri thức hay dựa trên quy luật mà trung tâm là một cơ sở tri thức bao gồm cái gọi là mờ cấu trúc *IF - THEN*.

Ví dụ: *IF* "Trời mưa" *THEN* "đất ướt".



Hình 3.5. Cấu hình cơ bản của hệ thống mờ với mờ hóa và giải mờ

Hệ mờ hoạt động như sau:

Mờ hóa các biến vào: Vì nhiều luật cho dưới dạng các biến ngôn ngữ với các từ thông thường. Như vậy với những giá trị quan sát được, đo được cụ thể, để có thể tham gia vào quá trình suy diễn thì cần thiết phải mờ hóa

Có thể định nghĩa mờ hóa là: ánh xạ từ không gian các giá trị quan sát được vào không gian các từ trên không gian nền của các biến ngôn ngữ.

Áp dụng các toán tử mờ (*AND, OR, NOT*) cho các giả thiết của từng luật

Giải mờ kết quả để tìm được một số rõ, một số cụ thể: Đây là khâu thực hiện quá trình xác định một giá trị rõ có thể chấp nhận được làm đầu ra của hàm thuộc của giá trị mờ đầu ra.

3.5 Các phép tính mờ

Định nghĩa 3.2

Sự bằng nhau, chứa nhau, bù, hợp, và giao của hai tập mờ *A* và *B* được định nghĩa như sau:

A và B bằng nhau khi và chỉ khi $\mu_A(x) = \mu_B(x)$ *với mọi* $x \in U$.

B chứa A, ký hiệu là $A \subset B$, *khi và chỉ khi* $\mu_A(x) \leq \mu_B(x)$ *với mọi* $x \in U$.

1) Phép bù

Bù của *A* là một tập mờ phủ định *A* trong *U*. Hàm thành viên của bù được định nghĩa:

$$\mu_{\bar{A}}(x) = 1 - \mu_A(x) \tag{3.4}$$

2) Phép hợp

Hợp của *A* và *B* là một tập mờ trong *U*, ký hiệu là *AUB*, trong đó hàm thành viên của phép hợp được xác định*:*

$$\mu_{A\cup B}(x) = max[\mu_A(x), \mu_B(x)] \tag{3.5}$$

3) Phép giao

Giao của *A* và *B* là một tập mờ $A \cap B$ trong *U* với hàm thành viên:

$$\mu_{A\cap B}(x) = min[\mu_A(x), \mu_B(x)] \tag{3.6}$$

Người đọc có thể tự hỏi tại sao sử dụng "max" cho hợp và "min" cho giao. Một cách trực quan hấp dẫn của việc xác định hợp như sau: Sự kết hợp của *A* và *B* là tập mờ nhỏ nhất có chứa cả *A* và *B.* Chính xác hơn, nếu *C* là bất kỳ tập mờ có chứa cả *A* và *B,* thì nó cũng chứa hợp của *A* và *B*. Để chứng minh định nghĩa trực quan hấp dẫn này là tương đương với *(3.5).* Lưu ý, *AUB* theo quy định của *(3.5)* có chứa cả *A* và *B* vì:

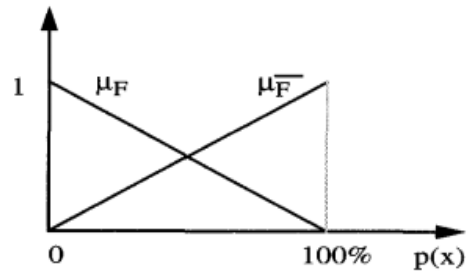
$$max[\mu_A, \mu_B] \geq \mu_A \text{ and } max[\mu_A, \mu_B] \geq \mu_B.$$

Hơn nữa, nếu *C* là tập mờ bất kỳ có chứa cả *A* và *B*, thì

$$\mu_C \geq \mu_A \text{ and } \mu_C \geq \mu_B.$$

Do đó: $\mu_c \geq max\ [\mu_a, \mu_b] = \mu_{A\cup B}$. (3.7)

Có nghĩa là, *A U B* theo quy luật của *(3.5)* là tập mờ nhỏ nhất có chứa cả *A* và *B*. Các giao theo quy luật của *(3.6)* có thể được được giải thích tương tự.



Hình 3.5. Các hàm thành viên cho *F* phủ định và *F*.

3.6 Mờ hóa

Được định nghĩa là sự ánh xạ từ các tập giá trị *x* thuộc *U* và là tập con của *R* thành tập các giá trị mờ *A* ở trong *U*.

Nguyên tắc chung việc thực hiện mờ hóa:

- Từ tập *x* giá trị đầu vào sẽ tạo ra tập giá trị mờ *A* với hàm thuộc có giá trị đủ rộng tại các điểm rõ *x*
- Nếu có nhiễu ở đầu vào thì mờ hóa sẽ góp phần khử nhiễu
- Việc mờ hóa cần làm đơn giản cho việc tính toán sau này

Thông thường, có 3 phương pháp mờ hóa:

- Mờ hóa đơn vị: lấy từ các điểm có giá trị thực *x* thuộc *U*; lấy giá trị đơn vị của tập mờ *A*
- Mờ hóa Gaus: lấy từ các điểm có giá trị thực *x* thuộc tập *U* và lấy giá trị đơn của tập mờ *A* thuộc hàm Gaus.
- Mờ hóa hình tam giác: lấy từ các điểm có giá trị thực *x* thuộc *U* và lấy giá trị đơn của tập mờ *A* thuộc hàm hình tam giác.

Ta thấy rằng mờ hóa đơn vị tính toán về sau đơn giản hơn so với 2 cách còn lại nhưng nó lại không khử được nhiễu đầu vào. Mờ hóa Gaus và hình tam giác cho tính toán phức tạp hơn nhưng khử được nhiễu đầu vào.

3.7 Giải mờ

Giải mờ là phép ngược lai với mờ hóa, tức là chuyển giá trị logic tính toán được thành giá trị thực. Để giải mờ có 3 điều cần lưu ý khi thực hiện giải mờ:

- Tính hợp lý của kết quả: Điểm rõ *y* thuộc tập *V* đại diện cho kết quả của tập mờ *B*.
- Tính toán đơn giản: Để tính toán nhanh vì các bộ điều khiển mờ thường làm việc trong thời gian thực.
- Tính liên tục: Mọi sự thay đổi nhỏ trong tập mờ *B* chỉ làm thay đổi nhỏ kết quả giải mờ.

Có 3 phương pháp giải mờ cơ bản: phương pháp cực đại, phương pháp tâm của trọng lực và phương pháp tìm tâm trung bình

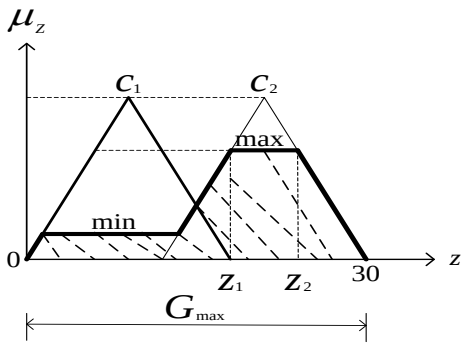
❖ *Phương pháp cực đại*

+ *Bước 1*: Xác định miền chứa giá trị rõ đầu ra (*G*)

$$G = \{z \in Z / \mu_{bom}(z) = max\}$$

+ *Bước 2*: Xác định giá trị y rõ

Ví dụ:



$$Z = \frac{z_1 + z_2}{2}$$

❖ *Phương pháp trọng tâm*

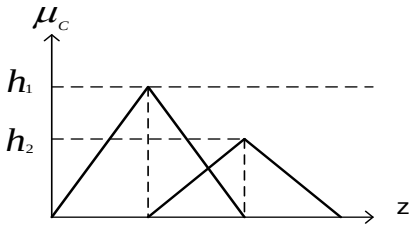
$$Z = \frac{\int_S z\mu_c(z)d(z)}{\int_S \mu_c(z)dz} = \frac{\int_0^{z_1} z_1\mu_{c1}(z)dz + \cdots + \int_0^{z_1} z_7\mu_{c7}(z)dz + \cdots}{\int_{s1} \mu_{c1}(z)dz + \cdots + \int_S \mu_c(z)dz + \cdots}$$

$$= \frac{\sum_{Si} z_i\mu_{ci}(z)}{\sum_{Si} \mu_{ci}(z)}$$

❖ *Phương pháp trung bình tâm*

$$y = \frac{\sum_{i=1}^{m} y_{ih_i}}{\sum h_i}$$

Ví dụ:



$$z = \frac{h_1z_1 + h_2z_2}{h_1 + h_2}$$

Bài toán minh họa

Cho bể nước cao 10; hồ trên tầng thượng cao 2m; 1 máy bơm nước bơm từ bể vào hồ. Hỏi bơm bao lâu thì hồ đầy? Biết bơm lâu mất 30 phút, bơm hơi lâu là 20 phút và bơm vừa mất 15 phút.

Bài này trong thực tế sẽ không giải được vì vẫn thiếu rất nhiều sự kiện nhưng vẫn có 1 cách để giải quyết bài toán. Đó là dùng hệ mờ để giải. Tuy nhiên, dùng hệ mờ chỉ tính được kết quả gần đúng không thể chính xác hoàn toàn.

Tóm tắt bài toán

+ Bể nước cao 10m có 3 mức nước: $x\begin{cases} cao: \mu_{cao}(x) \\ lưng: \mu_{lưng}(x) \\ cạn: \mu_{cạn}(x) \end{cases}$

+ Hồ nước cao 2m có 3 mức nước: $y\begin{cases} đầy: \mu_{đầy}(y) \\ vừa: \mu_{vừa}(y) \\ cạn: \mu_{cạn}(y) \end{cases}$

+ Bơm có 3 trạng thái: $z\begin{cases} bơm\ lâu\ 30' \\ bơm\ hơi\ lâu\ 20' \\ bơm\ vừa\ 15' \end{cases}$

Cho $y_0 = 1^m, x_0 = 2^m$. Hãy tính thời gian cần bơm bằng hệ mờ để hồ đầy.

Nhận xét:

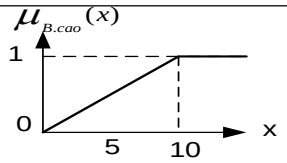
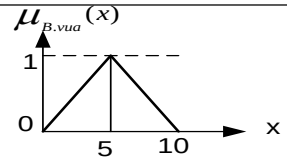
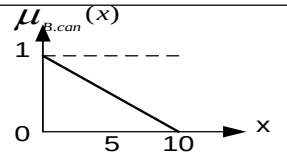
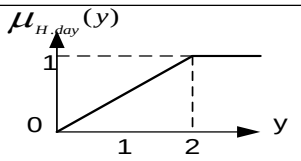
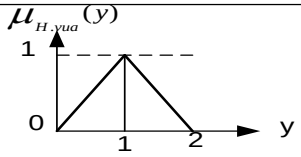
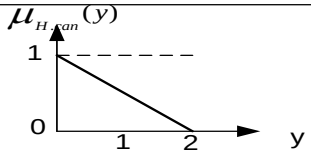
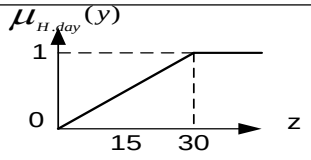
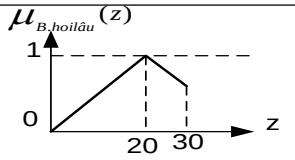
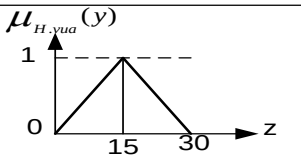
Bài toán này theo tính toán thông thương không giải được do thiếu dữ liệu để tính dung tích của bể, của hồ và thiếu thông tin về công suất của máy bơm. Nhưng có thể dùng hệ mờ để xấp xỉ thời gian cần bơm

Bước 1: Xây dựng biến ngôn ngữ và luật (trạng thái)

X(bể) \ Y(hồ)	Y là đầy	Y là vừa	Y là cạn
X là cao	0	Z là bơm vừa	Z là bơm lâu
X là lưng	0	Z là bơm vừa	Z là bơm hơi lâu
X là cạn	0	0	0

Luật 1: Nếu X là Bồn Cạn and Y là Bể Đầy
Thì Z là Bơm Lâu
Luật 2: Nếu X là Bồn Cạn and Y là Bể Vừa
Thì Z là Bơm Hơi Lâu
Luật 3: Nếu X là Bồn Lưng and Y là Bể Vừa
Thì Z là Bơm Vừa
Luật 4: Nếu X là Bồn Lưng and Y là Bể Đầy
Thì Z là Bơm Vừa

Bước 2: Xây dựng các hàm thành viên (giả thiết các hàm là tuyến tính)

Nhóm	Hàm thành viên	Hàm	Đồ thị
Bể (x)	$\mu_{B.cao}(x)$	$\mu_{B.cao}(x) = \frac{1}{10}x$	
	$\mu_{B.vua}(x)$	$\mu_{B.vua}(x) = \begin{cases} \frac{1}{5}x & \text{Nếu x=[0;5]} \\ -\frac{1}{5}x + 2 & \text{Nếu x=[5;10]} \end{cases}$	
	$\mu_{B.can}(x)$	$\mu_{B.cân}(x) = -\frac{1}{10}x + 1$	
Hồ (y)	$\mu_{H.day}(y)$	$\mu_{H.day}(y) = \frac{1}{2}x$	
	$\mu_{H.vua}(y)$	$\mu_{H.vua}(y) = \begin{cases} y & \text{Nếu y=[0;1]} \\ -y + 2 & \text{Nếu y=[1;2]} \end{cases}$	
	$\mu_{H.can}(y)$	$\mu_{H.can}(y) = -\frac{1}{2}y + 1$	
Bơm (z)	$\mu_{B.lâu}(z)$	$\mu_{B.lâu}(z) = \frac{1}{30}z$	
	$\mu_{B.hoilâu}(z)$	$\mu_{B.hoilâu}(z) = \begin{cases} \frac{1}{20}z & \text{Nếu z=[0;20]} \\ -\frac{1}{20}z + 2 & \text{Nếu z=[20;30]} \end{cases}$	
	$\mu_{B.vua}(z)$	$\mu_{B.vua}(z) = \begin{cases} \frac{1}{15}z & \text{Nếu z=[0;15]} \\ -\frac{1}{15}z + 2 & \text{Nếu z=[15;30]} \end{cases}$	

Bước 3: Viết luật.

Từ bảng trạng thái, có 4 luật

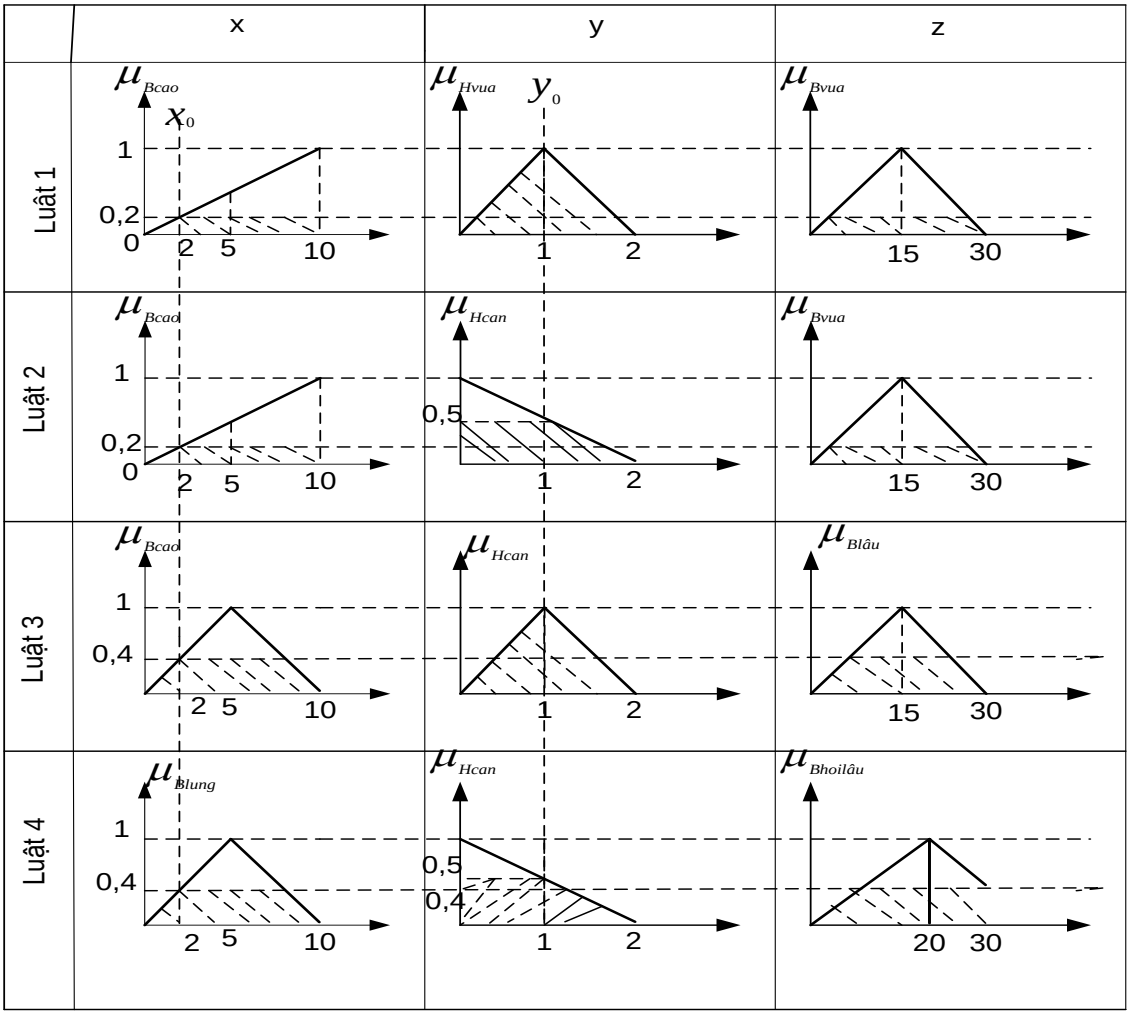
Luật 1: *IF*(*x* là cao) ∧ (*y* là vừa) *THEN* (z là bơm vừa)

Luật2: *IF* (x là lưng) ∧ (*y* là vừa) *THEN* (*z* là bơm vừa)

Luật 3: *IF* (*x* là cao) ∧ (*y* là cạn) *THEN* (*z* là bơm lâu)

Luật 4: *IF* (*x* là lưng) ∧ (*y* là cạn) *THEN* (z bơm hơi lâu)

Bước 4: Tính toán mờ, hệ mờ



Bước 5: giải mờ.

Để đơn giản trong tính toán, ta có thể sử dụng phương pháp trung bình Max.

Trên đây, chúng ta đã sử dụng bài toán đơn giản để mô tả hệ mờ. Hoàn toàn có thể tính được lượng thời gian cần bơm.

CÂU HỎI VÀ BÀI TẬP

1. Sử dụng công cụ FuzzyToolbox trong MATLAB, xây dựng các hầm thuộc về cho ba biến bơn, bể, hồ ở ví dụ chương 3.
2. Cho 1 điều hòa (có quạt đẩy không khá), 2 cảm biến nhiệt độ. Nhiệt độ trong nhà T_t; Nhiệt độ ngoài trời Nhiệt độ trong nhà T_n ; $0^{0C} \leq T_t, T_n \leq 50^{0C}$. Quạt có tốc độ (miền xác định): $0 \leq v \leq 400$ $^{vòng}/_{phút}$

+ Miền giá trị vào $\begin{cases} T_t, T_n \leq 20^0 : Nhiệt\ độ\ thấp \\ T_t, T_n = 25^0 : Nhiệt\ độ\ vừa \\ T_t, T_n \geq 30^0 : Nhiệt\ độ\ cao \end{cases}$;

+ Miền giá trị ra $\begin{cases} v \leq 100 \frac{vòng}{phút} : chậm \\ v = 200 \frac{vòng}{phút} : vừa \\ v \geq 300 \frac{vong}{phút} : nhanh \end{cases}$

a) Hãy xây dựng các hàm thành viên Nhiệt độ ngoài trời (x); Nhiệt độ trong nhà (y); Tốc độ quạt (*v*)

b) Giả sử cho vào các giá trị thật $\begin{cases} x_0 : T_n : 24^0 \\ y_0 : T_t : 28^0 \end{cases}$ và 4 luật điều khiển

+ IF(x là thấp) ∧ (y là vừa) THEN (z là chậm);
+ IF(x là thấp) ∧ (y là cao) THEN (z là nhanh)
+ IF(x là vừa) ∧ (y là vừa) THEN (z là chậm);
+ IF(x là vừa) ∧ (y là cao) THEN (z là vừa)

Dùng suy diễn Mamdani và phương pháp giải mờ "trung bình MAX" tính tốc độ quạt.

CHƯƠNG 4:
MẠNG NƠ-RON NHÂN TẠO

Ngành khoa học nơ ron đã có từ lâu, nhưng phỏng theo mạng nơ ron để áp dụng vào các ngành khoa học kỹ thuật khác nhau đang là vấn đề nóng hổi của thế giới và là thời sự ở nước ta đặc biệt trong các lĩnh vực tin học, viễn thông, điều khiển v.v... Theo *Wiener*, khoa học nơ ron cùng với thông tin và điều khiển là một trong ba lĩnh vực dưới ngọn cờ chung là điều khiển học (*Cybernetics*).

Ngày nay mạng nơ ron được bắt đầu ứng dụng trong một số lĩnh vực như nhận dạng, điều khiển, y tế, viễn thông…

4.1 Nguồn gốc của mạng nơ ron

4.1.1. Quá trình phát triển và nghiên cứu mạng nơ ron

Nghiên cứu bộ não con người, cụ thể là tế bào thần kinh (*nơ ron*) là ước muốn từ lâu của nhân loại. Từ đó, các nhà khoa học đã không ngừng nghiên cứu và tìm hiểu về mạng nơ ron. Với khoảng hơn 15 tỷ nơ ron ở não người, nơ ron có thể nhận hàng ngàn hàng vạn tín hiệu từ các khớp thần kinh và được coi là một cơ chế sinh vật phức tạp nhất. Não người có khả năng giải quyết những vấn đề như: nghe, nhìn, nói, hồi ức thông tin, phân biệt các mẫu mặc dầu sự kiện bị méo mó, thiếu hụt. Não thực hiện những nhiệm vụ như vậy nhờ các phần tử tính toán là nơ ron. Não phân bố việc xử lí cho hàng tỷ nơ ron có liên quan, điều khiển các mối liên hệ giữa các nơ ron đó. Nơ ron không ngừng nhận và truyền thông tin lẫn nhau. Các nơ ron tự liên kết với nhau thành mạng trong xử lý. Mỗi mạng gồm hàng vạn các phần tử nơ ron khác nhau và mỗi phần tử nơ ron có khả năng liên kết với hàng ngàn nơ ron khác. Lý thuyết về mạng nơ ron đã hình thành và đang phát triển, đặc biệt là nghiên cứu các ứng dụng của chúng.

Quá trình nghiên cứu và phát triển mạng nơ ron nhân tạo có thể được chia thành 4 giai đoạn như sau:

- *Giai đoạn thứ nhất:* Giai đoạn thứ nhất có thể tính từ nghiên cứu của *William* 1890 về tâm lý học với sự liên kết các nơ ron thần kinh. Từ năm 1940 *Mc.Culloch* và *Pitts* đã cho biết: nơ ron có thể được mô hình hoá như thiết bị ngưỡng (giới hạn) để thực hiện các phép tính logic. Cũng thời gian đó *Wiener* đã xét các mối liên hệ giữa nguyên lý phản hồi và chức năng bộ não.
- *Giai đoạn thứ hai:* Giai đoạn thứ hai vào những năm 1960, gần như đồng thời một số mô hình nơ ron hoàn hảo hơn đã được đưa ra, đó là mô hình:

- Perceptron của Rosenbatt

- Adaline của Widrow
- Ma trận học của Stinbuck

trong đó, mô hình Perceptron rất được quan tâm vì nguyên lý đơn giản, nhưng nó cũng có hạn chế vì như Minsky và Papert đã chứng minh nó không dùng được cho các hàm logic phức. Adaline là mô hình tuyến tính, tự chỉnh, được dùng rộng rãi trong điều khiển thích nghi, tách nhiễu, phát triển và ứng dụng cho đến ngày nay.

- *Giai đoạn thứ ba:* Giai đoạn thứ ba có thể được tính là khoảng đầu những năm 80. Những đóng góp lớn cho mạng nơ ron trong giai đoạn này phải kể đến *Grossberg, Kohonen và Hopfield*. Đóng góp lớn của *Hopfield* gồm hai mạng phản hồi: mạng rời rạc 1982 và mạng liên tục năm 1984. Đặc biệt, ông đã dự kiến nhiều khả năng tính toán lớn của mạng mà một nơ ron không có khả năng đó. Cảm nhận của Hopfied đã được Rumelhart, Hinton đề xuất thuật toán sai số truyền ngược nổi tiếng để huấn luyện mạng nơ ron nhiều lớp nhằm giải bài toán mà mạng khác không thực hiện được.

- *Giai đoạn thứ tư:* Giai đoạn thứ tư là từ năm 1987 đến nay, hàng năm thế giới đều mở hội nghị toàn cầu chuyên nghành nơ-ron IJCNN (*Internatonal Joint Conference on Neral Networks*). Các công trình nghiên cứu để hoàn thiện thêm về lý thuyết mạng nơ ron như: mở rộng, hoàn thiện các lớp mạng, phân tích ổn định, kết hợp lý thuyết mạng nơ ron với các lý thuyết khác cũng không ngừng được đưa ra. Hàng loạt lĩnh vực khác như: kỹ thuật tính, tối ưu, sinh học, y học, thống kê, giao thông, hoá học, truyền thông… đã đóng góp nhiều công trình nghiên cứu và ứng dụng mạng nơ ron vào lĩnh vực của mình và đem lại những kết quả đáng khích lệ. Ở trong nước, mạng nơ ron được nghiên cứu từ những năm 1980, đi vào ứng dụng trong các lĩnh vực tin học viễn thông, đo lường điều khiển… Một số chip nơ ron đã được dùng trong kỹ thuật lọc và một số ứng dụng khác.

4.1.2. Mô hình tổng quát của nơ ron sinh vật

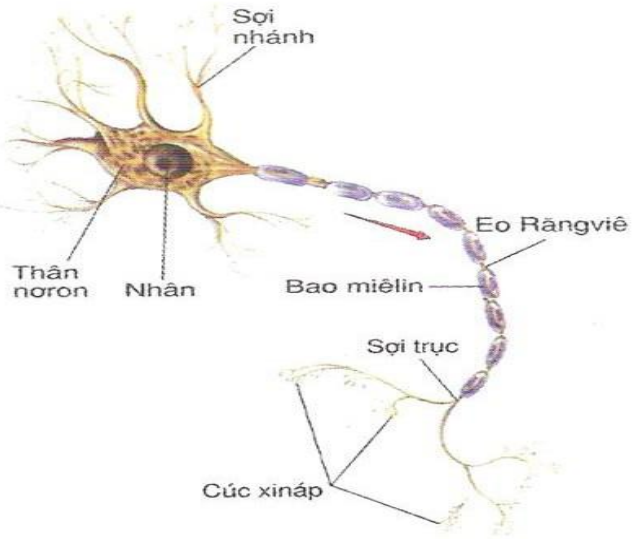
Nơ ron sinh vật có nhiều dạng khác nhau như dạng hình tháp, dạng tổ ong, dạng rễ cây. Tuy nhiên, chúng có cấu trúc và nguyên lý hoạt động chung. Tế bào nơ ron gồm 4 phần cơ bản:

- *Các nhánh và rễ :* là các bộ phận nhận thông tin, các đầu nhạy hoặc các đầu ra của các nơ ron khác bám vào rễ hoặc nhánh của một nơ ron. Khi các đầu vào từ ngoài này có sự chênh lệch về nồng độ K+, Na+ hay Cl- so với nồng độ bên trong của nó thì xảy ra hiện tượng thấm từ ngoài vào trong thông qua một cơ chế màng thấm đặc biệt. Hiện tượng thẩm thấu như vậy tạo nên một cơ chế truyền đạt thông tin với hàng ngàn hàng vạn lối vào trên một nơ ron sinh vật, ứng với hàng nghìn hàng vạn liên kết khác nhau. Mức độ thẩm thấu được đặc trưng bởi cơ chế màng tượng trưng bằng một tỷ lệ. Tỷ lệ đó được gọi là tỷ trọng hay đơn giản gọi là trọng (*Weight*).

- *Thân thần kinh (Soma):* chứa các nhân và cơ quan tổng hợp protein. Các ion vào được tổng hợp và biến đổi. Khi nồng độ các ion đạt đến một giá trị nhất định, xảy ra quá trình phát xung (hay kích thích). Xung đó được phát ở các đầu ra của nơ ron. Dây dẫn đầu ra xung được gọi là dây thần kinh.

- *Dây thần kinh (Axon):* là đầu ra. Đó là phương tiện truyền dẫn tín hiệu. Dây thần kinh được cấu tạo gồm các đốt và có thể dài từ micro mét đến vài mét tuỳ từng kết cấu cụ thể. Đầu ra này có thể truyền tín hiệu đến các nơ ron khác.

- *Khớp thần kinh (Synape):* là bộ phận tiếp xúc của đầu ra nơ ron với rễ, nhánh của các nơ ron khác. Chúng có cấu trúc màng đặc biệt để tiếp nhận các tín hiệu (hình 4.1) khi có sự chênh lệch về nồng độ ion giữa bên trong và bên ngoài. Nếu độ lệch về nồng độ càng lớn thì việc truyền các ion càng nhiều và ngược lại. Mức độ thẩm thấu của các ion có thể coi là một đại lượng thay đổi tuỳ thuộc vào nồng độ như một giá trị đo thay đổi được gọi là trọng.



Hình 4.1 Mô hình nơ ron sinh học tổng quát

4.1.3. Cấu trúc mạng nơ ron sinh vật

Mạng nơ ron sinh vật tổ chức thành từng lớp (layer). Ta có thể phân loại các mạng nơ ron nhân tạo (phỏng sinh học) như sau:

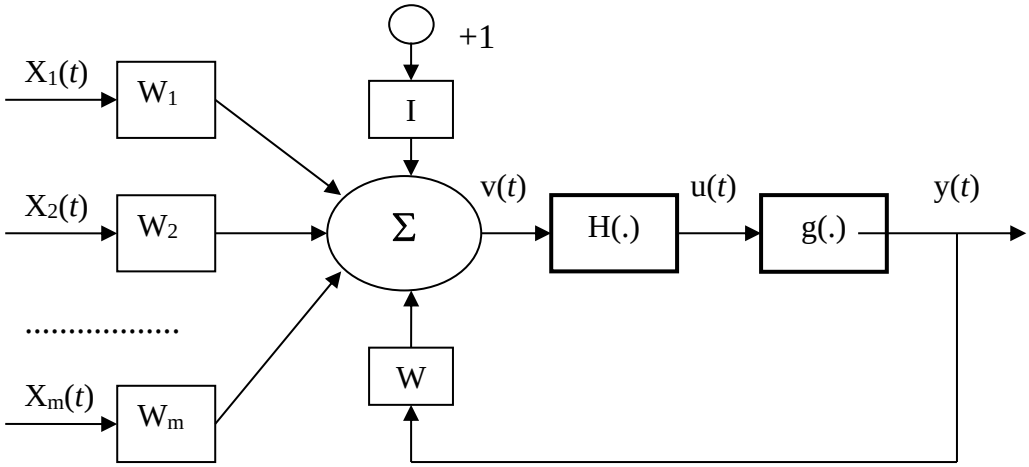
- *Mạng một lớp:* là tập hợp các phần tử nơ ron có đầu vào và đầu ra trên mỗi một phần tử. Nếu mạng nối đầu ra của các phần tử này với đầu vào của phần tử kia gọi là mạng tự liên kết (Autoassociative).

- *Mạng hai lớp:* gồm một lớp đầu vào và một lớp đầu ra riêng biệt.

- *Mạng nhiều lớp:* gồm một lớp đầu vào và một lớp đầu ra riêng biệt. Các lớp nằm giữa lớp đầu vào và lớp đầu ra gọi là lớp ẩn (hidden layers).
- *Mạng truyền thẳng:* là mạng hai hay nhiều lớp mà quá trình truyền tín hiệu từ đầu ra lớp này đến đầu vào lớp kia theo một hướng.
- *Mạng truyền ngược:* là mạng mà trong đó một hoặc nhiều đầu ra của các phần tử lớp sau truyền ngược tới đầu vào của lớp trước.
- *Mạng tự tổ chức:* là mạng có khả năng sử dụng những kinh nghiệm của quá khứ để thích ứng với những biến đổi của môi trường (không dự báo trước). Loại mạng này thuộc nhóm hệ học, thích nghi không cần có tín hiệu chỉ đạo từ bên ngoài.

4.2 Mô hình mạng nơ ron nhân tạo và luật học

4.2.1. Mô hình tổng quát của nơ ron nhân tạo



Hình 4.2. Mô hình một Nơ ron nhân tạo

Từ mô hình nơ ron sinh vật, dễ dàng phỏng thành mô hình nơ ron nhân tạo. Mô hình một phần tử nơ ron nhân tạo được xây dựng từ ba thành phần chính: i) bộ tổng các liên kết đầu vào, ii) phần động học tuyến tính, iii) phần phi tuyến không động học (*hình 4.2*)

- ***Bộ tổng liên kết:*** Bộ tổng hợp các liên kết đầu vào của một phần tử nơ ron có thể mô tả như sau:

$$\int v(t) = \sum_{k=1}^{m} w_k x_k(t) - \theta \qquad (4.1)$$

trong đó:

- *v(t):* tổng tất cả các đầu vào mô tả toàn bộ thế năng tác động ở thân nơ ron;

- $x_k(t)$: các đầu vào ngoài, mô tả tín hiệu vào từ các đầu nhạy thần kinh hoặc từ các nơ ron khác đưa vào;

- w_k: trọng liên kết vào ngoài, là hệ số mô tả mức độ liên kết giữa các đầu vào ngoài tới nơ ron hiện tại, m là số đầu vào; $k=1,...,m;$

- $y(t)$: đầu ra nơ ron mô tả tín hiệu đưa ra;

- θ : hằng số, còn gọi là ngưỡng, xác định ngưỡng kích thích hay ức chế.

4.2.2. Phần động học tuyến tính

Đầu vào của phần động học là ***v(t)***. Đầu ra của nó ***u(t)*** gọi là đầu ra tương tự. Hàm truyền tương ứng của phần động học tuyến tính có thể mô tả dưới dạng:

$$U(s) = H(s)\, V(s) \qquad (4.2)$$

$$H(s) = \frac{U(s)}{V(s)}$$

Bảng 4.1. Một số hàm H(s) thường dùng cho mô hình nơ ron nhân tạo

H(s)	*1*	$\frac{1}{s}$	$\frac{1}{1-sT}$	*Exp(-sT)*
vào/ra	$u(t) = v(t)$	$\frac{du(t)}{dt} = v(t)$	$T\frac{du(t)}{dt} + u(t) = v(t)$	$u(t)=v(t-T)$

Một số dạng hàm khác cũng được sử dụng như: dạng hàm Gauss, hàm Logarit, hàm mũ…

- ***Các hàm phi tuyến:*** phần sử dụng hàm quan hệ phi tuyến *f(.)* cho đầu ra *y*, để chặn tín hiệu ở đầu ra.

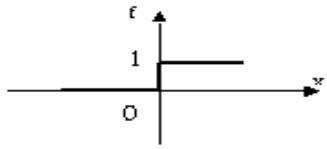
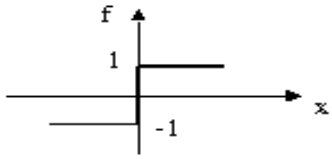
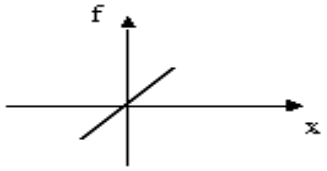
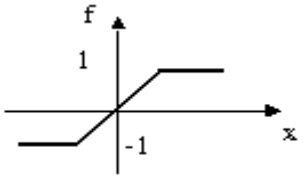
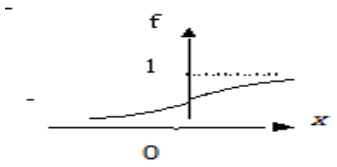
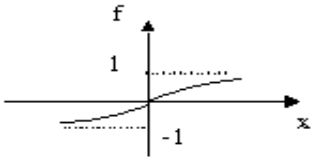
$$\boldsymbol{y} = f(u(t)) = f\left(\sum_{i=1}^{m} x_i(t) w_i - \theta\right) \qquad (4.3)$$

- Hàm phi tuyến ở đây có thể chia thành: nhóm hàm bước nhảy và nhóm hàm liên tục.

4.2.2 Mạng nơ ron nhân tạo

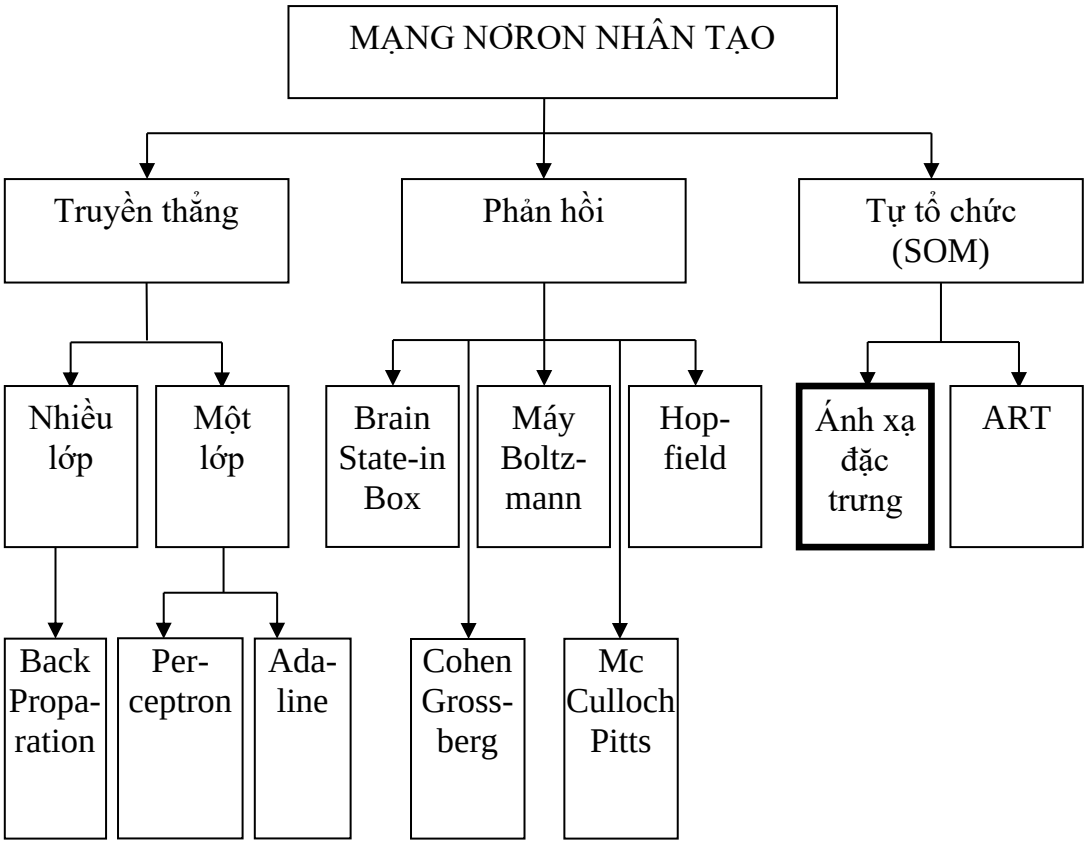
Cũng giống các nơ ron sinh vật, các nơ ron nhân tạo có thể liên kết với nhau để tạo thành mạng nơ ron nhân tạo (Artificiac Nơ ron Network). Có nhiều cách kết hợp các nơ ron nhân tạo thành mạng, mỗi cách kết hợp tạo thành một lớp mạng nơ ron nhân tạo khác nhau.

Bảng 4.2. Một số hàm phi tuyến thường được sử dụng trong các mô hình nơ ron

Tên hàm	Công thức	Đặc tính
Hàm Bước nhảy đơn vị	$f(x)=\begin{cases}1 & if\ x\geq 0\\ 0 & if\ x<0\end{cases}$	
Hard Limiter Hay hàm sgn(.)	$f(x)=sgn(x)=\begin{cases}1 & if\ x\geq 0\\ -1 & if\ x<0\end{cases}$	
Hàm tuyến tính	$f(x)=x$	
Hàm tuyến tính bão hoà đối xứng	$f(x)=\begin{cases}1 & if\ x>1\\ x & if\ -1\leq x\leq 1\\ -1 & if\ x<0\end{cases}$	
Hàm Sigmoid	$f(x)=\frac{1}{1+e^{-\lambda x}}$	
Hàm Sigmoid lưỡng cực		

Phân loại mạng nơ ron nhân tạo

Có nhiều cách để phân loại mạng nơ ron nhân tạo i)Dựa vào số lượng lớp có trong mạng ta phân thành: Mạng một lớp; Mạng nhiều lớp. ii)Dựa vào đường truyền tín hiệu trong mạng ta phân thành: Mạng truyền thẳng; Mạng phản hồi; Mạng tự tổ chức. (Một kiểu điển hình như *hình 4.3).*



Hình 4.3 Phân loại mạng nơ ron

Trong mỗi lớp mạng lại có nhiều mạng với các tên gọi và các đặc trưng khác nhau.

1. Khái quát về luật học

Khái niệm học trong mạng nơ ron được hiểu theo hai nghĩa: học về cấu trúc và học về tham số (Parameter Learning). Trong phạm vi chương trình này tập trung cho học tham số. Tư tưởng của việc học tham số là thay đổi, cập nhật các trọng liên kết. Hầu hết các luật học tồn tại thuộc kiểu học tham số. Trong phần này, các kiến trúc mạng điển hình đưa ra cũng thuộc dạng học tham số. Thông thường, luật học tham số được chia thành 3 dạng chính: học giám sát, học không giám sát và học củng cố.

a) Học có giám sát (Supervised Learning): còn gọi là học tín hiệu chỉ đạo

Trong những thập kỷ qua, một loạt các thuật học đã được nghiên cứu để xấp xỉ trọng liên kết. Các thuật học đó có thể chia làm hai nhóm theo sự tồn tại của tín hiệu chỉ đạo (giám sát) hay không (không giám sát): học có tín hiệu chỉ đạo (thầy giáo) hoặc học không có tín hiệu chỉ đạo. Một số các thuật học mở rộng cũng được phát triển và dựa theo các luật học này.

Bảng 4.3. So sánh một số mạng nơ ron

	Perceptron	MLP	BP	Hopfield	SOM
Tác giả, năm đề xuất	Rosenblatt, 1958	Minsky Papert, 1969	Rumelhart và R.J. Williams, 1986	Hopfield, 1982	T. Kohonen, 1982
Phân loại	Truyền thẳng	Truyền thẳng	Truyền thẳng	Phản hồi	Truyền thẳng, Phản hồi
Số lớp	1 vào,1 ra	3 lớp trở lên	3 lớp trở lên	1 lớp	1 vào,1 ra
Kiểu dữ liệu	binary	Binary	Binary	binary	binary, real
Hàm truyền*	hard limiter	hard limiter sigmoid	sigmoid	signum hard limiter	sigmoid
Phương pháp học	Có giám sát	Có giám sát	Có giám sát	Không giám sát	Không giám sát
Thuật học	Luật Hebb	Delta , BP	BP	Delta, Luyện thép	SOM
Ứng dụng chủ yếu	Phép logic; Phân lớp mẫu	Phép logic phức tạp; Phân lớp	Phép logic phức tạp; Phân lớp mẫu; Phân tích tiếng nói	Liên kết mẫu; Tối ưu	Phân lớp; Tối ưu; Đồ thị

Trong học giám sát, tại mỗi thời điểm có đầu vào mạng nơ ron thì đầu ra mong muốn của hệ sẽ được cho sẵn. Nói một cách rõ ràng hơn: mạng được cung cấp một tập các mẫu $(x^{(1)},d^{(1)})$, $(x^{(2)},d^{(2)})$, . . . $(x^{(k)},d^{(k)})$ là các cặp đầu vào - đầu ra mong muốn. Khi một đầu vào $x^{(k)}$ được đưa vào mạng, đầu ra mong muốn $d^{(k)}$ cũng được đưa vào mạng. Như *hình 4.4(a),* sai khác giữa giá trị đầu ra thực sự $y^{(k)}$ và đầu ra mong muốn $d^{(k)}$ sẽ là cơ sở tạo tín hiệu lỗi để mạng sửa đổi trọng sao cho đầu ra thực sự gần với đầu ra mong muốn hơn.

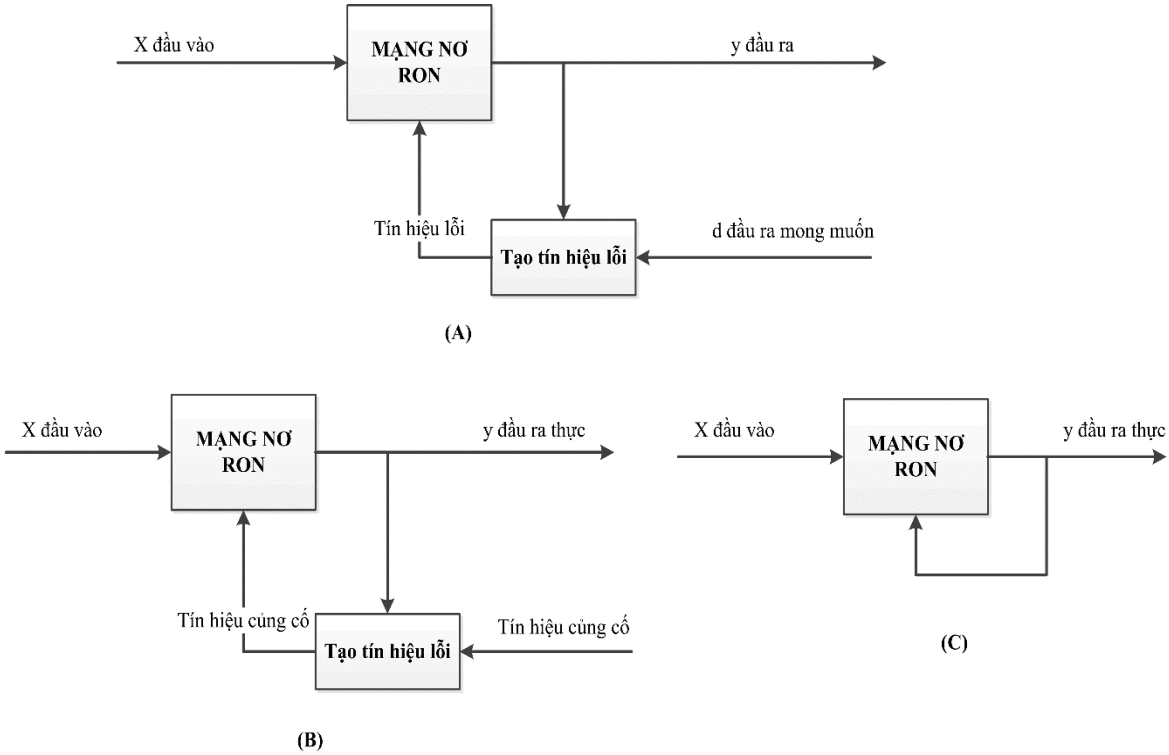
b) Học củng cố (Reinforcement Learning)

Trong luật học có giám sát, các giá trị đầu ra đích được biết chính xác đối với mỗi đầu vào. Tuy nhiên, trong một số trường hợp, chỉ biết được ít thông tin chi tiết, chẳng hạn: mạng chỉ biết giá trị đầu ra thực sự "quá cao" hay "chính xác 50%", hay mạng chỉ có được thông tin đầu ra đúng hay sai. Luật học dựa trên thông tin đánh giá này được gọi là luật học củng cố, và thông tin phản hồi được gọi là thông tin củng cố. Như *hình 4.4(b),* luật học củng cố là một dạng của luật học giám sát vì mạng vẫn có được thông tin phản hồi từ môi trường. Tuy nhiên, thông tin phản hồi này chỉ mang tính đánh giá chứ không phải mang tính chất học. Có nghĩa là, thông tin chỉ đưa ra đánh giá về đầu ra tốt hay xấu mà không đưa ra câu trả lời đúng là gì. Tín hiệu củng

cố này được mạng sử dụng để điều chỉnh trọng với hy vọng được đánh giá phản hồi tốt hơn trong lần học tiếp theo.

c) Học không giám sát (Unsupervised Learning)

Trong dạng học không giám sát này không có bất kì một thông tin phản hồi từ môi trường nào. Mạng phải tự tìm ra các mẫu, đặc tính, tính qui tắc, sự tương quan trong dữ liệu đầu vào và tập hợp lại để tạo đầu ra. Khi tự tìm ra các đặc điểm này, mạng đã trải qua các thay đổi về tham số của nó. Quá trình này được gọi là tự tổ chức.



Hình 4.4. Các dạng học *(a): Học giám sát; (b): Học củng cố; (c): Học không giám sát*

Các luật học có thể khái quát thành dạng chung với lượng điều chỉnh trọng như sau:

$$\Delta w_{ij} = \alpha r x_j(t) \qquad (4.4)$$

Trong đó *i:* nơ ron thứ *i; j*: là đầu vào thứ *j;* α : là hằng số học (dương) xác định tốc độ học và được xác định bằng thực nghiệm; *r* : tín hiệu học. Tín hiệu học tổng quát là một hàm của *w, x* và *d* tức là *r = f(w, x, d).*

Đối với các trọng biến đổi liên tục, có thể sử dụng dạng sau:

$$\frac{dw_i(t)}{dt} = \alpha r x(t) \qquad (4.5)$$

2. Học cấu trúc (Structure Larning)

Tư tưởng của học cấu trúc là thay đổi số nơ ron, kiểu liên kết để làm cấu trúc mạng thay đổi. Đối với học tham số, chúng ta giả sử cấu trúc mạng đã có, sau đó đưa ra các thuật học cho các tham số mạng (hay các trọng liên kết) để huấn luyện mạng thực hiện được nhiệm vụ như mong muốn. Còn việc học mức cấu trúc có thể sử dụng các kỹ thuật liên quan đến thuật toán GA (*Genetic Algorithm*) và lập trình tiến hoá (*EP: Evolutionary Programming*). Các cách tìm kiếm trong GA và EP là khá tiêu tốn thời gian ngay cả đối với mạng có kích thước trung bình. Do đó, còn có thể sử dụng các kỹ thuật sửa đổi hay xây dựng mạng dần dần từ một cấu trúc mạng ban đầu. Các kỹ thuật này bao gồm cắt xén bớt mạng nơ ron. Phát triển mạng và kết hợp cả hai cách cắt xén và phát triển mạng nơ ron.

Phần tiếp theo, chúng ta nghiên cứu một số mạng điển hình cho ba nhóm mạng nơ ron: mạng truyền thẳng; mạng có phản hồi (hay truy hồi); mạng tự tổ chức (*SOM*).

4.3 Các mạng truyền thẳng

Có hàng chục kiến trúc mạng truyền thẳng nhiều lớp (nối tiếp) như Multi Layer Perceptron (MLP) ; Adaline ; Lan truyền ngược; RBF… Trong phạm vi giáo trình chúng ta giới hạn nghiên cứu một số mạng điển hình như dưới đây.

4.3.1 Mạng 1 lớp truyền thẳng - Mạng Perceptron

a) *Cấu trúc*. Các nơ ron tạo thành lớp, trong đó mỗi tín hiệu vào có thể được đưa vào cho tất cả các nơ ron của lớp. Mỗi nơ ron có nhiều đầu vào và một đầu ra trên mỗi nơ ron đó. Cấu trúc của mạng Perceptron được chỉ ra hình 4.5.

Đầu vào của mạng có thể được mô tả dạng vector $X=[x_1, x_2,...,x_m]^T$, trong đó m là số lượng đầu vào, T là ký hiệu chuyển vị. Với n nơ ron, vector đầu ra thực tế là

$$Y=[y_1, y_2, \ ...y_n]^T$$

Mạng Perceptron sử dụng luật học có giám sát. Do đó, mẫu đầu vào là vector

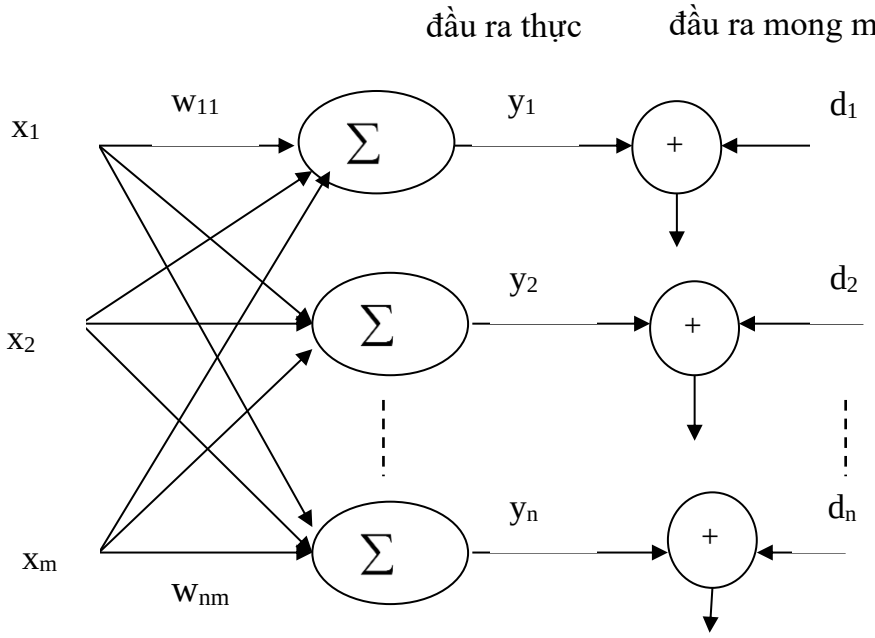
$$X^{(k)}=[\ x_1^{(k)}, x_2^{(k)},...,x_m^{(k)}]^T,$$

mẫu đầu ra mong muốn là vector

$$d^{(k)}=[d_1^{(k)}, d_2^{(k)},...,d_n^{(k)}]^T.\ k=1,2,...,p,$$

trong đó m là số đầu vào, n là số đầu ra, p là số cặp vào ra theo tập luyện tập. Chúng ta muốn đầu ra thực sự $y^{(k)}= d^{(k)}$ sau quá trình học và có thể được mô tả như sau:

$$y_i^{(k)}= f(W_i^T x_i^{(k)}) = f(\sum_{j=1}^{m} W_{ij}\, x_j^{(k)}) = d_i^k\ ,\ i = 1,2,...,n;\ k = 1,2,... \qquad (4.6)$$



Hình 4.5. Mạng Perceptron một lớp đơn

Trong mạng perceptron sử dụng hàm phi tuyến là hàm dấu nên phương trình (4.6) có thể viết thành:

$$y_i^{k)} = Sign(W_i^T x_i^k) = d_i^k \qquad ; k=1,2,3,\ldots,p \qquad (4.7)$$

b) Luật học perceptron

Luật học trong mạng nơ ron nhân tạo là qui tắc, trong đó các giá trị được cập nhật sau một số biến đổi. Có nhiều luật học khác nhau trong mạng nơ ron, nhưng tất cả đều có thể qui về dạng chung nhất, dưới dạng

$$\Delta W_{ij} \equiv \alpha\, r\, x_j(t). \qquad (4.8)$$

Phương trình *(4.8)* được áp dụng cho chuỗi biến đổi trọng rời rạc. Đối với các trọng biến đổi liên tục có thể biểu diễn như sau:

$$dW_i(t)/dt = \alpha\, r\, x(t) \qquad (4.9)$$

Đối với mạng Perceptron, chúng ta sử dụng luật học Perceptron. Trong luật học Perceptron, tín hiệu học $r = d_i - y_i$. Do đầu ra mong muốn chỉ dữ lại 2 giá trị 1 và -1 nên ta có:

$$\Delta W_{ij} = \alpha(d_i - y_i)x_j = \alpha\,(d_i - Sign(w_i^T x))x_j = \begin{cases} 2\alpha d_i x_i & if \quad y_i \neq d_i \\ 0 & else \end{cases} \qquad (4.10)$$

Như vậy, các trọng chỉ được cập nhật khi đầu ra thực sự y_i khác với d_i. Các trọng được khởi tạo với giá trị bất kì. Luật học Perceptron sẽ hội tụ sau một số bước hữu hạn.

4.3.2 Mạng nơ ron Adaline (Adaptive Linear Element)

Đây là mạng với phần tử tuyến tính đơn do *Widrow* nêu ra vào năm 1962 có quan hệ vào - ra tuyến tính. Giống như mạng *Perceptron*, cho tập mẫu $p\ \{(x^{(1)}, d^{(1)}), (x^{(2)}, d^{(2)}),.., (x^{(p)}, d^{(p)})\}$ cần tìm các trọng W_i sao cho

$$\sum\ w_j x_j^{(k)} = d^{(k)}; \qquad k = 1, 2 ,...,p \tag{4.11}$$

Để tìm các trọng từ phương trình *(4.11)* cần xác định hàm giá (hàm chi phí) *E(w)*

$$E(w)= \frac{1}{2}\Sigma (d^{(k)} - y^{(k)})^2=\frac{1}{2}\sum_{K=1}^{P}\ (d^{(k)} - w^T x^{(k)})^2= \frac{1}{2}\sum\ (d^{(k)} - \sum\ w_j x_j^{(k)})^2$$

$$k=1, 2, ..., p\ ;\ j =1, 2 ,...,m \tag{4.12}$$

Rõ ràng là, nếu *E(w)* càng nhỏ thì w_j càng tốt hơn. *E(w)* là dương, nhưng sẽ tiệm cận về *0* khi $y^{(k)}$ tiến đến $d^{(k)}$. Do đó, cố gắng tìm các trọng mà làm tối thiểu trung bình bình phương *E(w).*

Widrow và *Hoff* đã sử phương pháp hạ gradient để điều chỉnh mỗi w_j bằng một lượng Δw_j tỷ lệ với âm gradient của *E(w)*

$$\Delta w_j = -\eta\frac{\partial \mathrm{E}}{\partial \mathrm{w_j}} = \eta\sum_{k=1}^{p}(d^{(k)} - W^T x^{(k)})x_j^k \qquad j=1,\ 2\ ,...,\ m \tag{4.13}$$

Nếu những thay đổi này được tạo riêng biệt từng mẫu đầu vào lần lượt $x^{(k)}$, khi đó thay đổi tương ứng với mẫu $x^{(k)}$ sẽ là

Nếu những thay đổi được tiến hành riêng rẽ cho mỗi mẫu vào $X^{(k)}$ thì

$$\Delta w_j = \alpha\,(d^{(k)} = w^T x^{(k)})x_j^{(k)} \qquad k=1,2,..,p$$

$$\Delta w_j = \alpha\,(d^{(k)} - w^T x^{(k)})x_j^{(k)} \qquad k=1,2,..,p \tag{4.14}$$

Luật học này là luật Adaline hay LMS (bình phương trung bình nhỏ nhất), *r* ở đây là:

$$r = d - y = d - w^T x$$

Áp dụng luật học trên cho mạng một lớp đơn, chứa nhiều phần tử Adaline với hàm kích hoạt là *f(.)*

$$E(w) = \frac{1}{2}\sum_{k=1}^{p}\sum_{i=1}^{n}(d_i^k - y_i^k)^2 = \frac{1}{2}\sum_{k=1}^{p}\sum_{i=1}^{n}[d_i^{(k)} - f(w_i^T x^{(k)})]^2 = \frac{1}{2}\sum_{k=1}^{p}\sum_{i=1}^{n}\left[d_i^{(k)} - f(\sum_{j=1}^{m} w_{ij} x_j^{(k)})\right]^2 \tag{4.15}$$

$$\Delta w_{ij} = -\eta\frac{\partial E}{\partial w_{ij}} = \eta\sum_{k=1}^{p}\left[d_i^k - f\left(net_i^{(k)}\right)\right]f'\left(net_i^{(k)}\right)x_i^{(k)} \tag{4.16}$$

Trong đó:

$$net_i^{(k)} = w_i^T x^{(k)}, \qquad k=1,2,...p;\ \ i=1,2,..,n;\ j=1,2,...,m$$

Như vậy, trọng được thực hiện điều chỉnh thích ứng với từng mẫu vào

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} = \eta\,[d_i^{(k)} - f(net_i^{(k)})]\; f'(net_i^{(k)}) \cdot x_j^{(k)} \qquad (4.17)$$

Luật học này là luật học delta. Theo luật học tổng quát, tín hiệu học ở đây là:

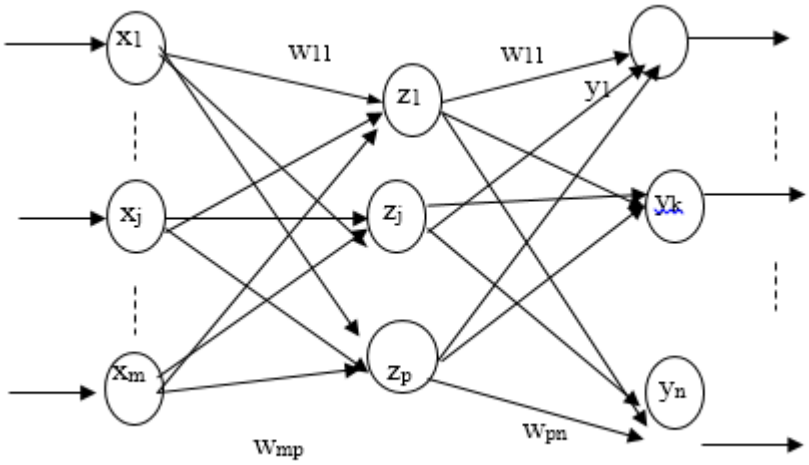
$$r = [d_i - f(w_i^T x)] f'(w_i^T x) \qquad (4.18)$$

Widrow và các tác giả cũng nghiên cứu và phát triển mạng Adiline nhiều lớp, gọi là MultiAdaline [2,8].

4.3.3 Mạng nhiều lớp lan truyền ngược (Back Propagation)

Thuật học lan truyền ngược là một trong những phát triển quan trọng trong mạng nơ ron. Thuật toán này được áp dụng cho các mạng nhiều lớp truyền thẳng (Feedforward) gồm các phần tử xử lý với hàm kích hoạt liên tục. Các mạng như vậy kết hợp với thuật toán học lan truyền ngược được gọi là mạng lan truyền ngược.

a) Cấu trúc



Hình 4.6. Mạng nhiều lớp lan truyền ngược

Luật học truyền ngược có ý nghĩa rất quan trọng trong việc cập nhật trọng của mạng nhiều lớp truyền thẳng. Nền tảng của thuật toán cập nhật trọng này cũng là phương pháp hạ Gradient. Thật vậy, cho cặp mẫu đầu vào - đầu ra $(x^{(k)}, d^{(k)})$, thuật toán lan truyền ngược thực hiện 2 pha. Đầu tiên, mẫu đầu vào $x^{(k)}$ được truyền từ lớp vào tới lớp ra và kết quả của luồng dữ liệu thẳng (forward) này là tạo đầu ra thực sự $y^{(k)}$. Sau đó, tín hiệu lỗi tạo từ sai khác giữa $d^{(k)}$ và $y^{(k)}$ sẽ được lan truyền ngược từ lớp ra quay trở lại các lớp trước đó để chúng cập nhật trọng. Để minh

hoạ chi tiết thuật toán lan truyền ngược, xét một mạng 3 lớp: lớp vào có m nơ ron, lớp ẩn có l nơ ron và lớp ra có *n* nơ ron (hình 4.6)

+ *Lớp ẩn*: với tập mẫu đầu vào *x*, nơ ron thứ *q* của lớp ẩn nhận tổng đầu là

$$net_q = \sum_{j=1}^{m} v_{jq} x_j \qquad j=1, 2, \dots, m ; \; q=1,2,\dots,l \qquad (4.19)$$

Và tạo đầu ra của lớp ẩn

$$z_q = f(net_q) = f(\sum_{j=1}^{m} v_{qj} x_j) \qquad (4.20)$$

trong đó, *f(.)* là hàm tương tác đầu ra.

+ *Lớp ra:* giả thiết hàm tương tác đầu ra của lớp ra giống các lớp khác, tức là *f(.)*. Khi đó, tổng đầu vào của nơ ron thứ *i* có thể xác định

$$net_i = \sum_{q=1}^{l} w_{iq} z_q = \sum_{q=1}^{l} w_{iq} f\left(\sum_{j=1}^{m} v_{qj} x_j \right) \qquad (4.21)$$

Và tạo đầu ra:

$$y_i = f(net_i) = f\left(\sum_{q=1}^{l} w_{iq} z_q \right) = f\left(\sum_{q=1}^{l} w_{iq} f\left(\sum_{j=1}^{m} v_{qj} x_j \right) \right) \qquad (4.22)$$

b) *Luật lan truyền ngược* **(**Backpropagation Learning Rule**)**

Cơ sở của luật học lan truyền ngược được xây dựng trên phương pháp hạ Gradient. Đầu tiên, xây dựng hàm chi phí (hay còn gọi là hàm sai số giữa đầu ra mong muốn d_i với đầu ra thực tế y_i)

$$E(w) = \frac{1}{2} \sum_{i=1}^{n} (d_i - y_i)^2 = \frac{1}{2} \sum_{i=1}^{n} \left[d_i - f(net_i) \right]^2 = \frac{1}{2} \sum_{i=1}^{n} \left[d_i - f\left(\sum_{q=1}^{l} w_{iq} z_q \right) \right]^2 \qquad (4.23)$$

Theo phương pháp hạ Gradient, trọng liên kết giữa lớp ẩn và lớp đầu ra được cập nhật bởi:

$$\Delta w_{iq} = -\eta \frac{\partial E}{\partial w_{iq}} = -\eta \left(\frac{\partial E}{\partial y_i} \right) \left(\frac{\partial y_i}{\partial net_i} \right) \left(\frac{\partial net_i}{\partial w_{iq}} \right) = \eta (d_i - y_i) (f'(net_i)) z_q = \eta \delta_{oi} z_q \qquad (4.24)$$

Với δ_{oi} là tín hiệu lỗi tại nơ ron thứ *i* trong lớp ra

$$\delta_{oi} = (d_i - y_i) f'(net_i) \qquad (4.25)$$

Đối với trọng liên kết giữa nơ ron thứ *j* của lớp vào và nơ ron thứ *q* của lớp ẩn sẽ được cập nhật theo:

$$\Delta v_{qj} = -\eta \frac{\partial E}{\partial v_{qi}} = -\eta \left(\frac{\partial E}{\partial net_q} \right) \left(\frac{\partial net_q}{\partial v_{qj}} \right) = -\eta \left(\frac{\partial E}{\partial z_q} \right) \left(\frac{\partial z_q}{\partial net_q} \right) \left(\frac{\partial net_q}{\partial v_{qj}} \right)$$

$$= \eta \sum_{i=1}^{n} \left[(d_i - y_i) f'(net_i) w_{iq} \right] f'(net_q) x_j \qquad (4.26)$$

Từ phương trình *(1-20)* ta có:

$$\Delta v_{qj} = \eta \sum_{i=1}^{n} \left[\delta_{oi} w \right] f'(net_q) x_j = \eta \delta_{hq} x_j \qquad (4.27)$$

Với δ_{hq} là tín hiệu lỗi của nơ ron thứ q trong lớp ẩn

$$\delta_{hq} = f'(net_q) \sum_{i=1}^{n} \delta_{0i} w_{iq} \qquad (4.28)$$

Rõ ràng, từ hàm sai lệch đầu ra *(4.23*), theo phương pháp hạ Gradient, chúng ta có thể tính ngược trọng từ lớp ra, tiếp theo đến trọng của lớp trước đó. Điều này thể suy luận và tính các lớp trọng cho một mạng nơ ron truyền thẳng có số lớp bất kỳ.

Ưu điểm: Mạng lan truyền ngược có ưu điểm tính toán trong chính xác.

Nhược điểm: Nhược điểm lớn nhất của mạng lan truyền ngược là dùng phương pháp học hạ gradieent chỉ đảm bảo tối ưu cục bộ. Để khắc phục, người ta đề xuất hàng chục thuật toán cải tiến để vượt khe cục bộ và các phương pháp lai, nhằm tìm bộ trọng số xuất phát ban đầu để hội tụ về điểm cực trị toàn cục.

4.4 Các mạng phản hồi

Các mạng phản hồi (truy hồi) điển hình gồm: Mạng *Hopfield* rời rạc (1982); Mạng *Hopfield* liên tục (1984); Mạng liên kết hai chiều BAM (thực chất là hai mạng Hopfield đấu phản hồi); mạng *Cohen-Grossberg* (thực chất là khái quát hóa mạng *Hopfield* liên tục thành định lý *Cohen-Grossberg*, nhưng rất khó thực hiện trong kỹ thuật); mạng nơ ron tế bào do *Chu* đề xuất và đã chế tạo thành máy tính đa năng hai chiều (thực chất là mạng nơ ron hai chiều của mạng *Hopfield*)… Điển hình của nhóm mạng này trong ứng dụng: dùng làm bộ nhớ địa chỉ hóa nội dung; dùng làm các bộ tối ưu; đặc biệt thành công là thực hiện để sản xuất các phần cứng máy tính kiểu tương tự. Dưới đây giới thiệu một số mạng nơ ron phản hồi và luật học, tính ổn định của các mạng điển hình nhất.

4.4.1 Mạng Hopfield rời rạc

Xét mạng *Hopfield* rời rạc (năm 1982). Phương trình mô tả luật tác động:

$$x_i(t) = \sum_{j=1}^{n} W_{ij}\, y_j(t) - I_i \quad i,j = 1,...,n; \tag{4.29}$$

Luật cập nhật đầu ra:

$$y_i(t+1) = \begin{cases} g(x_i(t)), & \text{nếu } x_i(t) \neq 0,\ i = p; \\ y_i(t), & \text{nếu } x_i(t) = 0,\ i \neq p \end{cases} \tag{4.30}$$

Hàm quan hệ vào ra là hàm phi tuyến bước nhảy

$$g(x_i(t)) = \begin{cases} 1, & \text{nếu } x_i(t) > 0; \\ 0, & \text{nếu } x_i(t) < 0 \end{cases} \tag{4.31}$$

Luật cập nhật trọng liên kết theo luật Hebb tương quan:

$$W_{ij} = \begin{cases} \sum_{p=1}^{h} (2y_{p,i} - 1)(2y_{p,j} - 1), & \text{nếu } i \neq j \\ 0, & i = j \end{cases} \tag{4.32}$$

$$I_i = \frac{1}{2} \sum_{j=1}^{n} W_{ij} \tag{4.33}$$

Trong đó, $x_i(t)$: tổng của tất cả các đầu vào; $y_i(t)$: đầu ra của nơ ron; W_{ij} : là trọng liên kết phản hồi từ nơ ron i tới nơ ron j ; I_i: hằng số của nơ ron i; h là số mẫu được cất giữ; n là số nơ ron; p là phần tử thứ p đang tác động.

Hopfield cũng nêu hàm năng lượng mạng (hay hàm thế năng):

$$E(y) = -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} y_i y_j + \sum_{i=1}^{n} \theta_i y_i \tag{4.34}$$

Nếu $W_{ij} = 0$ và $W_{ij} = W_{ji}$ thì mỗi thay đổi không đồng bộ của y_p năng lượng sẽ giảm phù hợp theo:

$$\Delta E = -[y_p(t+1) - y_p(t)]\, [\sum_{j=1}^{n} a_{pj} y_j - w_p] \tag{4.35}$$

4.4.2 Mô hình mạng Hopfield liên tục chuẩn

Hopfield (1984) đa ra mô hình mạng mô tả bằng tập các phương trình vi phân

$$C_i \dot{x}_i = -\frac{x_i}{R_i} + I_i + \sum_{j=1}^{n} W_{ij} y_j \qquad (4.36)$$

$$y_j = g_j(x_j) \qquad (4.37)$$

$$x_i = g_i^{-1}(y_i) \qquad (4.38)$$

Trong đó, C_i và R_i là các hằng số; I_i là ngưỡng; W_{ij} là trọng liên kết giữa phần tử nơ ron thứ j với nơ ron thứ i; x_i là trạng thái nơ ron thứ i.

Hopfield nêu hàm Liapunov với dạng sau:

$$V(x) = \sum_{i=1}^{n} (1/R_i) \int_0^{y_i} g_i^{-1}(\zeta) d\zeta - \sum_{i=1}^{n} I_i y_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} y_i y_j \,. \qquad (4.39)$$

a) Ứng dụng mạng Hopfield cho các bài toán tối ưu

Để giải quyết các vấn đề tối ưu óa thì trong mạng hopfield, các hàm năng lượng được sử dụng tương đương như hàm mục tiêu để mà tối thiểu hoá.

Việc tìm hàm tối thiểu trong mạng *Hopfield* chính là tìm lời giải cho các vấn đề tối ưu. Kết quả là phải đưa ra một vấn đề tối ưu với một hàm mục tiêu chính xác mà nó có thể được dùng để cấu thành một mạng hopfield, đặc biệt là tìm các trọng (weight) của chúng.

Khi ta sử dụng mạng noron để giải quyết các vấn đề tối ưu, thì phải xây dựng chính xác từng loại thuật toán song song phù hợp với lời giải đó.

Ví dụ 1. Trong ví dụ này chúng ta sẽ thiết kế bộ chuyển đổi A/D 4 bít mà sử dụng mạng *Hopfield* đơn liên tục.

Mục đích là chuyển đổi từ một giá trị đầu vào liên tục là x $(0 < x < 15)$ và đầu ra là $y = [\, y_3 , y_2 , y_1 , y_o \,]^T$ với $y_i \in \{0,1\}$; để giá trị thập phân của

$8y_3 + 4y_2 + 2y_1 + y_o$ và giá trị của x được gần nhau nếu có thể.

Sai số của bộ chuyển đổi A/D

$$E_c = \frac{1}{2}\left(x - \sum_{i=0}^{3} 2^i y_i \right)^2$$

Rừ ràng là tối thiểu hoỏ hàm năng lượng tương đương với việc tối thiểu hoá sai số chuyển đổi của bộ chuyển đổi A/D.

Mục đích là phải xây dựng mạng *Hopfield* liên tục có 4 nút với hàm một hàm kích hoạt để tối thiểu hoá.

Để phục vụ cho mục đích này, chúng ta phải tìm ra các thông số chính xác, gồm các trọng và đầu vào mở rộng của mạng *Hopfied*. Việc này có thể được thực hiện được bằng cách so sánh giữa E_c và E_q (hàm năng lượng của mạng *Hopfield* liên tục).

Tuy vậy, trong biểu thức E_c có y_i^2 *(i = 0, 1, 2, 3)* với hệ số khác 0 thì cũng làm cho w_{ii} trong mạng *Hopfield* khác không. Sự mâu thuẫn này được định nghĩa trong mạng *Hopfield*. Vì vậy E_a được thêm vào như sau:

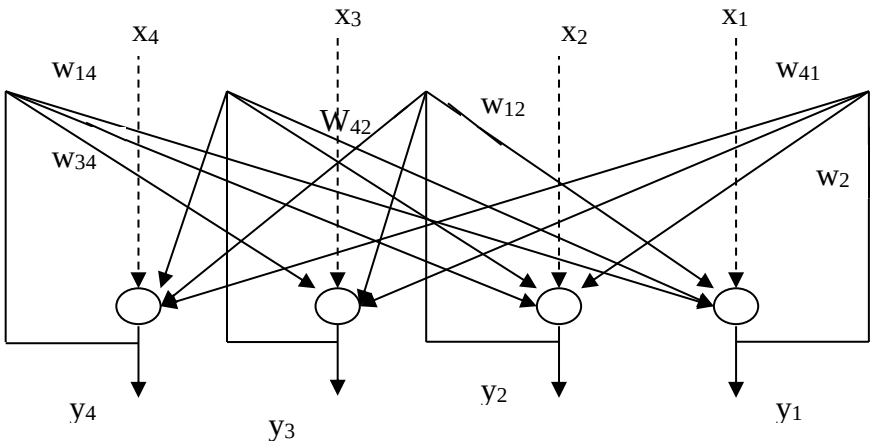
$$E_a = -\frac{1}{2}\sum_{i=0}^{3} 2^{2i} y_i(y_i - 1)$$

Hàm tổng năng lượng là:

$$E = E_c + E_a = \frac{1}{2}\left(x - \sum_{i=0}^{3} 2^i y_i\right)^2 - \frac{1}{2}\sum_{i=0}^{3} 2^{2i} y_i(y_i - 1) \qquad (4.40)$$

Chú ý rằng E_a không âm và đạt giá trị thấp nhất khi $y_i=0$ hoặc $y_i=1$. Do đó E_a có thể cho trạng thái mạng phải vào các góc của hình sườn khối lập phương (*Hypercube*) khi E đạt cực tiểu. Ta có hàm năng lượng E của mạng *Hopfield* liên tục, cứ 1 lớp 4 noron. Với các đầu vào ngoài $x = [x_3, x_2, x_1, x_o]^T$ và đầu ra $y=[y_3, y_2, y_1, y_o]^T$

$$E = \frac{1}{2}\sum_{i=0}^{3}\sum_{\substack{i=0 \\ j\neq i}}^{3} w_{ij} y_i y_j - \sum_{i=0}^{3} x_i y_i + \frac{1}{\lambda}\sum_{1}^{n} G_i \int_0^{y_i} a^{-1}(y)dy \qquad (4.41)$$



Hình 3.7. Mô hình mạng nơ ron Hopfield làm bộ A/D 4 bit

So sánh (4.40) và (4.41) ta có:

$w_{ij} = -2^{i+j}$ và $x_i = -2^{2i-1} + 2^i x$ với $i, j = 0,1,2,3;\ i \neq j$

Do đó:

$$W = -\begin{bmatrix} 0 & 2 & 4 & 8 \\ 2 & 0 & 8 & 16 \\ 4 & 8 & 0 & 32 \\ 8 & 16 & 32 & 0 \end{bmatrix} \quad \text{và} \quad x = -\begin{bmatrix} 0,5 & -x \\ 2 & -2x \\ 8 & -4x \\ 32 & -8x \end{bmatrix}$$

Với ma trận trọng như vậy, ta có sơ đồ mạng *Hopfield* như sau:

Có hai kiểu bộ nhớ liên kết là bộ nhớ liên kết tự động và bộ nhớ liên kết không đồng nhất (*Hereoassociative Memory*)

Xem bộ nhớ liên kết mạng *Hopfield* với *m* đầu vào và *n* đầu ra nhận các giỏ trị 1 hoặc -1

$$x \in \{+1,-1\}^m \, , \, y \in \{+1,-1\}^n, \, y=I(x)$$

Mạng lưu trữ gồm tập *p* mẫu $\{(x^1, y^1), (x^2,y^2),\ldots,(x^p,y^p)\}$ thông qua các trọng số W_{ij} nhờ thuật toán lưu trữ $W=F(x^r, y^r)$ nếu ta đưa vào mạng mẫu *x* thì khi mạng ổn định, sẽ cho kết quả $y=y^r$ sao cho x^r tương ứng giống *x* nhất trong *p* mẫu lưu trữ.

Kiểu bộ nhớ tự liên kết: $y^r=I(x)=x^r$

Kiểu bộ nhớ không đồng nhất: $y^r \neq x^r$

Khái niệm gần nhất "close" có thể xem xét như là một số phép xác định khoảng cách. Xét khoảng cách của *Oclit* và khoảng cách *Hamming*:

Khoảng cách *Oclit d* của 2 vector $x=(x_1,x_2,\ldots,x_n)^T$ và $x'=(x'_1,x'_1,\ldots,x'_1)^T$ được định nghĩa là $d=[(x_1-x'_1)^2+(x_2-x'_2)^2+\ldots+(x_n-x_n')^2]^{1/2}$

Khoảng cách *Hamming HD(x, x')* xác định số lượng các cặp không bằng nhau giữa 2 vector *x* và *x'*

Ví dụ: *Nếu* $x=(1,1,0,1)^T$ *và* $x'=(0,1,0,0)^T$, khi đó $HD(x,x')=2$

b) Ứng dụng mạng Hopfield làm bộ nhớ tự liên kết hồi quy (Bộ nhớ Hopfield)

Đây là mạng *Hopfield* rời rạc với các ngưỡng và các đầu ngoài vào bằng 0 (chỉ cần thành phần hồi quy (hay đơn giản là phản hồi))

Thuật toán lưu trữ:

$$W=\sum_{k=1}^{p} x^k(x^k)^T-I \qquad (4.42)$$

Hay

$$w_{ij}=\sum_{k=1}^{p} x^k{}_i x^k{}_j \qquad i \neq j;\ w_{ii}=0; \qquad (4.43)$$

Trong đó $x^k=(x^k{}_1,x^k{}_2,\ldots,x^k{}_n)^T$ và *I* là ma trận xác định xấp xỉ

Nếu x_i là ma trận nhị phân đơn cực, tức là $x_i \in \{0,1\}$:

Thuật toán lưu trữ:

$$W=\sum_{k=1}^{p} x^k(x^k)^T-I \qquad (4.44)$$

Hay $w_{ij}=\sum_{k=1}^{p} x^k{}_i x^k{}_j \qquad i\neq j;\ w_{ii}=0;$

Nếu $x_i \in \{0,1\}$ thì:

$$w_{ij}=\sum_{k=1}^{p} (2x^k{}_i-1)(2x^k{}_j-1) \qquad i\neq j;\ w_{ii}=0; \qquad (4.45)$$

Công thức xác định *(4.44)* dựa trên luật học Hebbian với trọng số ban đầu là *0*. Vì vậy luật học được gọi là luật học kiểu *Hebbian* hay luật học tích ngoài.

Ta có thể cộng thêm vào bộ nhớ bằng cách tăng ma trận trọng số, cũng như có thể giảm đi. Việc này không bị ảnh hưởng bởi thứ tự lưu trữ các mẫu.

Ví dụ: Xem xét sử dụng bộ nhớ *Hopfield* để lưu trữ 2 vector x^1 và x^2

$x^1 = [1,-1,-1,1]^T$ và $x^2 = [-1,1,-1,1]^T$

Theo luật học trên ta có ma trận trọng số:

$$W=\begin{bmatrix} 0 & -2 & 0 & 0 \\ -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 \\ 0 & 0 & -2 & 0 \end{bmatrix}$$

Ví dụ trên chỉ ra 1 tính chất quan trọng của bộ nhớ *Hopfield* đó là sự đầy đủ của một vector lưu trữ cũng là một vector lưu trữ. Bởi vì chúng có cùng một giá trị năng lượng $E(x) = E(\bar{x})$. Yếu tố chủ chốt là sự hội tụ tương tự giữa vector đầu ra khởi tạo và x

Có 2 vấn đề lớn của bộ nhớ *Hopfield*. Thứ nhất, là trạng thái ổn định không định trước gọi là *trạng thái ổn định giả tạo* nguyên nhân bởi sự tối thiểu các hàm năng lượng thêm vào cái chúng ta cần. Tràn bộ nhớ là kết quả của việc khoảng cách *Hamming* giữa các mẫu lưu trữ nhỏ vì không cung cấp các lỗi và sự phục hồi hiệu quả cho các mẫu lưu trữ. Có thể thấy, có mối quan hệ giữa số lượng các trạng thái giả tạo giảm và kích cỡ (chiều) của vector lưu trữ tăng trên khía cạnh số lượng các vector lưu trữ.

Khả năng lưu trữ của bộ nhớ Hopfield.

Khả năng lưu trữ này có mối quan hệ với kích thước của mạng. Một phép đo hữu hiệu cho việc ước lượng dung lượng của bộ nhớ là bán kính hấp dẫn ρ. Bán kính của một bộ nhớ tự liên

kết là: là khoảng cách mà mọi vector nằm trong vùng bán kính đó vẫn đạt được trạng thái ổn định. Khoảng cách ở đây thường dùng là khoảng cách *Hamming* bởi vì giá trị của nó là các số nguyên; nghĩa là, khi đưa vào mẫu *x* có $HD(x, x^r) <= \rho n$ thì ở đầu ra nhận được mẫu lưu trữ x^r

Dung lượng c là số mẫu x^r tối đa có thể được lưu trữ *(p <= c)* để đảm bảo các mẫu trong bán kính ρ vẫn có thể nhận được đúng. Có thể ước lượng cho bộ nhớ tự liên kết gồm có *n* neuron (theo *McEliece*, 1987):

$$C = \frac{(1-2\rho)^2}{4 \ln n} \qquad 0 \le \rho < 1/2 \qquad (4.46)$$

4.4.3 Mạng liên kết hai chiều

1. Giới thiệu

Một bộ nhớ liên kết hai chiều (*BAM**:** Bidirectional Associative Memory*) có thể lưu trữ một tập các mẫu như các bộ nhớ. Khi ta đưa vào bộ nhớ liên kết một mẫu, nó sẽ trả lại kết quả gần với giá trị tương ứng của mẫu với mẫu đưa vào. Vì vậy, việc xác định thông qua các mẫu với các thông tin cần nhớ. Các mẫu được nhớ vào và gọi ra theo nội dung của nó, nên nó được gọi là bộ nhớ nội dung địa chỉ hoá, khác với bộ nhớ truyền thống của máy tính số là bộ nhớ địa chỉ-địa chỉ hoá. Bộ nhớ liên kết chính là một dạng của mạng *Hopfield*.

2. Bộ nhớ liên kết 2 chiều sử dụng BAM

BAM là loại bộ nhớ hồi quy liên kết không đồng nhất gồm có 2 lớp và có thể coi như là một sự mở rộng của mạng *Hopfield*. Nó thực thi việc tìm kiếm xuôi và ngược trong các bộ lưu trữ các tác nhân đáp ứng liên kết.

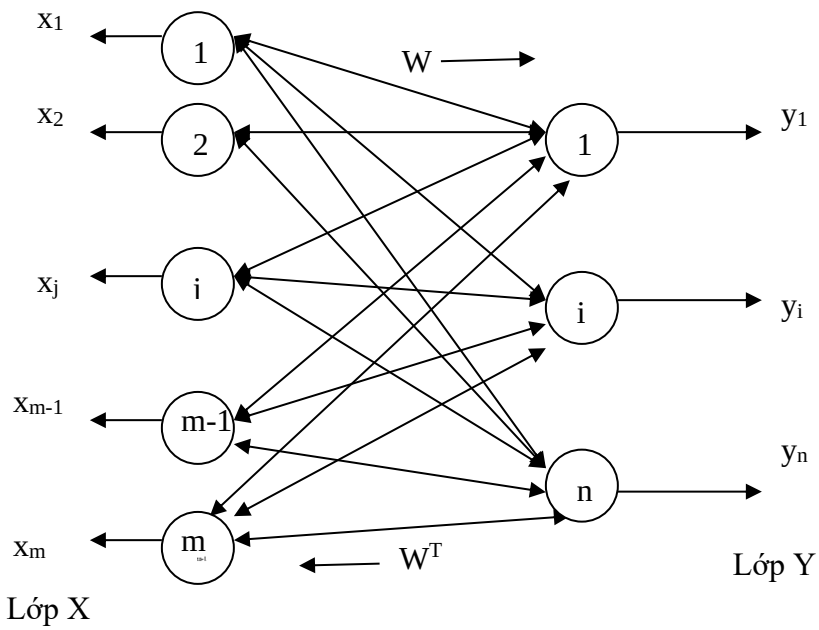
Sau đây là cấu trúc của bộ nhớ *BAM* rời rạc. Khi các noron nhớ được kích hoạt bởi vector khởi tạo X tạo đầu vào. Khi đó mạng tiến tới có 2 phần trạng thái ổn định mà đầu này sẽ là đầu ra của đầu kia. Chức năng của mạng gồm có 2 tầng tương tác. Giả sử có một vector khởi tạo *X* cung cấp cho đầu vào của lớp noron *Y*. Đầu vào sẽ được xử lý và chuyển đổi thành đầu ra của *Y* theo sau:

$$y' = f(Wx) \quad \text{hoặc} \quad y_i' = f\left(\sum_{j=1}^{m} w_{ij}x_j\right) \; i=1..n;$$

Đầu ra của lớp *Y*: $y' = f(Wx)$ hoặc $y'_I = f\left(\sum_{j=1}^{m} w_{ij}x_j\right) + I \qquad =1..n$

Đầu ra lớp X:

$x^{'}=a(W^{T}y^{'})$ hoặc $x'_j= a(\sum_{j=1}^{n} w_{ji}y_i)$ $j=1,2..m$



Hình 4.8. Mô hình mạng BAM

f (.) là hàm ngưỡng. Vector *y'* cung cấp cho lớp *X* và vector *x'* cung cấp đầu vào cho lớp *Y* cho đầu ra *y''*.Quá trình sẽ tiếp tục cho tới khi cập nhật *x* và *y* dừng lại. Quá trình truy hồi đệ quy có thể gồm các bước sau

$y^{(1)}=a(Wx^{(0)})$ *(qua hướng đi lần thứ nhất)*

$x^{(2)}=a(W^{T}y^{(1)})$ *(qua hướng về lần thứ nhất)*

……

$y^{(k-1)}=a(Wx^{(k-2)})$ *(qua hướng đi lần thứ k/2)*

$x^{(k)}=a(W^{T}y^{(k-1)})$ *(qua hướng về lần thứ k/2)*

Trạng thái cập nhật có thể là đồng bộ hoặc không đồng bộ.

Thuật toán lưu trữ

Với *p* cặp vector liên kết lưu trữ trong *BAM*:

$\{(x^1,y^1), (x^2, y^2), ..., (x^n, y^n)\}$

trong đó $x^k=(x_1^k,x_2^k,...,x_m^k)^T$ và $y=(y_1^k, y_2^k,..., y_n^k)^T$

$W= \sum_{k=1}^{p} (y^k x^k)^T$ cho các vector lưỡng cực : {1, -1}

$$W=\sum_{k=1}^{p}(2y^k-1)(2x^k-1)^T$$ cho các vector nhị phân : {0, 1}

hoặc $w_{ij}=\sum_{k=1}^{p} y^k{}_i x^k{}_j$ Cho các vector lưỡng cực :{1, -1}

$$w_{ij}=\sum_{k=1}^{p}(2y^k{}_i-1)(2x^k{}_j-1)$$ cho các vector nhị phân :{0, 1}

Tính ổn định của *BAM* (được chứng minh dùng định lý của *Lyapunov*)

Hàm năng lượng:

$$E(x,y)=-1/2x^TW^Ty-1/2y^TWx=-y^TWx \qquad (4.47)$$

Xem xét ΔE sinh ra do Δy theo 11.36 ta có

$$\Delta E_{yi}=\frac{\partial E}{\partial yi}\Delta y_i=Wx\Delta y_i=-(\sum_{j=1}^{m} w_{ij}x_j)\Delta y_i$$

Có 3 trường hợp xảy ra:

$$y_i^{(k)}=-1 \text{ và } y_i^{(k+1)}=+1\rightarrow\sum_{j=1}^{m} w_{ij}x_j>0,\ \Delta y_i=2 \text{ Do đó } \Delta E_{yi}<0$$

$$y_i^{(k)}=+1 \text{ và } y_i^{(k+1)}=-1\rightarrow\sum_{j=1}^{m} w_{ij}x_j<0,\ \Delta y_i=-2 \text{ Do đó } \Delta E_{yi}<0$$

$y_i^{(k)}=y_i^{(k+1)}\rightarrow \Delta y_i=0$. Do đó $\Delta E_{yi}=0$

Đối với Δ*x* làm tương tự

Dung lượng bộ nhớ

Ước lượng $p \le min(m,n)$ *hoặc* có thể xác định $p=\sqrt{\min(m,n)}$

Ví dụ dùng mạng BAM để nhớ, nhận dạng - gán nhãn

Ví dụ này minh họa khả năng của mạng *BAM* dùng làm bộ nhớ liên kết địa chỉ hóa nội dung với khich thước *m*n* ; nhận mẫu đầu vào, gán nhãn đầu ra; khả năng chị lỗi.

Bộ nhớ liên kết 2 chiều thường được dùng để minh hoạ việc cô lập lỗi và điều khiển. *BAM* là mạng phản hồi 2 lớp của các thành phần tương tác như là các bộ lưu trữ nhớ liên kết (*Recall Stored Sssociations*) với (X_i, Y_i) $i=1...q$. Như vậy, một vector *x* có *n* chiều đầu vào sẽ cho kết quả ra là vector *y* có *m* đầu ra. Mạng được xây dựng từ ma trận trọng cố định *W* kích cỡ *m*n*.

Quá trình xử lý phần tử tại bước thứ *k* tại lớp ra *y* được cập nhật như sau:

$y_i(k+1)=1$ *nếu* $y_i^*(k+1)>0$

$=y(k)$ *nếu* $y_i^*(k+1)=0$

$= 0$ *nếu* $y_i^*(k+1)<0$

Cấu trúc lớp ra được mô tả $y_i^*(k+1)= \sum_{j=1}^{n} w_{ij}x_j$

Quá trình xử lý phần tử tại bước thứ *k* tại lớp *X* được cập nhật như sau:

$x_i(k+1) = 1$ *nếu* $x_i^*(k+1)>0$

$= x(k)$ *nếu* $x_i^*(k+1)=0$

$= 0$ *nếu* $x_i^*(k+1)<0$

Cấu trúc lớp vào được mô tả $x_i^*(k+1)= \sum_{j=1}^{m} y_j w_{ji}$

Việc cập có thể được thực hiện đồng bộ tức là tất cả các phần tử xử lý được cập nhật trong một chu trình đồng hồ, hoặc được thực hiện không đồng bộ khi chỉ có một tập con được cập nhật tại mỗi thời điểm.

Hàm năng lượng được xác định như công thức ở trên

Luật *Hebb* ($W=\sum_{j=1}^{p} XY$ *với l:* số mẫu) có thể được dùng để mờ hóa *q* liên kết (X_i,Y_i) trong *BAM* trong việc thể hiện vector dạng nhị phân (*Binary Representation*) thành dạng lưỡng cực (*Bipolar Representation*)

Như thay 0 thành -1.

Cho (A_i, B_i) là dạng lưỡng cực thì kết quả ma trận trọng số là:

$$W=B_1^TA_1+B_2^TA_2+\ldots+B_q^TA_q$$

Vấn đề cho các bộ cảm ứng và bộ xác định cô lập lỗi (actuator failure isolation) trong việc quan tâm đến việc cải thiện bộ tin cậy của hệ thống điều khiển.

Sau đây là một *BAM* dùng như bộ ánh xạ liên kết từ không gian đặc trưng hệ thống sang không gian lỗi nhãn hệ thống.

Xác định định được 3 lớp lỗi thông qua các vector chức năng và tương ứng với các vector nhón

Kết quả của sản phẩm đầu ra đưa đến ma trận trọng số

$$W=L'^TA'+M'^TB'+N'^TC'$$

Giả sử có ba mẫu (k=3) cần nhớ được mã hoá từ trước ở dạng các giá trị đặc trưng là 1 hoặc 0 với năm phần tử nơ ron ra; 6 nơ ron đầu vào.

Vector đặc trưng	Vector nhãn
A=(1 0 1 0 11) A'=(1-1 1-1 1 1)	L=(1 1 1 1 1) L'=(1 1 1 11)
B=(1 0 1 0 0 0) B'=(1-1 1-1-1-1)	M=(0 1 1 0 0) M'=(-1 1 1 -1-1)
C=(0 1 0 1 1 1) C'=(-1 1-1 1 1 1)	N=(1 0 0 1 1) N'=(1 -1 -1 11)

Kết quả của sản phẩm đầu ra đưa đến ma trận trọng số

$$W=L'^{T}A'+M'^{T}B'+N'^{T}C' = \begin{bmatrix} -1 & 1 & -1 & 1 & 3 & 3 \\ 3 & -3 & 3 & -3 & -1 & -1 \\ 3 & -3 & 3 & -3 & -1 & -1 \\ -1 & 1 & -1 & 1 & 3 & 3 \\ -1 & 1 & -1 & 1 & 3 & 3 \end{bmatrix}$$

$$W^{T} = \begin{bmatrix} -1 & 3 & 3 & -1 & -1 \\ 1 & -3 & -3 & 1 & 1 \\ -1 & 3 & 3 & -1 & -1 \\ 1 & -3 & -3 & 1 & 1 \\ 3 & -1 & -1 & 3 & 3 \\ 3 & -1 & -1 & 3 & 3 \end{bmatrix}; AW^{T}= [1\ 0\ 1\ 0\ 1]\ W^{T}= [1\ 5\ 5\ 1] \Rightarrow (1\ 1\ 1\ 1) = L$$

$$B.W^{T} = [1\ 0\ 1\ 0\ 0\ 0] \begin{bmatrix} -1 & 3 & 3 & -1 & -1 \\ 1 & -3 & -3 & 1 & 1 \\ -1 & 3 & 3 & -1 & -1 \\ 1 & -3 & -3 & 1 & 1 \\ 3 & -1 & -1 & 3 & 3 \\ 3 & -1 & -1 & 3 & 3 \end{bmatrix} = [-2\ 6\ 6\ -2\ -2] \Rightarrow (0\ 1\ 1\ 0\ 0) = M$$

$$C.W^{T} = [0\ 1\ 0\ 1\ 1\ 1] \begin{bmatrix} -1 & 3 & 3 & -1 & -1 \\ 1 & -3 & -3 & 1 & 1 \\ -1 & 3 & 3 & -1 & -1 \\ 1 & -3 & -3 & 1 & 1 \\ 3 & -1 & -1 & 3 & 3 \\ 3 & -1 & -1 & 3 & 3 \end{bmatrix} = [8\ -8\ -8\ 8\ 8] \Rightarrow (0\ 1\ 1\ 0\ 0) = N$$

Sử dụng công thức, ví dụ trên có thể được xác định lại:

$$LW= [1\ 1\ 1\ 1] \begin{bmatrix} -1 & 1 & -1 & 1 & 3 & 3 \\ 3 & -3 & 3 & -3 & -1 & -1 \\ 3 & -3 & 3 & -3 & -1 & -1 \\ -1 & 1 & -1 & 1 & 3 & 3 \\ -1 & 1 & -1 & 1 & 3 & 3 \end{bmatrix} = [4\ -4\ 4\ -4\ 4]^T \Rightarrow (1\ 0\ 1\ 0\ 1)=A$$

Như vậy *(A, L), (B, M), (C < N)* là các điểm cố định cho BAM bởi công thức và ma trận trọng W xác định nhờ công thức

Hơn nữa nếu cho vector (A+S) (thay đổi một chỳt vector A) vào BAM, thì nó vẫn hội tụ gần nhất tới lỗi nhóm L. Vớ dụ

$$S = (0\ 1\ 0\ 0\ 0);\ A+S = (1\ 1\ 1\ 0\ 1);\ (A+S)\ W^T = (2\ 2\ 2\ 2) \quad thì \Rightarrow (1\ 1\ 1\ 1)=L$$

Chú ý rằng việc học với *BAM* là cố định, vì vậy nó không đủ mạnh trong trường hợp đẩy 1 bit trong 1 mẫu có thể kết quả hội tụ được là sai. *BAM* là ví dụ của mạng ánh xạ.

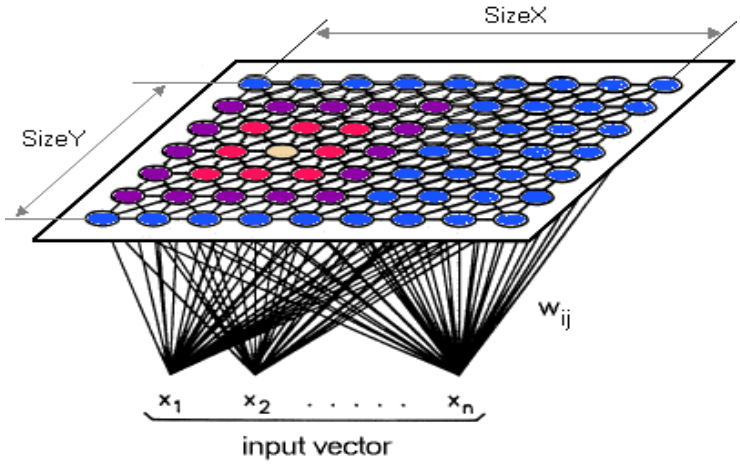
4.5 Mạng nơ ron tự tổ chức

Mạng nơ ron tự tổ chức *SOM (Self-Organizing Map)* được đề xuất bởi giáo sư *Teuvo Kohonen* vào năm 1982. Nó còn gọi là: Bản đồ/Ánh xạ đặc trưng tự tổ chức (*SOFM-Self Organizing Feature Map*) hay đơn giản hơn là mạng nơ ron *Kohonen*. *SOM* được coi là một trong những mạng nơ ron hữu ích nhất cho việc mô phỏng quá trình học của não người. Nó không giống với các mạng nơ ron khác chỉ quan tâm đến giá trị và dấu hiệu của thông tin đầu vào, mà có khả năng khai thác các mối liên hệ có tính chất cấu trúc bên trong không gian dữ liệu thông qua một bản đồ đặc trưng. Bản đồ đặc trưng bao gồm các nơ ron tự tổ chức theo các giá trị đầu vào nhất định; do đó nó có thể được huấn luyện để tìm ra các quy luật và sự tương quan giữa các giá trị nhập vào, từ đó dự đoán các kết quả tiếp theo. Có thể nói, nếu một hệ thống mô phỏng quá trình học của não người được thực hiện thì bản đồ đặc trưng của *SOM* đóng vai trò như trái tim của hệ thống.

Tính tự tổ chức của *SOM* được thực hiện bởi nguyên tắc học cạnh tranh, không giám sát nhằm tạo ra ánh xạ của dữ liệu từ không gian nhiều chiều về không gian ít chiều hơn (thường là hai chiều), nhưng vẫn đảm bảo được quan hệ về mặt hình trạng của dữ liệu. Điều này có nghĩa là các dữ liệu có đặc trưng tương đồng nhau thì sẽ được đại diện bởi cùng một nơ ron hoặc các nơ ron gần nhau và các nơ ron gần nhau thì sẽ tương đồng với nhau hơn so với những nơ ron ở xa. Kết quả là hình thành bản đồ đặc trưng của tập dữ liệu. Đây thực chất là một phép chiếu phi tuyến tạo ra "ánh xạ đặc trưng" cho phép phát hiện và phân tích những đặc trưng trong không gian đầu vào; do đó, *SOM* là một công cụ hiệu quả cho việc phân cụm trực quan và phân tích dữ liệu nhiều chiều.

4.5.1 Mô hình cấu trúc của mạng Kohonen

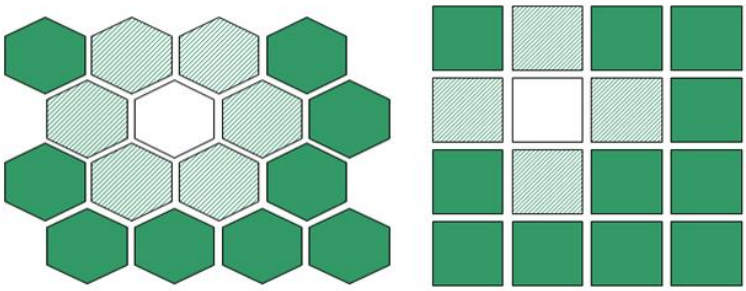
Mạng nơ ron *SOM* có cấu trúc đơn lớp, bao gồm: các tín hiệu vào và lớp ra *Kohonen* (*Hình 4.9*), trong đó tất cả các đầu vào được kết nối đầy đủ với mọi nơ ron trên lớp ra Kohonen. Kiến trúc mạng của *SOM* thuộc đồng thời cả hai nhóm mạng truyền thẳng và mạng phản hồi, do dữ liệu được truyền từ đầu vào tới đầu ra và có sự ảnh hưởng giữa các nơ ron trong lớp *Kohonen*. Lớp *Kohonen* thường được tổ chức dưới dạng một ma trận 2 chiều các nơ ron theo dạng lưới hình chữ nhật hoặc hình lục giác. Mỗi đơn vị *i* (nơ ron) trong lớp *Kohonen* được gắn một vector trọng số $w_i = [w_{i,1}, w_{i,2}, \ldots, w_{i,n}]$, với *n* là kích thước (số chiều) vector đầu vào; $w_{i,j}$ là trọng số của nơ ron *i* ứng với đầu vào *j*.



Hình 4.9. Cấu trúc mạng *SOM* với lớp *Kohonen* 2 chiều

Các nơ ron của lớp ra được sắp xếp trên một mảng 2 chiều. Mảng này được gọi là lớp ra *Kohonen*. Lớp đầu ra này rất khác với lớp đầu ra của mạng nơ ron truyền thẳng. Đối với mạng truyền thẳng, nếu chúng ta có một mạng nơ ron với 5 nơ ron đầu ra, chúng sẽ có thể cho kết quả bao gồm 5 giá trị. Còn trong mạng nơ ron *Kohonen* chỉ có một nơ ron đầu ra cho ra một giá trị. Giá trị duy nhất này có thể là đúng hoặc sai. Dữ liệu đầu ra từ mạng nơ ron Kohonen thường là các chỉ số của nơ ron.

Trong trường hợp lưới hai chiều, các nơ ron nằm trên bản đồ có thể tồn tại hai loại cấu trúc liên kết là hình lục giác hoặc hình chữ nhật. Tuy nhiên, cấu trúc liên kết hình lục giác đều thì tốt hơn trong tác vụ trực quan hoá vì mỗi nơ ron có 6 nơ ron lân cận trong khi với cấu trúc hình chữ nhật thì chỉ là 4.



Hình 4-10: Cấu trúc hình lục giác đều và cấu trúc hình chữ nhật.

trong đó:

- Lớp vào (Input Layer): dùng để đưa dữ liệu huấn luyện vào mạng *Kohonen*. Kích thước lớp vào tương ứng với kích thước của mỗi mẫu học.
- Lớp ra (Output Layer): các nơ ron của lớp ra được sắp xếp trên mảng hai chiều. Mảng này gọi là lớp ra *Kohonen*.
- Tất cả các noron của lớp vào đều được nối với các nơ ron trên lớp ra. Mỗi liên kết giữa đầu vào và đầu ra của mạng *Kohonen* tương ứng với một trọng số. Kích thước của mỗi véc tơ trọng số bằng kích thước của lớp vào. Nói cách khác, mỗi nơ ron của lớp *Kohonen* sẽ có thêm một vector trọng số *n* chiều (với *n* là số đầu vào).

4.5.2 Học ganh đua

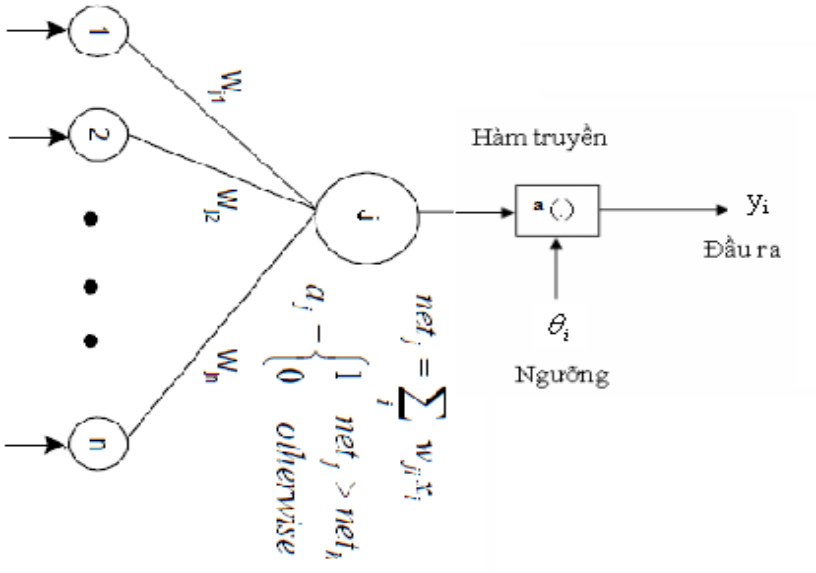
SOM là một kỹ thuật mạng nơ ron truyền thẳng sử dụng thuật toán học không giám sát (học ganh đua) và qua quá trình "tự tổ chức", sắp xếp đầu ra cho một thể hiện hình học của dữ liệu ban đầu.

Học không giám sát liên quan đến việc dùng các phương pháp quy nạp để phát hiện tính quy chuẩn được thể hiện trong tập dữ liệu. Mặc dù có rất nhiều thuật toán mạng nơ ron cho học không giám sát, trong đó có thuật toán học ganh đua (*Conpetitive Learning*, *Rumelhart & Zipser, 1985*). Học ganh đua có thể coi là thuật toán học mạng nơ ron không giám sát thích hợp nhất trong khai phá dữ liệu, và nó cũng minh họa cho sự phù hợp của các phương pháp học mạng nơ ron một lớp.

Nhiệm vụ học xác định bởi học ganh đua là sự phân chia một ví dụ huấn luyện cho trước vào trong một tập các cụm dữ liệu. Các cụm dữ liệu sẽ thể hiện các luật biểu diễn trong tập dữ liệu như các minh hoạ giống nhau được ánh xạ vào các lớp giống nhau.

Biến thể của học ganh đua mà chúng ta xét ở đây đôi khi được gọi là học ganh đua đơn điệu, liên quan đến việc học trong mạng nơ ron một lớp. Các đơn vị đầu vào trong mạng có các giá trị liên quan đến lĩnh vực đang xét, và *k* đơn vị đầu ra thể hiện *k* lớp ví dụ đầu

vào được phân cụm.



Hình 4-11: Đơn vị (nơ ron) xử lý ganh đua

Giá trị đầu vào cho mỗi đầu ra trong phương pháp này là một tổ hợp tuyến tính của các đầu vào:

$$net_j = \sum_i w_{ji} x_i \qquad (4.48)$$

Trong đó:

+ x_i là đầu vào thứ i; $i = 1,2,...,n$.
+ w_{ji} là trọng số liên kết đầu vào thứ i với đầu ra thứ j, $j = 1,2, ...,m$.

Gọi $S(net_j)$ là hàm chuyển tín hiệu (hàm tương tác hay hàm kích hoạt đầu ra), có thể là hàm đơn điệu không giảm liên tục như hàm Sigmoid hoặc hàm bước nhẩy đơn vị sau:

$$a_j = \begin{cases} 1 & if \sum_i w_{ji} x_i > \sum_i w_{hi} x_i \forall h \neq j \\ 0 & else \end{cases} \qquad (4.49)$$

Đơn vị đầu ra có giá trị đầu vào lớn nhất được coi là chiến thắng, và kích hoạt đó được coi bằng 1 (thắng cuộc), còn các kích hoạt khác của các đầu ra còn lại được cho bằng 0 (thua cuộc). Quá trình như vậy được gọi là ganh đua.

Quá trình huấn luyện cho học ganh đua liên quan đến hàm chi phí:

$$C = \frac{1}{2} \sum_j \sum_i a_j (x_i - w_{ji})^2 \qquad (4.50)$$

trong đó:

+ a_j là kích hoạt của đầu ra thứ *j*.
+ x_i là đầu vào thứ *i*.
+ w_{ji} là trọng số từ đầu vào thứ *i* với đầu ra thứ *j*.

Luật cập nhật các trọng số là:

$$\Delta w_{ji} = -\alpha \partial C \partial w_{ji} = \alpha a_j (x_i - w_{ji}) \qquad (4.51)$$

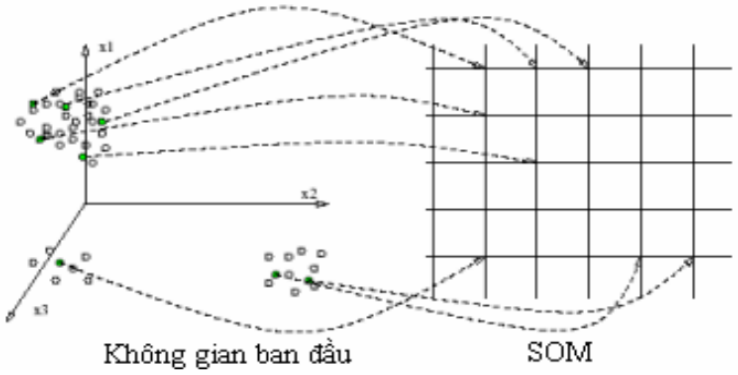
với α là hằng số, chỉ tốc độ học.

Ý tưởng chính của học ganh đua là đối với mỗi đầu ra là lấy ra "độ tin cậy" cho tập con các ví dụ huấn luyện. Chỉ một đầu ra là chiến thắng trong số ví dụ đưa ra, và vectơ trọng số cho đơn vị chiến thắng được di chuyển về phía vectơ đầu vào. Giống như quá trình huấn luyện, vectơ trọng số của mỗi đầu ra di chuyển về phía trung tâm của các ví dụ. Huấn luyện xong, mỗi đầu ra đại diện cho một nhóm các ví dụ, và vectơ trọng số cho các đơn vị phù hợp với trọng tâm của các nhóm.

Học ganh đua có liên quan mật thiết với phương pháp thống kê nổi tiếng như là phương pháp phân cụm *K* thành phần chính. Khác nhau cơ bản giữa hai phương pháp là học ganh đua là phương pháp trực tuyến, nghĩa là trong suốt quá trình học nó cập nhập trọng số mạng sau mỗi ví dụ được đưa ra, thay vì sau tất cả các ví dụ được đưa ra như được làm trong phương pháp phân cụm *K* thành phần chính. Học ganh đua phù hợp với các tập dữ liệu lớn, vì các thuật toán trực tuyến thường có giải pháp nhanh hơn trong mọi trường hợp.

4.5.3 Thuật toán *SOM*

Về bản chất, *SOM* được biết đến như một kỹ thuật nén dữ liệu dựa trên vecto trọng số (trực quan hóa dữ liệu).



Hình 4-12: Không gian ban đầu và không gian sau khi thực hiện thuật toán *SOM*

Input: Dữ liệu huấn luyện gồm tập *n* vectơ: $V=\{V_1, V_2, ..., V_i, ..., V_n\}$, mỗi vectơ ứng với một nơ ron (nút) đầu vào**;** Trong đó, mỗi vecto V_i gồm *p* chiều: $V_i=\{v_1, v_2, ..., v_p\}$.

Khởi tạo tham số số lần lặp *t=1*

- **Bước 1**: Khởi tạo vecto trọng số cho mỗi nơ ron

Tương ứng với mỗi vector V_i, vecto trọng số W_i cũng gồm p chiều

$W_i=\{w_1, w_2, \ldots, w_p\}$

Và tập vecto trọng số có m bộ: $W=\{W_1, W_2, ..., W_i, ..., W_m\}$

- **Bước 2**: Chọn ngẫu nhiên một vecto $V_i \epsilon V$ làm mẫu huấn luyện
- **Bước 3:** Tìm mẫu khớp tốt nhất BMU (Best Matching Unit)–phần tử nơ ron chiến thắng

Tìm phần tử khớp nhất giữa các vecto trọng số $Wi \epsilon W$ và vecto đầu vào V_i. Nơ ron nào có vecto trọng số W_i gần với vecto đầu vào V_i nhất thì nơ ron đó được chọn là BMU.

Để xác định BMU, ta tính khoảng cách Euclid giữa các vecto trọng số W_i với vecto V_i chọn ở Bước 2 theo công thức sau:

$$Dist_1 = \sqrt{\sum_{i=1}^{p}(V_i - W_i)^2} \qquad (4.52)$$

trong đó:

+ $Dist_1$: khoảng cách giữa vecto trọng số W_i và vecto đầu vào V_i
+ V_i: vecto đầu vào đang xét
+ W_i: vecto trọng số của phần tử được chọn
+ $Dist_1$ min thì nơ ron có vecto trọng số tương ứng được chọn là BMU.

- **Bước 4**: Xây dựng các phần tử lân cận

Bước này sẽ xác định các nơ ron nào thuộc vùng lân cận của *BMU*. Để xác định vùng lân cận của *BMU*, tính bán kính lấy tâm là *BMU* tới các nơ ron còn lại, gọi là bán kính lân cận. Nơ ron nào có khoảng cách tới *BMU* nằm trong bán kính lân cận thì nơ ron đó là phần tử lân cận của *BMU*. Bán kính lân cận được xác định lớn nhất thường là bán kính tính theo kích thước của mạng, nhưng sau đó giá trị này sẽ giảm dần sau một số bước thực hiện.

Bán kính lân cận của *BMU* tại thời điểm t được xác định bằng công thức:

$$\sigma(t) = \sigma_0 . \exp\left(-\frac{t}{\lambda}\right) \qquad (4.53)$$

*t*rong đó:

+ $\sigma(t)$: bán kính lân cận của *BMU* tại thời điểm *t.*
+ σ_0: bán kính lân cận của *BMU* tại thời điểm t_0.
+ σ_0 được tính bằng công thức: $\sigma_0 = \max(width, height)\ /2.$
+ *Width*: chiều rộng mạng *Kohonen* (do người dùng tự định nghĩa).

+ Height: chiều dài mạng *Kohonen* (do người dùng tự định nghĩa).
+ t: bước lặp hiện tại.
+ λ: hằng số thời gian. Trong đó: $\lambda = \frac{N}{\log(\sigma_0)}$
+ N: số lần lặp để chạy thuật toán.

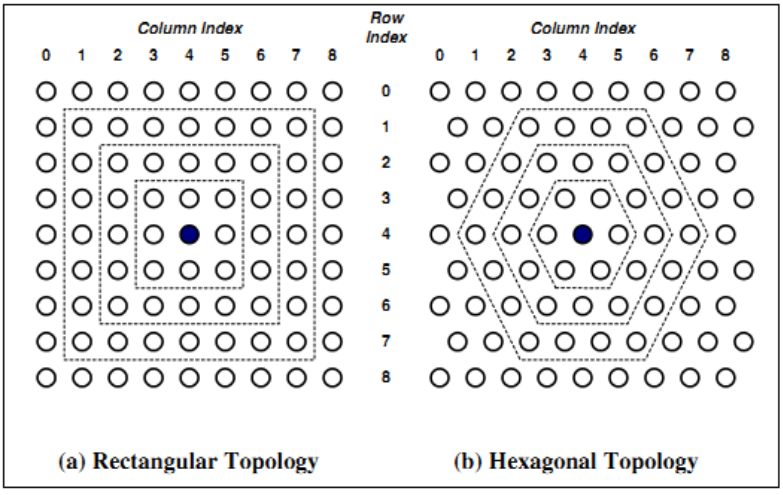
Sau khi tính được bán kính lân cận, ta tính khoảng cách từ *BMU* tới các nơ ron còn lại để xác định nơ ron nào là phần tử lân cận của *BMU*. Nếu khoảng cách đó nhỏ hơn bán kính thì nơ ron tương ứng là phần tử lân cận của *BMU*.

Khoảng cách từ *BMU* tới các nơ ron được tính theo công thức Euclid:

$$Dist_2 = \sqrt{\sum_{i=1}^{p}(BMU_i - W_i)^2} \qquad (4.54)$$

$Dist_2$: khoảng cách từ *BMU* tới các nơ ron còn lại.

Các phần tử lân cận bao gồm *BMU* sẽ được cập nhật lại trọng số ở bước tiếp theo.



Hình 4-13: **Bán kính lân cận và phần tử lân cận sau một số lần lặp**

a) Lưới hình chữ nhật; b) Lưới hình lục giác

- **Bước 5**: Hiệu chỉnh trọng số các phần tử lân cận – Quá trình học của *SOM*

Trọng số của các phần tử lân cận đã xác định ở bước 4 bao gồm cả *BMU* sẽ được hiệu chỉnh để chúng có giá trị gần giống với giá trị của vecto đầu vào đang xét.

Các vecto trọng số được hiệu chỉnh theo công thức:

$$W(t+1)=W(t) + \theta(t)L(t)(V(t)-W(t)) \qquad (4.55)$$

trong đó:

+ *W(t+1):* vecto trọng số tại bước lặp (t+1).

+ *W(t):* vecto trọng số tại bước lặp *t*.
+ $\theta(t)$*:* hàm nội suy theo thời gian học, nó thể hiện sự tác động của khoảng cách đối với quá trình học.

Hàm nội suy $\theta(t)$ được tính theo công thức:

$$\theta(t) = exp\ (-\frac{Dist_2{}^2}{2\sigma^2(t)}) \qquad (4.56)$$

trong đó:

+ $Dist_2$: khoảng cách từ *BMU* tới các phần tử lân cận.
+ *L(t):* hàm nội suy tốc độ học cho mỗi bước lặp được tính theo công thức:

$$L(t)=L_0.\ exp(-\frac{t}{\lambda}) \qquad (4.57)$$

+ L_0: giá trị khởi tạo ban đầu của tốc độ học.

Tốc độ học được nội suy dần sau một số lần lặp và giá trị của hàm sẽ tiền dần về 0 khi số lần lặp đạt tới những bước cuối cùng.

- **Bước 6**: Tăng *t,* lấy mẫu huấn luyện tiếp theo

Lặp lại bước 2 cho đến giải thuật tối ưu tức *W(t+1)=W(t)* hoặc đạt đến số lần lặp xác định *N* cho trước (*t=N*). Thuật toán dừng khi thực hiện đủ số lần lặp hoặc không có sự thay đổi nào của các vecto trọng số.

Quá trình thực hiện thuật toán *SOM* được tóm tắt như sau:

+ ***Bước 1****: Khởi tạo giá trị cho các vecto trọng số.*
+ ***Bước 2****: Chọn một vecto từ tập vecto đầu vào.*
+ ***Bước 3****: Tìm mẫu khớp tốt nhất (Best Matching Unit - BMU)*
  Tính toán khoảng cách giữa vecto đầu vào với tất cả các vecto trọng số theo công thức Euclid:

$$Dist = \sqrt{\sum_{i=1}^{p}(V_i - W_i)^2} \qquad (4.58)$$

  Dist min thì nơ ron có vecto trọng số tương ứng được chọn làm BMU
+ ***Bước 4****: Tìm các phần tử lân cận.*
+ ***Bước 5****: Cập nhật lại trọng số của các phần tử lân cận và BMU*

$$W(t+1)=W(t) + \theta(t)L(t)(V(t)-W(t))$$

+ ***Bước 6****: Tăng t, lặp tiếp bước 2.*

4.5.4 SOM với bài toán phân cụm

Với khả năng mạnh mẽ trong việc biểu diễn dữ liệu từ không gian nhiều chiều về không gian ít chiều hơn mà vẫn có thể bảo tồn được quan hệ hình trạng của dữ liệu trong không gian đầu vào,

nên chức năng chính của *SOM* là trình diễn cấu trúc của toàn bộ tập dữ liệu và giúp quan sát trực quan cấu trúc cũng như sự phân bố tương quan giữa các mẫu dữ liệu trong không gian của tập dữ liệu. Do đó, *SOM* được ứng dụng rất nhiều trong các bài toán thực tế, từ những bài toán có tính chất nền tảng của khai phá dữ liệu như phân cụm, phân lớp cho tới những bài toán ứng dụng trong các lĩnh vực khác. Cụ thể, từ năm 1980 đến nay, đã có hàng nghìn bài báo, công trình nghiên cứu liên quan đến *SOM*, được liệt kê trong các *"Bibliography of selforganizing map (SOM) papers*. Trong những năm gần đây, có thể kể ra một số nghiên cứu ứng dụng *SOM* đại diện trong các lĩnh vực như: thị giác máy và phân tích ảnh, nhận dạng và phân tích tiếng nói, phân tích dữ liệu y tế, xử lý tín hiệu trong viễn thông, công nghiệp và các dữ liệu thế giới thực khác, xử lý dữ liệu video giao thông, xử lý các loại dữ liệu liên tục theo thời gian...

SOM là phương pháp phân cụm theo cách tiếp cận mạng nơ ron và thuật toán học ganh đua. Vectơ trọng số của ma trận *SOM* chính là trọng tâm cụm, việc phân cụm có thể cho kết quả tốt hơn bằng cách kết hợp các đơn vị trong ma trận để tạo thành các cụm lớn hơn. Một điểm thuận lợi của phương pháp này là vùng *Voronoi* của các đơn vị ma trận là lồi, bằng cách kết hợp của một số đơn vị trong ma trận với nhau tạo nên các cụm không lồi. Việc sử dụng các độ đo khoảng cách khác nhau và các chuẩn kết liên kết khác nhau có thể tạo thành các cụm lớn hơn.

Ma trận khoảng cách: chiến lược chung trong phân cụm các đơn vị của *SOM* là tìm ma trận khoảng cách giữa các vectơ tham chiếu và sử dụng giá trị lớn trong ma trận như là chỉ số của đường biên cụm. Trong không gian ba chiều, các cụm sẽ được thể hiện như "các thung lũng". Vấn đề là làm sao để quyết định các đơn vị trong ma trận thuộc về một cụm nào đó cho trước.

Để giải quyết được vấn đề này, người ta thường sử dụng thuật toán tích tụ (*Agglomerative Algorithm*), gồm các bước:

1. Quy cho mỗi đơn vị trong ma trận một cụm riêng.
2. Tính toán khoảng cách giữa tất cả các cụm.
3. Ghép hai cụm gần nhất.
4. Nếu số cụm tồn tại bằng số cụm do người dùng định nghĩa trước thì dừng, nếu không lặp lại từ bước 2.

SOM cũng được áp dụng trong phân cụm tập dữ liệu không chuẩn hoá. Dùng luật của học ganh đua, vectơ trọng số có thể điều chỉnh theo hàm phân bố xác suất của các vectơ đầu vào. Sự tương đồng giữa vectơ đầu vào *x* và vectơ trọng số *w* được tính toán bằng khoảng cách *Oclit*. Trong suốt quá trình huấn luyện một vectơ trọng số w_j tuỳ ý được cập nhập tại thời điểm *t* là:

$$\Delta w_j(t) = \alpha(t) h_{cj}(t) \left[x(t) - w_j(t) \right] \qquad (4.59)$$

Với $\alpha(t)$ là tỷ lệ học giảm dần trong quá trình huấn luyện, và $h_{Ci}(t)$ là hàm lân cận giữa vectơ trọng số chiến thắng w_C, và vectơ trọng số w_j , $h_{Ci}(t)$ cũng giảm dần trong quá trình huấn luyện. Mối quan hệ lân cận được xác định bằng cấu trúc hình học và mối quan hệ này cố định trong suốt quá trình học. Kết thúc quá trình học, điều chỉnh lại bán kính lân cận đủ nhỏ để cập nhập lại cho các vectơ trọng số chiến thắng w_C và các lân cận gần chúng nhất. Đối với cấu trúc một chiều nó có thể được biểu diễn bằng luật huấn luyện. Công thức trên là một sấp xỉ của hàm đơn điệu của phân bố xác suất trên các vectơ đầu vào. Trong cấu trúc hai chiều thì kết quả trả về là một sự tương quan giữa độ xấp xỉ và bình phương lỗi tối thiểu của vectơ lượng tử.

Ưu điểm, nhược điểm của thuật toán SOM

a) Ưu điểm

- Ưu điểm tốt nhất của huật toán *SOM* là nó rất dễ hiểu. *SOM* được cài đặt đơn giản và làm việc rất tốt.
- Thuật toán *SOM* có ưu điểm chính là biểu diễn trực quan dữ liệu đa chiều vào không gian ít chiều hơn và đặc trưng của dữ liệu được giữ lại trên bản đồ.
- Thuật toán *SOM* rất có hiệu quả trong quá trình phân tích đòi hỏi trí thông minh để đưa ra quyết định nhanh chóng trên thị trường. Nó giúp cho người phân tích hiểu vấn đề hơn trên một tập dữ liệu tương đối lớn.
- Thuật toán *SOM* xác định các cụm dữ liệu giúp cho việc tối ưu phân bổ nguồn lực
- Thuật toán *SOM* được ứng dụng rất nhiều trong nhận dạng tiếng nói, điều khiển tự động, hóa-sinh trắc học, phân tích tài chính và xử lý ngôn ngữ tự nhiên…

b) Nhược điểm

- Chi phí cho việc tính toán lớn khi số chiều của dữ liệu tăng lên
- Việc xác định ranh giới giữa các nhóm trên lớp ra *Kohonen* gặp nhiều khó khăn, trong trường hợp dữ liệu biến thiên liên tục, sự phân chia giữa các nhóm là rất nhỏ.
- Khối lượng tính toán của thuật toán *SOM* là tương đối lớn, do vậy tốc độ xử lý của giải thuật cũng là một thách thức cần xét tới. Xét một mạng Kohonen sử dụng thuật toán *SOM* với kích thước *20x30=600* nơ ron, độ phân giải bức ảnh đầu vào được tính bằng đơn vị megapixel tức là có tới hàng triệu điểm ảnh. Như vậy, riêng trong quá trình huấn luyện, việc tìm *BMU* đã phải duyệt qua khoảng 600 triệu lần các nơ ron mà chưa tính đến các phép tính khoảng cách, so sánh, cập nhật trọng số…
- Một vấn đề nữa là mỗi lần chạy thuật toán *SOM*, ta đều tìm thấy độ tương tự khác nhau giữa các vecto mẫu. Các vecto mẫu thường được bao quanh bởi mẫu tương tự, tuy nhiên các mẫu tương tự không phải lúc nào cũng gần nhau. Nếu có rất nhiều sắc thái màu tím, không phải

lúc nào cũng thu được một tập lớn tất cả các màu tìm trong một cụm, đôi khi các cụm sẽ phân chia và sẽ có hai nhóm màu tím. Sử dụng dữ liệu màu sắc, ta có thể nói hai cụm dữ liệu này là tương tự nhưng với hầu hết các dữ liệu khác, hai cụm trông hoàn toàn không liên quan tới nhau. Vì vậy, cần lưu ý rất nhiều để xây dựng được một bản đồ tốt cuối cùng.

CÂU HỎI VÀ BÀI TẬP

1. Cơ sở của nơ ron nhân tạo là gì? Hãy mô tả các thành phần của nơ ron sinh học?

2. Mô tả các thành phần của nơ ron nhân tao

3. Mô tả mô hình của mạng Perceptron và luật học

4. Mô tả mạng lan truyền ngược và luật học

5. Mô tả mạng BAM và thuật học

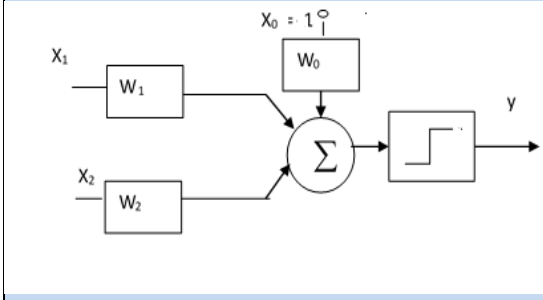
6. Cho tập dữ liệu đầu vào (véc tơ đặc trưng đầu vào), dữ liệu đầu ra (véc tơ đặc trưng đầu ra)

Vector đặc trưng	Vector nhãn
A=(1 0 1 0 11) A'=(1-1 1-1 1 1)	L=(1 1 1 1 1) L'=(1 1 1 11)
B=(1 0 1 0 0 0) B'=(1-1 1-1-1-1)	M=(0 1 1 0 0) M'=(-1 1 1 -1-1)
C=(0 1 0 1 1 1) C'=(-1 1-1 1 1 1)	N=(1 0 0 1 1) N'=(1 -1 -1 11)

a) Thiết kế mạng BAM; vẽ sơ đồ chi tiết

b) Tính đầu ra N khi đầu vào C=(0 1 0 1 1 1) có sai số 1 bit trở thành R= (0 0 1 0 0 0) từ đó đưa ra kết luận.

7. Hãy thiết kế và tính toán bằng thuật toán Perceptron để xác định các tham số cho một nơ ron nhân tạo dùng làm mạch OR theo sơ đồ biểu diễn và các tham số khởi đầu như sau



Bảng chân lý của hàm OR

STT	x_1	x_2	$d(=y)= x_1$ OR x_2
1	0	0	0
2	0	1	1
3	1	0	1
4	1	1	1

Trọng số khơi tạo (cho ban đầu)

w_0= 0,2; w_1= 0,4; w_2= 0,5; x_0=+1

Tốc độ học: α = 0,3; Sai số cần đạt: e=0.

CHƯƠNG 5:

GIẢI THUẬT DI TRUYỀN

5.1 Khái niệm về giải thuật di truyền

Giải thuật di truyền (Genetic Algorithm: GA) là kỹ thuật chung giúp giải quyết vấn đề-bài toán bằng cách mô phỏng sự tiến hóa của con người hay của sinh vật nói chung (dựa trên thuyết tiến hóa muôn loài của Darwin) trong điều kiện qui định sẵn của môi trường. Mục tiêu của GA không đưa ra lời giải chính xác mà đưa ra lời giải tương đối tối ưu.

Mục tiêu của GA được khái quát như sau:

- Trừu tượng hoá và mô phỏng quá trình thích nghi trong hệ thống tự nhiên.
- Thiết kế phần mềm, chương trình mô phỏng, nhằm duy trì các cơ chế quan trọng của hệ thống tự nhiên.

Giải thuật di truyền sử dụng một số thuật ngữ của ngành di truyền học như: NST, quần thể (Population), Gen... NST được tạo thành từ các Gen (được biểu diễn một chuỗi tuyến tính từ các Gen). Mỗi Gen mang một số đặc trưng và có vị trí nhất định trong NST. Mỗi NST sẽ biểu diễn một lời giải của bài toán. Bảng dưới đây cho biết những khái niệm về thuật ngữ và tham số cơ bản của sinh học và chuyển đổi sang CNTT.

STT	Sinh học	Công nghệ Thông tin
1	Gen	Hệ đếm: Nhị phân, Bát phân, Hecxa, Thập phân
2	Nhiễm sắc thể	Tập hợp *n* bit. Ví dụ, *n=5* cụ thể 1 NST[01100]
3	Quần thể	Tập hợp nhiều NST (011001, 00000, 11111)
4	Thế hệ	

5.2 Các toán tử trong giải thuật di truyền

5.2.1 Toán tử sinh sản

Toán tử sinh sản gồm hai quá trình: sinh sản (phép tái sinh), chọn lọc (phép chọn).

a) Phép tái sinh: là quá trình các NST được sao chép trên cơ sở độ thích nghi. Độ thích nghi là một hàm được gán giá trị thực, tương ứng với mỗi NST trong quần thể. Quá trình này, được mô tả như sau:

Xác định độ thích nghi của từng NST trong quần thể ở thế hệ thứ t, lập bảng cộng dồn các giá trị thích nghi (theo thứ tự gán cho từng nhiễm sắc thể). Giả sử, quần thể có *n* cá thể. Gọi độ thích nghi của NST_i tương ứng là *fi* tổng cộng dồn thứ *i* là f_{ti} được xác định bởi:

$$f_{tj} = \sum_{j=1}^{t} f_j \qquad (5.1)$$

Gọi F_n là tổng độ thích nghi của toàn quần thể. Chọn một số ngẫu nhiên *f* trong khoảng từ 0 tới F_n. Chọn cá thể thứ k đầu tiên thoả mãn $f \geq f_{tk}$ đưa vào quần thể mới.

b) Phép chọn: là quá trình loại bỏ các NST kém thích nghi trong quần thể. Quá trình này được mô tả như sau:

- Sắp xếp quần thể theo thứ tự mức độ thích nghi giảm dần.
- Loại bỏ các NSTở cuối dãy. Giữ lại n cá thể tốt nhất.

5.2.2 Toán tử ghép chéo

Ghép chéo là quá trình tạo NST mới trên cơ sở các NST cha-mẹ bằng cách ghép một đoạn trên NST cha-mẹ với nhau. Toán tử ghép chéo được gán với một xác suất *pc*. Quá trình được mô tả như sau:

- Chọn ngẫu nhiên một cặp NST (cha-mẹ) trong quần thể. Giả sử, NST cha-mẹ có cùng độ dài m.
- Tạo một số ngẫu nhiên trong khoảng từ 1 tới *m-1* (gọi là điểm ghép chéo). Điểm ghép chéo chia NSTcha-mẹ thành hai chuỗi con có độ dài m_1, m_2. Hai chuỗi con mới được tạo thành là: $m_{11}+ m_{22}$ *và* $m_{21}+m_{12}$. Đưa hai NST mới vào quần thể.

5.2.3 Toán tử đột biến

Đột biến là hiện tượng NST con mang một số đặc tính không có trong mã di truyền của cha-mẹ.

• Chọn ngẫu nhiên một NST trong quần thể;

• Tạo một số ngẫu nhiên *k* trong khoảng từ 1 tới $m, 1 \leq k \leq m$;

• Thay đổi bit thứ *k*. Đưa NST này vào quần thể để tham gia quá trình tiến hoá ở thế hệ tiếp theo.

5.3 Giải thuật di truyền

5.3.1 Các bước cơ bản của giải thuật di truyền

Một giải thuật di truyền đơn giản bao gồm các bước sau:

Bước 1: Khởi tạo một quần thể ban đầu gồm các chuỗi nhiễm sắc thể.

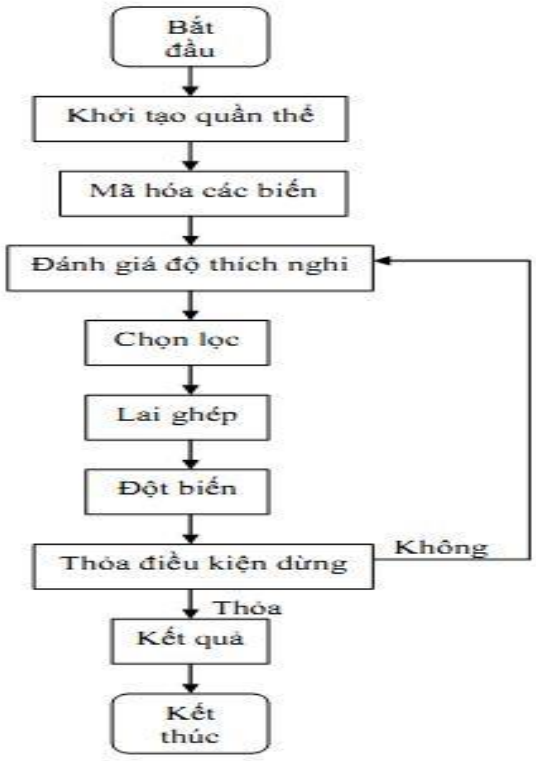
Bước 2: Xác định giá trị mục tiêu cho từng NST tương ứng.

Bước 3: Tạo các NST mới dựa trên các toán tử di truyền.

Bước 4: Xác định hàm mục tiêu cho các NST mới và đưa vào quần thể.

Bước 5: Loại bớt các NST có độ thích nghi thấp.

Bước 6: Kiểm tra thỏa mãn điều kiện dừng. Nếu điều kiện đúng, lấy ra NST tốt nhất, giải thuật dừng lại; ngược lại, quay về bước 3.



Hình 5.1 : Cấu trúc thuật giải di truyền tổng quát

Bắt đầu

t =0;

Khởi tạo P(t)

Tính độ thích nghi cho các cá thể thuộc P(t);

Khi (điều kiện dừng chưa thỏa) lặp

t = t + 1;

Chọn lọc P(t);

Lai ghép P(t);

Đột biến P(t);

Hết lặp

Kết thúc

5.3.2 Các công thức của giải thuật di truyền

Tính độ thích nghi *eval(v_i)* của mỗi NST v_i (*i =1*..kích thước quần thể):

$$eval(v_i) = \frac{f(v_i)}{\sum_{i=1}^{kích\ thước\ quần\ thể} f(v_i)} \quad (5.2)$$

Với *f (vi)* là hàm mục tiêu

Tìm tổng giá trị thích nghi của quần thể

$$F = \sum_{i=1}^{kích\ thước\ quần\ thể} eval(v_i) \quad (5.3)$$

Tính xác suất chọn p_i cho mỗi NST v_i

$$p_i = \frac{eval(v_i)}{\sum_{i=1}^{kích\ thước\ quần\ thể} eval(v_i)} \quad (5.4)$$

Tính xác suất tích lũy q_i cho mỗi NSTv_i

$$q_i = \sum_{j=1}^{i} p_i \quad (5.5)$$

Tiến trình chọn lọc được thực hiện bằng cách quay bánh xe rulet trên kích thước quần thể. Mỗi lần chọn ra một NST từ quần thể hiện hành vào quần thể mới theo cách sau: Phát sinh một số ngẫu nhiên *r* trong khoảng *[0, 1]*. Nếu $r < q_1$ thì chọn NST v_1, ngược lại chọn NSTv_i ($2 \le i \le$ *kích thước quần thể*) sao cho q_i-$1 < r \le q_i$

a. Hàm mục tiêu

Cứ sau mỗi thế hệ được hình thành, chúng ta cần tính lại độ thích nghi cho từng cá thể để chuẩn bị cho một thế hệ mới. Do số lượng các cá thể tăng lên, độ thích nghi giữa các cá thể không có sự chênh lệch đáng kể. Do đó, các cá thể có độ thích nghi cao chưa hẳn chiếm ưu thế trong thế hệ tiếp theo. Vì vậy, cần ấn định tỷ lệ đối với hàm thích nghi nhằm tăng khả năng cho các NST đạt độ thích nghi cao. Có 3 cơ chế định tỷ lệ như sau:

- ***Định tỷ lệ tuyến tính***

Độ thích nghi được xác định theo công thức:

$$f(v_i)' = a * f(v_i) + b \quad (5.6)$$

Cần chọn các tham số a, b sao cho độ thích nghi trung bình được ánh xạ vào chính nó. Tăng độ thích nghi tốt nhất bằng cách nhân nó với độ thích nghi trung bình. Cơ chế này có thể tạo ra các giá trị âm cần xử lý riêng. Ngoài ra, các tham số *a, b* thường gắn với đời sống quần thể và không phụ thuộc vào bài toán.

- ***Phép cắt Sigma***

Phương pháp này được thiết kế vừa để cải tiến phương pháp định tỷ lệ tuyến tính vừa để xử lý các giá trị âm, vừa kết hợp thông tin mà bài toán phụ thuộc. Ở đây, độ thích nghi mới được tính theo công thức:

$$f(v_i)' = f(v_i) + (\overline{f(v_i)} - c * \sigma) \quad (5.7)$$

Trong đó c là một số nguyên nhỏ (thường lấy giá trị từ 1 tới 5);σ là độ lệch chuẩn của quần thể. Với giá trị âm thì *f'* được thiết lập bằng 0.

- ***Định tỷ lệ cho luật dạng luỹ thừa***

Trong phương pháp này, độ thích nghi lúc khởi tạo có năng lực đặc biệt:

$$f(v_i)' = f(v_i)^k \quad (5.8)$$

với *k gần bằng 1*. Tham số *k* định tỷ lệ *hàm f(.)*. Tuy nhiên, một số nhà nghiên cứu cho rằng nên chọn k độc lập với bài toán. Bằng thực nghiệm cho thấy nên chọn *k =1.005*.

b. Điều kiện dừng của giải thuật

Chúng ta sẽ khảo sát điều kiện đơn giản nhất để dừng khi số thế hệ vượt quá một ngưỡng cho trước. Trong một số phiên bản về chương trình tiến hoá không phải mọi cá thể đều tiến hoá lại. Vài cá thể trong đó có khả năng vượt từ thế hệ này sang thế hệ khác mà không thay đổi gì cả. Trong những trường hợp như vậy, chúng ta đếm số lần lượng hàm. Nếu số lần lượng hàm vượt quá một hằng xác định trước thì dừng việc tìm kiếm.

Chúng ta nhận thấy, các điều kiện dừng ở trên giả thiết rằng người sử dụng đã biết đặc trưng của hàm, có ảnh hưởng như thế nào tới chiều dài tìm kiếm. Trong một số trường hợp khó có thể xác định số lượng thế hệ (hay lượng giá hàm) phải là bao nhiêu. Giải thuật có thể kết thúc khi cơ hội cho một cải thiện quan trọng chưa bắt đầu.

Có hai loại điều kiện dừng cơ bản. Các điều kiện này dùng các đặc trưng tìm kiếm để quyết định ngừng quá trình tìm kiếm:

- Dựa trên cấu trúc nhiễm sắc thể: do sự hội tụ của quần thể bằng cách kiểm soát số alen được hội tụ, ở đây alen được coi như hội tụ nếu một số phần trăm quần thể đã định trước có cùng (hoặc tương đương đối với các biểu diễn không nhị phân) giá trị trong alen này. Nếu số alen hội tụ vượt quá số phần trăm nào đó của tổng số alen, việc tìm kiếm sẽ kết thúc.
- Dựa trên ý nghĩa đặc biệt của một nhiễm sắc thể: đo tiến bộ của giải thuật trong một số thế hệ cho trước. Nếu tiến bộ này nhỏ hơn một hằng số ε xác định, kết thúc tìm kiếm.

5.4 Ví dụ về giải thuật di truyền

5.4.1 Ví dụ giải thuật di truyền với hàm một biến

Bài toán: Tìm giá trị lớn nhất của hàm $(15x\text{-}x^2)$ với x trong khoảng [0;15]. Chúng ta có thể giả định x chỉ nhận giá trị nguyên, do đó NST có thể được xây dựng với các gen:

Integer	Binary code	Integer	Binary code	Integer	Binary code
1	0001	6	0110	11	1011
2	0010	7	0111	12	1100
3	0011	8	1000	13	1101
4	0100	9	1001	14	1110
5	0101	10	1010	15	1111

Giả sử kích thước của quần thể NST$N = 6$. Theo các tài liệu thống kê dược, trung bình: xác suất lai ghép $p_c = 0,7$, và các đột biến $p_m = 0,001$. Hàm $f(x) = 15x\text{-}x^2$ của GA tạo quần thể

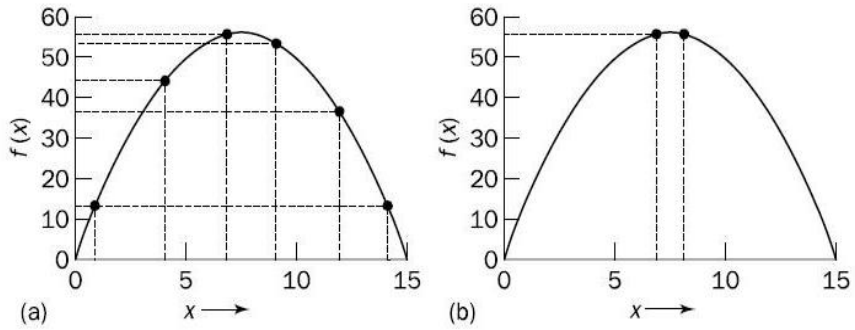
NST ban đầu bằng cách điền các chuỗi 4-bit với những giá trị ngẫu nhiên 1 và 0. Quần thể ban đầu như trong trên.

Một vấn đề khó khăn trong tính toán là một quần thể có hàng ngàn nhiễm sắc thể.

Bước tiếp theo là tính toán sự phù hợp của mỗi NSTriêng lẻ. Các kết quả cũng được thể hiện trong Bảng 5.2. Sự tương thích trung bình của quần thể ban đầu là 36. Để cải thiện nó, quần thể ban đầu được thay đổi bằng cách sử dụng lựa chọn, chéo và đột biến, toán tử di truyền. Trong chọn lọc tự nhiên, chỉ có các loài thích hợp nhất có thể sống sót, giống, và do đó truyền gen cho thế hệ tiếp theo. GAS sử dụng một cách tiếp cận tương tự, nhưng không giống như bản chất, quy mô quần thể NSTkhông thay đổi so một thế hệ kế tiếp.

Bảng 5.2. Bảng quần thể ngẫu nhiên ban đầu của nhiễm sắc thể

Chromosome label	Chromosome string	Decoded integer	Chromosome fitness	Fitness ratio, %
X1	1 1 0 0	12	36	16.5
X2	0 1 0 0	4	44	20.2
X3	0 0 0 1	1	14	6.4
X4	1 1 1 0	14	14	6.4
X5	0 1 1 1	7	56	25.7
X6	1 0 0 1	9	54	24.8



Hình 5.2. Hàm huấn luyện và phân bố nhiễm sắc thể

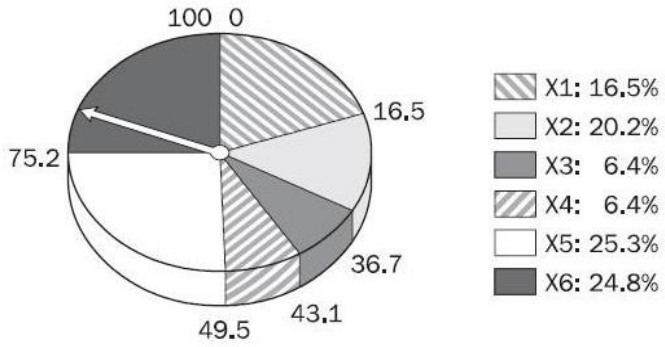
(a)Sự phân bố của nhiễm sắc thể ban đầu;(b) Sự phân bố của nhiễm sắc thể sau huấn luyện

Làm thế nào chúng ta có thể duy trì kích thước của các hằng số, và đồng thời cải thiện sự tương thích trung bình của nó?

Cột cuối cùng trong Bảng 5.2 cho thấy tỷ lệ tương thích NST của cá nhân với tổng thể của quần thể. Tỷ lệ này xác định NST được chọn để giao phối. Như vậy, các NST X_5 và X_6 có

cơ hội được chọn bằng nhau, trong khi NST X_3 và X_4 có xác suất được chọn rất thấp. Kết quả là, sự tương thích trung bình của NSTcải thiện từ một thế hệ tiếp theo.

Một trong những lựa chọn kỹ thuật thường được sử dụng NSTlà lựa chọn bánh xe roulette (Goldberg, 1989; Davis, 1991). *Hình 5.4.1* minh họa ví dụ của chúng tôi. Như bạn có thể thấy, mỗi NSTđược đưa ra một lát của một bánh xe tròn. Các khu vực của các slice trong các bánh xe bằng với tỷ lệ NST tương thích (xem Bảng 5.4.1). Ví dụ, các NST và X_5, X_6 (NST phù hợp nhất) chiếm diện tích lớn nhất, trong khi các NST X_3 và X_4 (phù hợp nhất) có phân đoạn nhỏ hơn nhiều trong bánh xe. Để chọn một NST cho lai, một số ngẫu nhiên được tạo ra trong khoảng [0; 100], và NST có đoạn kéo dài số ngẫu nhiên được chọn. Nó cũng giống như quay một bánh xe tròn nơi mỗi NST có một phân khúc trên các bánh xe tỷ lệ với sự tương thích của mình. Các bánh xe tròn được chia, và khi mũi tên đi kèm với phần còn lại trên một trong các phân đoạn, tương ứng NSTđược chọn.



Hình 5.3. Vòng tròn lựa chọn (Roulette Wheel Selection)

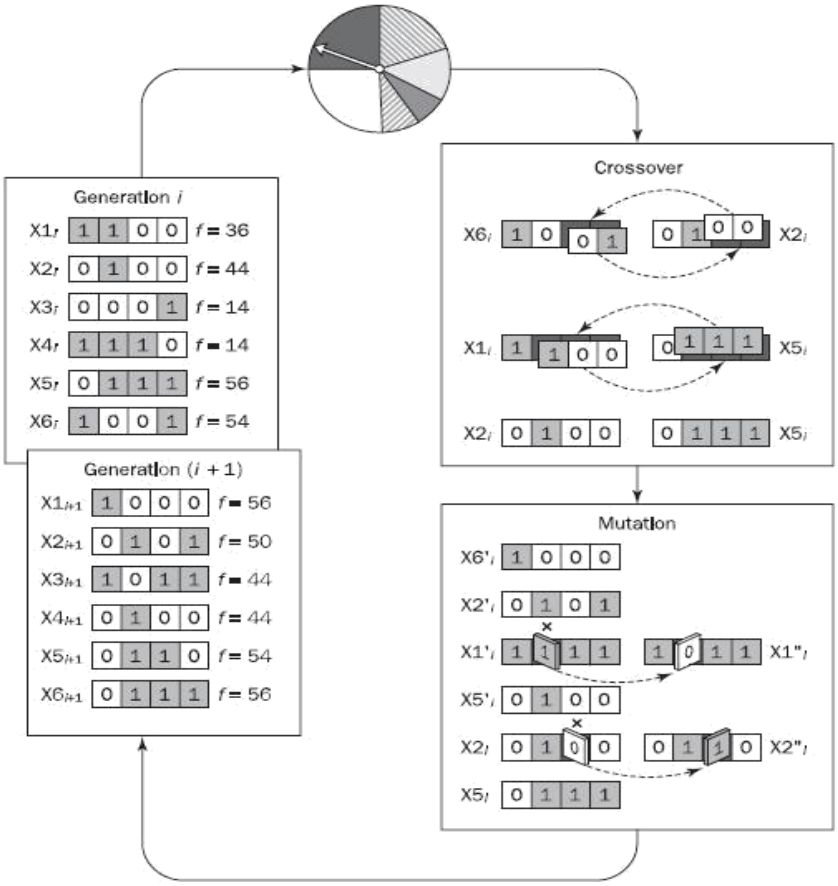
Trong ví dụ của chúng tôi, chúng tôi chọn một quần thể ban đầu của sáu nhiễm sắc thể. Vì vậy, để lập quần thể cùng trong thế hệ tiếp theo, các đường tròn sẽ được tách sáu lần. Hai lần đầu tiên có thể chọn NST X_6 và X_2 đến trở thành cha mẹ, cặp thứ hai của lần tiếp theo có thể chọn NST X_1 và X_5, và hai lượt cuối cùng có thể chọn NST X_2 và X_5.

Khi một cặp NST cha mẹ được chọn, các toán tử chéo được áp dụng.

Làm thế nào để lai (hay ghép chéo)?

Đầu tiên, các nhà ghép chéo chọn ngẫu nhiên một điểm giao nhau nơi hai NST cha mẹ khác nhau, và sau đó trao đổi các phần NST sau điểm đó. Kết quả là, hai đứa con mới được tạo ra. Ví dụ, các NST X_6 và X_2 có thể vượt qua sau khi các gen thứ hai trong mỗi để sản xuất hai con, như thể hiện trong *hình 5.3*.

Nếu một cặp NST không vượt qua, sau đó NST nhân bản, con được tạo ra như là bản sao chính xác của mỗi cặp bố mẹ. Ví dụ như, các NST mẹ X_2 và X_5 có thể không vượt qua. Thay vào đó, họ tạo ra thế hệ lai là bản sao chính xác của cặp NSY, như thể hiện trong *hình 5.3*.



Hình 5.4. Kết quả thế hệ lai các cặp NST được lựa chọn

Một giá trị của 0,7 cho xác suất chéo thường cho kết quả tốt. Sau khi lựa chọn , sư tương thích trung bình của quần thể NST đã được cải thiện và đi từ 36-42.

Đột biến đại diện cho những gì?

Đột biến, đó là sự kiện hiếm trong tự nhiên, đại diện cho một sự thay đổi trong gen. Nó có thể dẫn đến cải thiện đáng kể trong tương thích, nhưng thường có kết quả chứ không có hại. Vì vậy, tại sao sử dụng đột biến ở tất cả? Hà Lan giới thiệu đột biến như một nền điều hành (Hà Lan, 1975). Vai trò của nó là đảm bảo rằng các tìm kiếm Thuật toán tối ưu. Các chuỗi các lựa chọn và hoạt động chéo có thể trì trệ tại bất kỳ bộ đồng nhất của các giải pháp. Dưới điều kiện như vậy, tất cả các NST giống hệt nhau, và do đó các tập huấn luyện trung bình của dân số không thể được cải thiện. Tuy nhiên, các giải pháp có thể xuất hiện trở nên tối ưu, hay đúng hơn là tối

ưu cục bộ, chỉ vì các thuật toán tìm kiếm là không thể tiến hành thêm nữa. Đột biến là tương đương với một tìm kiếm ngẫu nhiên, và trợ chúng tôi trong việc tránh mất đa dạng di truyền.

Làm thế nào để công việc điều hành đột biến?

Lựa chọn ngẫu nhiên một nhiễm sắc thể. Ví dụ, các NST X_{10} có thể được đột biến ở gen thứ hai của mình, và các NST X_2 trong gen thứ ba của nó, như thể hiện trong hình 5.4.3 Đột biến có thể xảy ra bất kỳ gen trong NST với một số xác suất. Xác suất đột biến là khá nhỏ trong tự nhiên, và được lưu giữ khá thấp đối với khí, thường nằm trong khoảng giữa 0,001 và 0,01. Các thuật toán di truyền đảm bảo cải tiến liên tục của sự tương thích trung bình của quần thể, và sau một số thế hệ (thường là vài trăm) cá thể tiến hóa để một giải pháp gần tối ưu.

Trong ví dụ này, vấn đề chỉ có một biến. Nó rất dễ dàng để đại diện. Nhưng giả sử đó là mong muốn tìm ra tối đa của 'đỉnh' chức năng của hai biến.

5.4.2 Ví dụ giải thuật di truyền với hai biến

Bài toán: Tìm cực trị của hàm hai biến số thực *x, y,* trong khoảng: -3 và 3

$$f(x,y) = (1-x)^2 e^{-x^2-(y+1)^2} - (x - x^3 - y^3)e^{-x^2-y^2}$$

Bước đầu tiên là trong đó mỗi tham số được đại diện bởi tám bit nhị phân. Sau đó, chọn mô tả cá thể nhiễm sắc thể, ví dụ 6, và ngẫu nhiên tạo ra một quần thể ban đầu.

$(10001010)_2 = 1 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = (138)_{10}$

Và $(00111011)_2 = 0 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = (59)$

Bước tiếp theo, tính sự tương thích của mỗi nhiễm sắc thể. Điều này được thực hiện trong hai giai đoạn. Đầu tiên, một NST được giải mã bằng cách chuyển đổi nó thành hai số thực *x, y,* trong khoảng thời gian giữa -3 và 3; Sau đó, giá trị giải mã của *x* và *y* được thay ra ở 'đỉnh' chức năng: 6/(256-1)= 0,0235294

$$x = (138)_{10} \times 0.0235294 - 3 = 0.2470588$$

và

$$y = (59)_{10} \times 0.0235294 - 3 = -1.6117647$$

Khi cần thiết, chúng ta cũng có thể áp dụng các kỹ thuật giải mã khác, chẳng hạn như Gray (Caruana và Schaffer, 1988). Sử dụng giá trị giải mã của x và y như là đầu vào trong các chức năng toán học, GA tính toán sự tương thích của mỗi nhiễm sắc thể.

Để tìm tối đa của 'đỉnh' chức năng, chúng tôi sẽ sử dụng chéo với *xác suất tương đương 0,7 và đột biến với xác suất bằng 0,001.* Như chúng ta đã đề cập trước đó, một thực tế phổ biến trong khí là để xác định số thế hệ. Giả sử số mong muốn của các thế hệ là 100. Đó là, GAsẽ tạo ra 100 thế hệ 6 NST trước khi dừng lại.

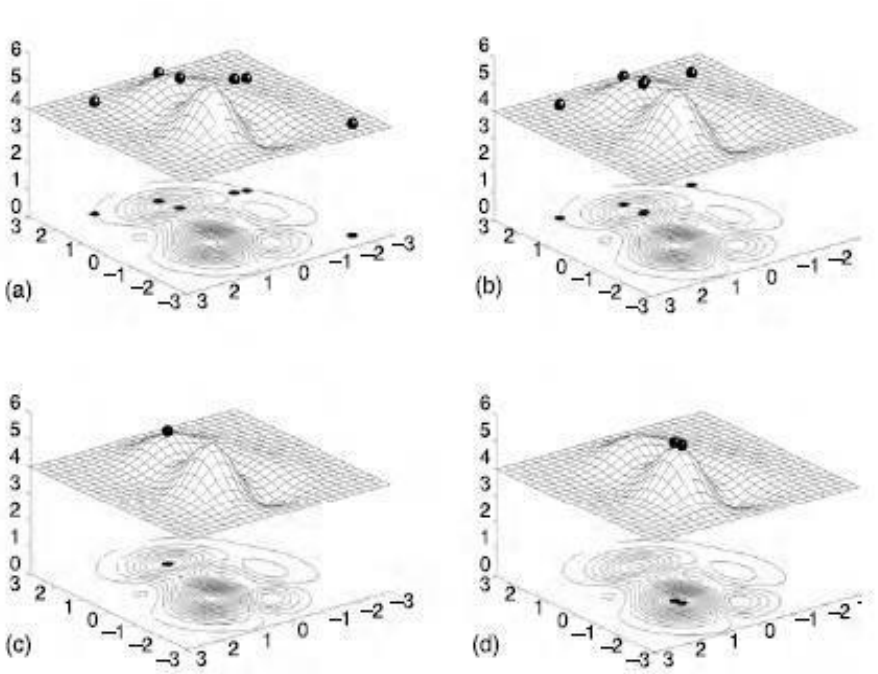


Figure 7.6 Chromosome locations on the surface and contour plot of the 'peak' function: (a) initial population; (b) first generation; (c) local maximum solution; (d) global maximum solution

Hình 5.5 (a) cho thấy các vị trí ban đầu của các NSTtrên bề mặt và lô đường viền của các 'đỉnh' chức năng. Mỗi NSTở đây được đại diện bởi một quả cầu. Quần thể ban đầu bao gồm các cá nhân được tạo ngẫu nhiên không giống nhau hoặc không đồng nhất. Tuy nhiên, bắt đầu từ thế hệ thứ hai, chéo bắt đầu tái kết hợp các tính năng của NST tốt nhất, và các cá thể bắt đầu hội tụ trên đỉnh chứa tối đa, như được hiển thị trong hình 5.5 (b). Từ đó cho đến thế hệ cuối cùng, GA được tìm kiếm xung quanh đỉnh cao này có đột biến, dẫn đến sự đa dạng. Hình 5.5 (c) cho thấy thế hệ nhiễm sắc thể. Tuy nhiên, quần thể đã hội tụ trên một NST nằm trên một vị trí tối đa của các 'đỉnh' chức năng.

Nhưng chúng ta đang tìm kiếm tối đa trên toàn tập, vì vậy chúng tôi có thể chắc chắn cho tìm kiếm các giải pháp tối ưu? Vấn đề nghiêm trọng nhất trong việc sử dụng GAS là có liên quan với chất lượng của các kết quả, đặc biệt là có hay không một giải pháp tối ưu được tìm thấy. Một cách để cung cấp một mức độ an toàn là để so sánh kết quả thu được theo tỷ lệ khác nhau của đột biến.Ví dụ, tăng tỷ lệ đột biến 0,01 và chạy lại GA, quần thể hiện nay có thể hội tụ trên các NST hiện trong hình 5.5(d). Tuy nhiên, để chắc chắn về ổn định kết quả chúng ta phải tăng kích thước của quần thể nhiễm sắc thể.

Một mặt khác một hàm toán học của các loại được đưa ra trong *Hình 5.5* là thuận tiện cho việc hiển thị hiệu suất. Tuy nhiên, tương thích chức năng cho các vấn đề thế giới thực không thể dễ dàng đại diện bởi hình ảnh. Thay vào đó, chúng ta có thể sử dụng đồ thị hiệu suất.

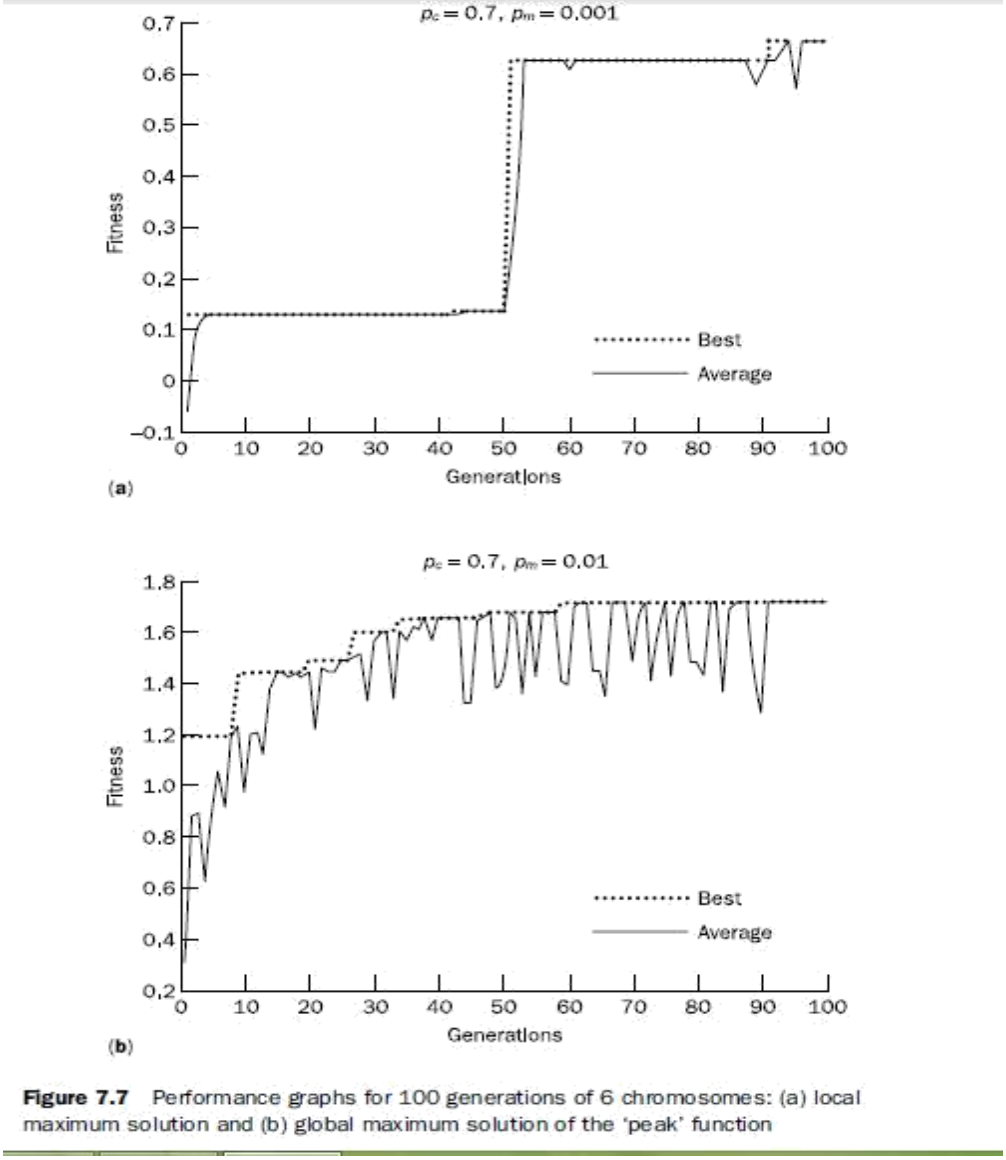


Figure 7.7 Performance graphs for 100 generations of 6 chromosomes: (a) local maximum solution and (b) global maximum solution of the 'peak' function

Một đồ thị hiệu suất là gì?

Kể từ khi các thuật toán di truyền là ngẫu nhiên, hiệu suất của chúng thường thay đổi từ thế hệ này sang thế hệ khác. Kết quả là, một đường cong cho thấy hiệu suất trung bình của toàn bộ quần thể NSTcũng như một đường cong hiển thị hiệu suất của các NST tốt nhất trong quần thể là một cách hữu hiệu để kiểm tra kết quả của một GA trong số lựa chọn của các thế hệ. Hình 5.6 (a) và (b) là hiển thị của các giá trị tốt nhất và trung bình tương thích hàm trên 100 thế hệ. Trục x của đồ thị hiệu suất chỉ số thế hệ đã được tạo ra và đánh giá tại các điểm cụ thể trong thời gian, và trục y hiển thị giá trị của hàm huấn luyện tại thời điểm đó.

Các vị trí thất thường của các đường cong hiệu suất trung bình là do đột biến.

Đột biến cho phép một GA tìm ra những vị trí khác biệt một cách ngẫu nhiên. Đột biến có thể dẫn đến sự cải thiện đáng kể trong quần thể tương thích, nhưng thường làm giảm nó. Để đảm bảo sự đa dạng và đồng thời để làm giảm tác hại của đột biến, chúng ta có thể làm tăng kích thước của quần thể nhiễm sắc thể. *Hình 5.6* cho thấy đồ thị hiệu suất cho 20 thế hệ 60 nhiễm sắc thể. Các đường cong tốt nhất và trung bình ở đây là tiêu biểu cho tập giá trị. Như bạn có thể thấy, các đường cong trung bình tăng lên nhanh chóng vào lúc ban đầu, nhưng sau đó khi dần hội tụ về các giải pháp tối ưu, nó tăng chậm hơn, và cuối cùng không thay đổi ở cuối.

CÂU HỎI VÀ BÀI TẬP

1. Thế nào là nhiễm sắc thể? Cách biểu diễn nhiễm sắc thể

2. Các toán tử sử dụng trong giải thuật di truyền

3. Trình bày thuật toán di truyền

4. Cho hàm hợp lý *(-x2 +15x)* với *x* trong khoảng *[0;15],* giả định *x* lấy giá trị nguyên.

   a) Xác định kích thước của nhiễm sắc thể với gen được mã hóa nhị phân *[0, 1];*

   b) Chỉ dùng toán tử lai ghép, tìm giá trị cực đại

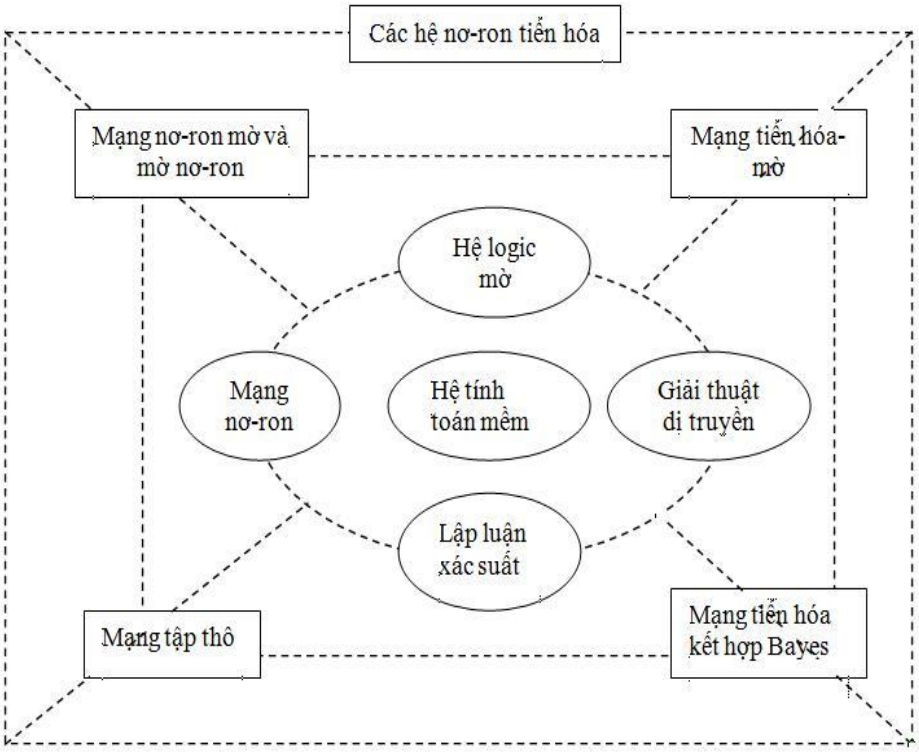
CHƯƠNG 6:

CÁC HỆ CƠ SỞ TRI THỨC LAI

Chương này giới thiệu một số hệ lai. Mỗi hệ cơ sở tri thức dựa luật đều có những ưu và nhược điểm riêng. Việc lai, kết hợp giữa các hệ tận dụng và hạn chế nhược điểm và phát huy điểm mạnh của hệ này cho hệ khác, tạo một hệ thống tích hợp khả dĩ hoàn chỉnh hơn.

6.1 Đặc tính của hệ tính toán mềm

6.1.1. Các khái niệm cơ bản về các hệ tính toán mềm



Hình 6.1: Thành phần của hệ tính toán mờ

Trong khi tính toán mềm có thể đưa lời giải, hay ước lượng từ các thông tin không đầy đủ, không chính xác, hoặc chỉ ước đoán mà tính toán cứng không giải được.

6.1.2. Các thành phần của hệ tính toán mềm

Hệ tính toán mềm gồm 4 hệ cơ bản: logic mờ, mạng nơ-ron, giải thuật di truyền lập luận xác suất; và các hệ lai của 4 hệ đó (Hình 6.1). Chúng ta đã nghiên cứu, tìm hiểu các hệ đó ở các chương trước có thể tổng hợp:

Hệ cơ sở	Ưu điểm
Logic mờ	Lập luận gần đúng và cảm nhận
Mạng nơ-ron	Học và biểu diễn tri thức ẩn
Giải thuật di truyền	Tiến hóa tự nhiên và tối ưu hóa
Lập luận xác suất	Tính không chắc chắn

Hệ thống phân chia này đã được đề xuất bởi McGarry và các đồng nghiệp của mình để phân loại các hệ lai thành 3 nhóm chính:

- Hệ lai thống nhất (Unified Hybrid Systems): Các hệ này xử lý bằng mạng nơ-ron

- Hệ thống lai truyền đạt (Transformational Hyrid Systems): trong hệ thống này cách mô tả bằng ký hiệu được chuyển vào mạng nơ-ron và ngược lại từ mạng nơ-ron chuyển ra.

- Hệ thống lai theo Modul (Modular Hyrid Systems): hệ thống lai này bao gồm các modul khác nhau, mỗi modul thực hiện một nhiệm vụ xác định sử dụng kỹ thuật thích hợp.

6.1.3. Các đặc trưng của hệ thống tính toán mềm

- *Mô phỏng của các chuyên gia*

Hệ tính toán mềm sử dụng logic mờ, trong đó cung cấp một cách tiếp cận linh hoạt để thực hiện với các thứ phân loại như con người vào nhóm có ranh giới của nó là không rõ ràng, với khái niệm biến ngôn ngữ mờ, chẳng hạn như xe hơi lớn, mùa nóng và người giàu. Suy diễn mờ cung cấp một lập luận xấp xỉ và có giải thích.

- *Kỹ thuật sáng tạo*

Hệ tính toán mềm cung cấp các kỹ thuật tiên tiến để tối ưu hóa, giải pháp tự tiến hóa, học máy, lý luận, và tìm kiếm từ các ngành khác nhau như giải thuật di truyền, mạng nơ-ron và logic mờ.

- *Tiến hóa tự nhiên*

Các thuật toán di truyền, khi lai trong hệ tính toán mềm, hỗ trợ trong các giải pháp tiến hóa tự nhiên. Một mạng nơ-ron nhân tạo cung cấp một phương tiện học tập tự họcbản thân, không có dữ liệu huấn luyện. Theo các này, hệ tính toán mềm cung cấp mô hình tính toán lấy cảm hứng từ sinh học cho nhận dạng mẫu, hồi quy phi tuyến, và tối ưu hóa.

- *Học theo mô hình tự do*

Trên tất cả, các ứng dụng mà không thể được giải quyết bằng một mô hình cụ thể có thể được giải quyết với một hệ lai tính toán mờ. Với sự giúp đỡ của giải thuật di truyền, từ ví dụ, các mô hình phù hợp để giải quyết vấn đề có thể tự phát triển từ đặc điểm của vấn đề. Tương tự, chỉ từ bộ dữ liệu giống nhau nhất định, mạng nơ-ron tính toán mềm có thể phát triển một mô hình có thể giải quyết một vấn đề với các dữ liệu thực tế tương tự.

- *Định hướng mục tiêu*

Mạng nơ-ron và giải thuật di truyền là mục tiêu đặt ra. Đó là, nó là giải pháp mà là quan trọng, không phải là con đường mạng / thuật toán sau. Tương tự như vậy, các hàm huấn luyện quyết định tính đúng đắn của giải pháp và quyết định sự tồn tại của các giải pháp như là một tiền đề trong các thế hệ tiếp theo.

- *Tính toán sâu rộng*

Hệ tính toán mềm dựa trên các thuật toán tính toán mở rộng được cung cấp bởi mạng nơ-ron, logic mờ và giải thuật di truyền, không giống như biểu tượng truyền thống về trí tuệ nhân tạo (AI). Điều này mở rộng phạm vi của hệ tính toán mềm ngoài các ứng dụng AI điển hình. Ví dụ trong đó tính toán số học phổ thông như vậy được yêu cầu bao gồm xử lý tín hiệu kiểm soát và hồi quy phi tuyến.

- *Xử lý thông tin không cân bằng và không đầy đủ*

Ngành như logic mờ và mạng nơ-ron nhân tạo đem đến cho hệ tính toán mềm khả năng giải quyết với những thông tin không đầy đủ, không chắc chắn và trừu tượng. Không giống như hệ thống truyền thống, các hệ tính toán mềm không có tài liệu cụ thể trong kiến thức cơ bản.

- *Tính chịu lỗi*

Hệ thống tính toán mềm sử dụng một mạng lưới nơ-ron nhân tao là một trong những thành phần của nó. Nơ-ron trong một kiến trúc mạng nơ-ron nhân tạo song song. Ngay cả khi một trong số đó không làm việc thì hệ thống sẽ không thất bại. Ví dụ trong một loạt lớn các đèn chiếu sang, ngay cả khi có một vài thành phần không làm việc, các mẫu đầy đủ có thể được nhìn thấy. Vì vậy, với các hệ thống logic mờ dựa trên: nếu một quy tắc bị xóa, hệ thống mờ vẫn làm việc. Như vậy hệ tính toán mềm là có tính chịu lỗi thật sự.

6.2 Hệ lai nơ ron mờ

6.2.1. Sự kết hợp giữa logic mờ và mạng nơ ron

1. Khái niệm

Khi khảo sát mạng nơ ron và logic mờ, ta thấy mỗi loại đều có điểm mạnh, điểm yếu riêng của nó.

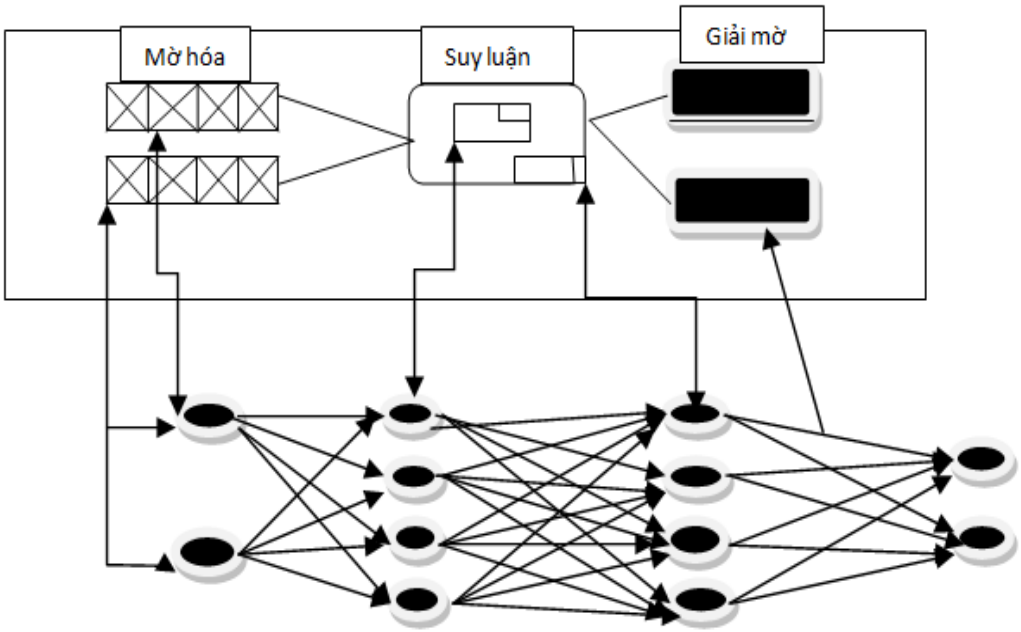
Đối với logic mờ, ta dễ dàng thiết kế một hệ thống mong muốn chỉ bằng các luật Nếu - thì (If-Then) gần với việc xử lý của con người. Với đa số ứng dụng thì điều này chophép tạo ra lời giải đơn giản hơn, trong khoảng thời gian ngắn hơn. Thêm nữa, ta dễ dàng sử dụng những hiểu biết của mình về đối tượng để tối ưu hệ thống một cách trực tiếp. Tuy nhiên, đi đôi với các ưu điểm hệ điều khiển mờ còn tồn tại một số khuyết như: việc thiết kế và tối ưu hóa hệ logic mờ cần phải có kinh nghiệm về điều khiển đối tượng. Mặt khác, còn hàng loạt những câu hỏi khác đặt ra cho người thiết kế mà nếu chỉ dừng lại ở tư duy logic mờ thì hầu như chưa có lời giải. Ví dụ: số tập mờ trong mỗi biến bao nhiêu thì tối ưu? hình dạng các tập mờ thế nào? đặt tập mờ ở đâu? kết hợp các tập mờ như thế nào? trọng số của mỗi luật bao nhiêu? tri thức được đưa vào huấn luyện nên ở dạng nào?, ….

Đối với mạng nơ ron, ưu điểm lớn nhất chính nằm ở việc xử lý song song khiến tốc độ xử lý rất nhanh. Mạng nơ ron có khả năng học hỏi. Ta có thể huấn luyện mạng để xấp xỉ một hàm phi tuyến bất kỳ, đặc biệt khi đã biết một tập dữ liệu vào/ra... Song, nhược điểm cơ bản của mạng nơ ron là khó giải thích rõ ràng hoạt động của mạng nơ ron như thế nào. Do vậy, việc chỉnh sửa trong mạng nơ ron rất khó khăn. Hai tiêu chí cơ bản trợ giúp cho người thiết kế ở logic mờ và ở mạng nơ ron thể hiện trái ngược nhau (Bảng 6.2).

Bảng 6.2: So sánh mạng nơ ron và logic mờ

	Mạng Neuron	Logic mờ
Thể hiện tri thức	Không tường minh,khó giải thích và khó sửa đổi.	Tường minh, dễ kiểm chứng hoạt động và dễ sửa đổi
Khả năng học	Có khả năng học thông qua các tập dữ liệu	Không có khả năng học,người thiết kế phải tự thiết kế tất cả.

Vì thế, nếu kết hợp logic mờ và mạng nơ ron, ta sẽ có một hệ lai với ưu điểm của cả haithiết kế dễ dàng, tường minh (của logic mờ) với việc học (của mạng nơ ron). Nó tự động sửa đổi các hàm phụ thuộc về hình dạng, vị trí và sự kết hợp… Điều này làm giảm bớt thời gian cũng như giảm bớt chi phí khi phát triển hệ (Hình 6.2).

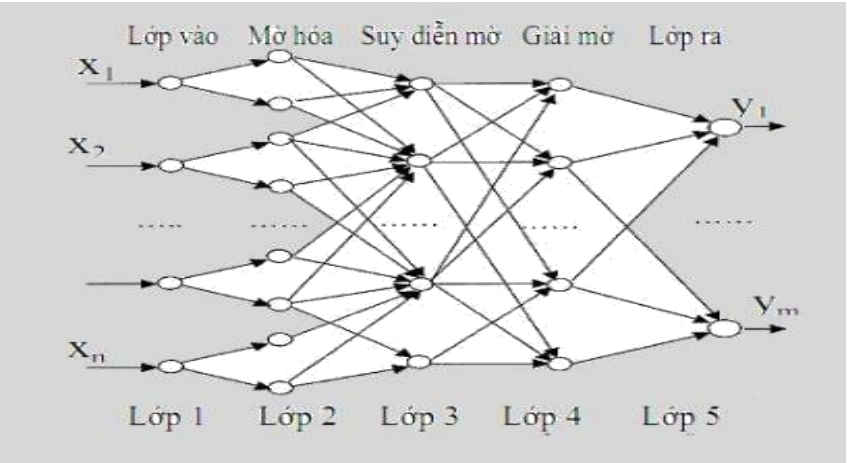


Hình 6.2 : Mô hình hệ mờ - nơ ron

2. Cấu trúc chung của hệ mờ - nơ ron

Có nhiều cách kết khác nhau để hợp mạng nơ ron với logic mờ. Cấu trúc chung của hệ Mờ-Nơ ron (Fuzzy-Neuro) như hình 6.2.

Sử dụng các nơ ron RBF mô tả dưới đây, sự mờ hoá có thể đạt được rất dễ dàng. Mỗi biến ngôn ngữ được xây dựng bằng 1 nơ ron. Chú ý rằng kiểu hàm của nơ ron không nhất thiết phải là hàm Gaus mà có thể là hàm khác. Trong phần này hàm liên thuộc kiểu tam giác có thể không được sử dụng vì chúng không trơn. Các nơ ron mờ hoá đóng vai trò lớp vào của mạng.



Hình 6.2 : Cấu trúc của hệ mờ-nơ ron

Tiếp theo, lớp ẩn là toán từ MIN. Đôi khi hàm này được thay bằng toán tử PROD. Đầu ra của nơ ron này là đầu vào của nơ ron tiếp theo.

Lớp thứ 3 được xây dựng bởi các nơ ron MAX (hoặc SUM). Lớp này tương tự lớp trước nhưng nó là tổng của các nơ ron đầu vào. Nếu đã biết luật, ta sẽ có mối liên hệ nơ ron PROD là tổng của các khối. Việc tính toán được định nghĩa ở ngay khi khởi tạo. Khi tối ưu mạng, giá trị của từng khối có thể là 1 hoặc 0 (hay hợp lệ hoặc không hợp lệ). Như vậy, các lớp luật sau mỗi được tính toán sẽ được thêm vào mạng. Cuối cùng, các nơ ron tổng được liên kết với nơ ron ban đầu tạo thành lớp. Khối này xác định một giá trị cứng bằng việc xây dựng tích của mỗi vị trí MAX của nơ ron với giá trị tương ứng của nó và phân chia tổng này theo vị trí nơ ron. Đây chính là phương pháp singleton để xác định giá trị rõ ở đầu ra. Mạng có tham số sau để thay đổi các đặc trưng của nó:

- Giá trị trung bình của mỗi hàm liên thuộc (vi là giá trị cực đại của nó).

- Chiều rộng của mỗi hàm liên thuộc.

- Tính hợp lệ (giá trị) của mỗi luật. Nhìn chung, giá trị của mỗi luật không nhất thiết phải là 1 hoặc 0, chủ yếu chúng nằm giữa 2 giá trị này. Nếu bằng 0 ta coi luật đó bị mất, bình thường ta coi một luật bằng 1 hoặc bằng 0 với một mức độ nhất định.

6.3 Biểu diễn luật If-Then theo cấu trúc mạng nơ ron

Xét hệ SISO, luật điều khiển có dạng:

$$Ri = Nếu\ x\ là\ Ai\ Thì\ y\ là\ Bi \qquad (6.1)$$

với Ai, Bi là các tập mờ, i = 1,..., n. Mỗi luật của (6.1) có thể chuyển thành một mẫu dữ liệu cho mạng nơ ron đa ất ng bằng cách lấy phần "Nếu" làm đầu vào và phần "Thì" làm đầu ra của mạng. Từ đó ta chuyển khối luật thành tập dữ liệu sau:

$$\{(A_1,B_1),...,(An,Bn)\}$$

Đối với hệ MISO, việc biểu diễn khối luật dưới dạng tập dữ liệu cũng tương ựt như đối với hệ SISO.

Ví dụ: Luật Ri :

$$Nếu\ x\ là\ Ai\ và\ y\ là\ Bi\ Thì\ z\ là\ Ci \qquad (6.2)$$

với *Ai, Bi, Ci* là các tập mờ, $i = 1,..., n$.

Tập dữ liệu của khối luật là:

$$\{(Ai, Bi), Ci\},\ \ 1 \le i \le n.$$

Còn đối với hệ MIMO thì khối luật Ri **:**

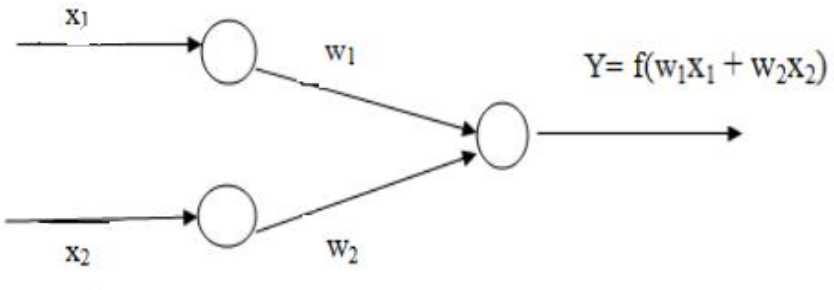
$$Nếu\ x\ là\ Ai\ và\ y\ là\ Bi\ Thì\ r\ là\ Ci\ và\ s\ là\ Di \qquad (6.3)$$

với *Ai, Bi, Ci, Di* là các tập mờ, $i = 1,..., n$.

Tập dữ liệu của khối luật là: $\{(Ai,Bi),(Ci,D)\},\ 1 \le i \le n$. Có hai cách để thực hiện luật "Nếu...Thì" (If...Then) dựa trên giải thuật lan truyền ngược sai lệch.

6.4 Nơron-mờ

Xét mạng nơ ron như hình 6.3. Trong đó: các tín hiệu vào-ra và các trọng số đều là số thực; Hai nơ ron ở đầu vào không làm thay đổi tín hiệu nên đầu ra của nó cũng là đầu vào.



Hình 6.3. Ví dụ hệ nơ ron-mờ

Tín hiệu xi kết hợp với trọng số wi tạo thành tích: $pi = w_i x_i$, $i = 1,2$. Đầu vào của nơ ron ở tầng ra là sự kết hợp của các *pi* theo phép cộng: $p_1 + p_2 = w_1x_1 + w_2x_2$. Nơ ron này dùng một hàm chuyển *f* để tạo đầu ra. Ví dụ hàm chuyển là hàm dạng chữ *S* đơn cực:

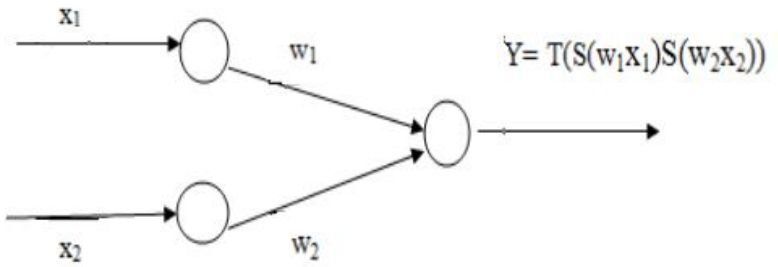
$$f(x) = \frac{1}{1+ e^{-x}} \tag{6.4}$$

$$y = f(w_1x_1 + w_2x_2), \quad f(x) = \frac{1}{1+ e^{-x}} \tag{6.5}$$

Mạng nơ ron dùng phép nhân, phép cộng và hàm dạng chữ S được gọi là mạng nơ ron chuẩn. Nếu mạng nơ ron dùng các phép toán khác như t-norm, t-conorm để kết hợp dữ liệu được gọi là mạng nơ ron lai. Mạng nơ ron lai là cơ sở để tạo ra cấu trúc nơ ron mờ dựa trên các phép toán mờ. Để có mạng nơ ron mờ ta thực hiện: Biểu diễn các đầu vào (thường là các độ phụ thuộc) x_1, x_2 và trọng số w_1, w_2 trên khoảng [0, 1]. - Mạng nơ ron lai có thể không dùng các phép toán nhân, phép toán cộng hoặc hàm dạng chữ S bởi vì kết quả của các phép toán này có thể không nằm trong khoảng [0, 1].

Định nghĩa: Mạng nơ ron lai là mạng nơ ron sử dụng tín hiệu rõ và hàm truyền rõ,song sự kết hợp x_1 và w_1 dùng các phép toán *t-norm, t-conorm* hay các phép toán liên tục khác và sự liên kết p_1 và p_2 dùng các hàm *t-norm, t-conorm* hay các hàm liên tục khác, hàm chuyển *f* có thể là một hàm liên tục bất kỳ.

Chú ý: Đối với mạng nơ ron mờ thì giá trị vào, giá trị ra, và trọng số là những sốthực nằm trong khoảng [0, 1].



Hình 6.4 : Nơ ron mờ AND

Tín hiệu *xi* và trọng số *wi* được kết hợp bởi conorm *S* tạo thành:

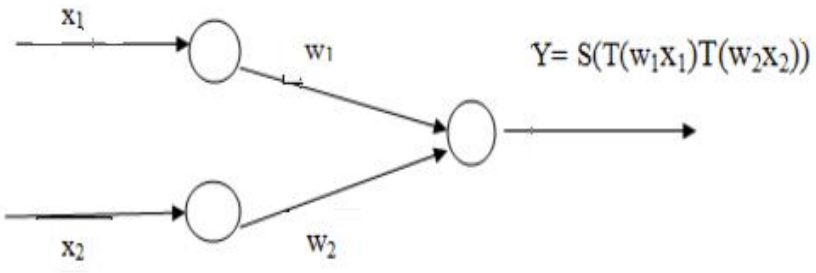
$$pi = S(wi,xi), \; i = 1,2 \tag{6.6}$$

Các pi được tính bởi *norm T* để tạo đầu ra của nơ ron.

$$y = AND(p_1,p_2) = T(p_1,p_2) = T(S(w_1,x_1),S(w_2,x_2)). \quad (6.7)$$

Nếu $T = min$ và $S = max$ thì nơ ron mờ AND chính là luật hợp thành

$$min\text{-}\ max\ y = min\{w_1 \vee x_1, w_2 \vee x_2\}. \quad (6.8)$$



Hình 6.5 : Nơ ron mờ OR

Tín hiệu *xi* và trọng số wi được kết hợp bởi *norm T* tạo thành :

$$pi = T(wi,xi),\ i = 1,2. \quad (6.9)$$

Các pi được tính bởi conorm S tạo đầu ra của nơ ron:

$$Y = OR(p_1,p_2) = S(p_1,p_2) = S(T(w_1,x_1),T(w_2,x_2)). \quad (6.10)$$

Nếu $T = min$ và $S = max$ thì nơ ron mờ *OR* chính là hợp thành:

$$max\text{-}min\ y = max\{w_1 \wedge x_1, w_2 \wedge x_2\}. \quad (6.11)$$

6.5 Huấn luyện mạng nơ ron mờ

Đối với mô hình mờ, mối quan hệ phi tuyến vào-ra phụ thuộc rất nhiều vào các phân vùng mờ của không gian vào-ra. Do đó việc chỉnh định hàm liên thuộc trong các mô hình mờ trở nên rất quan trọng. Trong mạng nơ ron mờ việc chỉnh định này có thể xem như là vấn đề tối ưu dùng giải thuật học để giải quyết. Đầu tiên ta giả định các hàm liên thuộc có một hình dạng nhất định. Sau đó ta thay đổi các thông số của hình dạng đó qua quá trình học bằng mạng nơ ron. Như vậy ta cần một tập dữ liệu ở dạng các cặp vào-ra mong muốn để cho mạng nơ ron học và cũng cần phải có một bảng các luật sơ khởi dựa trên các hàm phụ thuộc đó. Giả sử cần thực hiện ánh xạ:

$$y^k = f(x^k) = f(x_1^k, \dots, x_n^k), với\ i\ k = 1, \dots, K \quad (6.12)$$

Ta có tập dữ liệu: *{(x1,y1),...,(xk,yk)}.* Dùng luật If-Then để thực hiện ánh xạ này:

Ri : Nếu x_1 *là Ail và... và xn là Ain thì* $y = zi,\ 1 \le i \le m$ (6.13)

với Aif là các tập mờ có dạng hình tam giác và zi là số thực. Đặt ok là giá trị ra của hệ khi ta đưa vào xk. Ký hiệu 1 là giá trị ra của luật thứ i, được định nghĩa theo tích Larsen:

$$\alpha_i = \prod_{j=1}^{n} A_{ij}(x^k) \quad (6.14)$$

(cũng có thể định nghĩa các t-norm khác).

Giải mờ theo phương pháp trung bình trọng tâm ta có:

$$o^k = \frac{\sum_{i=1}^{n} \alpha_i Z_i}{\sum_{i=1}^{m} \alpha_i} \quad (6.15)$$

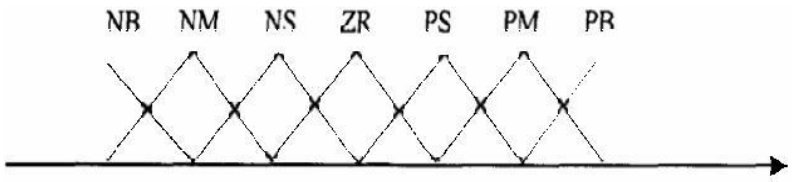
Sai lệch của mẫu thứ k là:

$$e^k = \frac{1}{2}(o^k - y^k)^2 \quad (6.16)$$

Dùng phương thức giảm để học zi trong phần kết quả của luật **Ri**:

$$z_{i(t/1)} = z_i(1) - \eta \frac{\partial E_k}{\partial Z} = z_i(1) - \eta (e^k - y^k) \frac{\alpha_i}{\alpha_{1+\dots+} \alpha_m}, \ i = 1, \dots, m \quad (6.17)$$

Cho rằng mỗi biến ngôn ngữ có 7 tập mờ như hình 6.6: {NB, NM, NS, ZE,PS, PM, PB}.



Hình 6.6 : Bảy vùng mờ

Các hàm liên thuộc có hình dạng tam giác được đặc trưng bởi 3 tham số: tâm, độ rộng trái, độ rộng phải. Các tham số này của tam giác cũng được học bằng phương thức giảm.

6.6 Phân loại kết hợp mạng nơ ron và logic mờ

Việc kết hợp giữa mạng nơ ron và logic mờ là cần thiết, có ba hướng kết hợp giữa mạng nơ ron và logic mờ theo từng chức năng xử lý riêng biệt

- *Neuro-Fuzzy Systems:* đưa kỹ thuật mạng nơ ron vào trong hệ thống suy diễn mờ.

- *Các mang mơ-nơron*: dùng logic mờ để mờ hóa các thông sô", hàm trong mạng nơ ron.

- *Fuzzy-Nơ ron Hybrid Systems:* kết hợp logic mờ và mạng nơ ron vào mô hình lai.

6.6.1. Neuro-Fuzzy Systems (NFS)

Nơ ron fuzzy systems hay *neuro-fuzzy systems* là một hệ suy diễn mờ đượ c tăng cường thêmkhả năng học của mạng nơ ron. Trong hệ thống này, mạng nơ ron được đưa vào làm tăng khả năng tự điều chỉnh các hệ sô" (biến) trong các luật mờ và hàm thành viên của hệ thông.

Với sự kết hợp này, khả năng học và suy diễn của hệ thống sẽ tốt hơn so với mạng nơ ron thông thường và tốc độ học cũng nhanh hơn. Các dạng NFS đã được giới thiệu: GARIC,

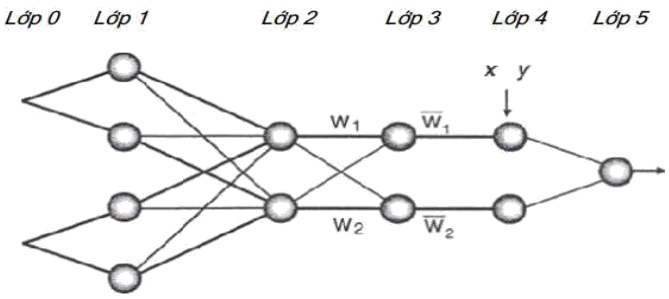
FALCON, ANFIS, NEFCON, FUN, SONFIN, FINEST, EFuNN, dmEFuNN...

Adaptive Neuro Fuzzy Inference System (ANFIS)

ANFIS là mô hình dựa trên dạng luật mờ do Sugeno đưa ra, đây là dạng NFS được sử dụng nhiều nhất. ANFIS là hệ thông suy diễn có khả năng tự điều chỉnh các luật cơ *sở* dựa trên khả năng học từ mẫu có sẩn của mạng nơ ron (mạng lan truyền ngược). Trong mô hình ANFIS, giả sử không mất tính tổng quát cho hai đầu vào *X* và *y*, một đầu ra z. Giả sử luật cơ sở được khởi tạo theo mô hình Sugeno như sau:

$$If\ x = A_1\ and\ y = B_1\ then\ f_1 = P_1x + q_1y + r_1 \qquad (6.18)$$

$$If\ x = A_2\ and\ y = B_2\ then\ f_2 = p_2x + q_2y + r_2 \qquad (6.19)$$



Hình 6.7 : Cấu trúc của một mạng NFS thường gặp

NFS có mô hình mạng nhiều lớp (1 lớp vào, *n* lớp ẩn, 1 lớp ra), sử dụng thuật toán lan truyền ngược cho quá trình học (luyện mạng). Mô hình mạng ANFIS thông thường bao gồm 5 lớp chính liên kết với nhau minh họa trong hình (sô' lớptrong mô hình ANFIS là cố định, tuy nhiên số' nút trong mạng sẽ tùy thuộc vào sô' luật khởi tạo ban đầu):

Lớp 0: chứa các nút nhập, các giá trị này truyền trực tiếp vào lớp 1.

Lớp 1: mỗi nút i trong lớp này được gắn với một hàm μ(x), μ (x) là một hàm thành viên),với X là đầu vào trực tiếp từ lớp 0. Giả sử chọn hàm Triangular, hàm μ(x) có dạng:

$$0_{1,i} = \mu(x) = \begin{cases} 0 & x \le a \\ \frac{x-a}{b-a} & a \le x \le b \\ \frac{c-x}{c-b} & b \le x \le c \\ 0 & c \le x \end{cases} \quad (6.20)$$

với *{a, b, c}* là tập hợp các tham số. Khi giá trị các tham *số* này thay đổi, thì hàm thành viên cũng có những thay đổi tương ứng. Các tham sô" này được xem như làcác tham sô" tiền đề trong mô hình ANFIS. Giá trị xuất của mỗi nút trong lớp này là giá trị mờ của nút nhập được tính thông qua hàm thành viên *μ(x).*

Lớp 2: tín hiệu vào của mỗi nút là giá trị ra của tất cả các nút ở lớp 1. Kết quả xuất ra ở nútnày là sự tích hợp của tất cả các gi
á trị vào.

$$0_{2,i} = w_i = \mu_{A_i}(x)\,\mu_{B_i}(y) \quad i = 1,2 \quad (6.21)$$

Mỗi nút xuất sẽ tính trị số của một luật (trọng sô" xuất ra), một cách tổng quát phép toán AND có thể được dùng như một hàm tính toán trên mỗi nút.

Lớp 3: mỗi nút trong lớp này nhận giá trị xuất của lớp trước và sau đó tính tỷ số như sau:

$$0_{3,i} = \overline{w_i} = \frac{w_i}{w_1 + w_2} \quad i = 1,2 \quad (6.22)$$

Khi đó mỗi luật sẽ có thêm một giá trị *Wj* sẽ sử dụng trong việc tính giá trị nút xuất ở lớp sau (giá trị này gọi là *normalized, firing trengths).*

Lớp 4: mỗi nút ở lớp này xem như là nút điều chỉnh với kết quả xuất ở nút cuối,

giá trị xuất ở mỗi nút có giá trị:

$$0_{4,i} = \overline{w_i} f_i = \overline{w_i}(px + qy + r) \quad (6.23)$$

tập *{p, q, r}* là tập các tham số trong lớp này. Các tham số' này được xem là tham số tổng hợp trong mô hình ANFIS. Hàm *f* tính theo mô hình Sugeno.

Lớp 5: ở lớp này chỉ có một nút tổng hợp kết quả xuất ra từ lớp trước, giá trị ra ở nút này làtổng các giá trị kết quả xuất ra từ lớp trước:

$$0_{4,i} = \sum \overline{w_i} f = \frac{\sum_i w_i f_i}{\sum_i w_i} \quad (6.24)$$

Cũng như thuật toán lan truyền ngược (Backpropagation) trong mạng nơ ron, mô hình ANFIS trong NFS cũng lặp lại quá trình tính toán tương tự lan truyền ngược. Quá trình lan truyền tiến sẽ tính giá trị xuất ra dựa trên các luật và các tập biến số *({a, b, c}, {p, q, r}* ở lớp 1, 4) được phát sinh ngẫu nhiên (hoặc được ấn định trước). Quá trình lan truyền ngược sẽ điều chỉnh các biến sô" sao cho sai sô" trung bình bình phương *E* là nhỏ nhất.

Quá trình học của ANFIS từ mẫu luyện

Khái niệm chính của việc phát sinh tự động luật mờ từ tập dữ liệu luyện *{Xi, Yi}i=l, 2, ...*

là dùng thuật toán vector lượng tử để tìm và cấp phát những vector lượng tử của dữ liệu luyện sang các lưới mờ trên không gian kết quả nhập xuất. Sau đó xác định trọng *số* của mỗi lưới mờ theo *số* vector lượng tử nằm trong lưới. Thuật toán vector lượng tử bao gồm thuật toán học theo Kohonen, kỹ thuật học luật. Cho *tl, t2*,…tm mô tả *m* vector được học từ không gian kết quả *XxY* (giá trị *m>r*). Trong trường hợp này, những lưới r.s hai chiều sẽ chứa ít n hất một vector lượng tử. Vector lượng tử tị phản ánh một luật mờ. Giả sử vector lượng tử *ki* phân bố trên lưới mờ thứ *i,* ta có:

$$k = k_1 + k_2 + \ldots + kr \quad (6.25)$$

Trọng số luật thứ i được tính:

$$w_i = \frac{k_i}{k} \quad (6.26)$$

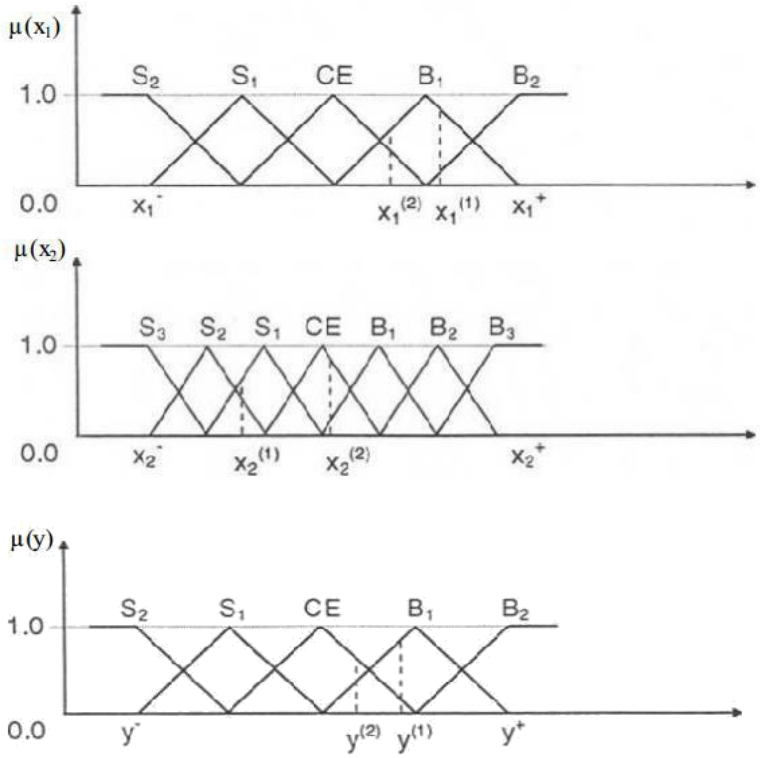
Trên thực tế, những luật được quan tâm là những luật có tần suất xuất hiện nhiều nhất.

Ví dụ: cho tập dữ liệu luyện *{X, Y},* giả sử cho tập ánh xạ mờ tương ứng NL (Negative

Large), NM (Negative Medium), NS (Negative Small), ZE (ZEro), PS (Positive Small), PM (Positive Medium), PL (Positive Large) trên không gian đầu vào *X*, đầu ra *Y (r = s = 7)*. Giả sử

sau thuật toán học, thu được 49 vector lượng tử *t1, t2, ...,t49* dữ liệu (vector) được phân trên 19 cụm. Trọng số của mỗi cụm có thể được xác định theo công thức 6.27. Ví dụ: luật (NS; PS) có trọng *số 8/ 49* =0.16, luật (NL, PL) có trọng *số* 2/ 49 = 0.04 và luật (PL, NL) có trọng số 1/ 49 = 0.02. Những ô còn lại trên lưới đều có giá trị bằng 0. Nếu chọn ngưỡng trọng số của luật là *wmin= 0.03* thì số luật trong không gian được phân cụm là 10 luật (chọn những cụm có trọng sô" lớn hơn ngưỡng). Mở rộng khái niệm trên cho tập luật *IF X is A AND y is B THEN z is c* hay *(A, B, C).* Khi đó, lưới luật 3 chiều và công thức 6.27 cũng được áp dụng tương tự. Rõ ràng theo luật mỗi tập A chỉ được ánh xạ vào một tập B duy nhất, vì vậy trên một cột của lưới chỉ xác định được duy nhất một luật. Nếu trên một cột có nhiều hơn một cụm thì cụm có trọng sô" lớn hơn sẽ được chọn. Theo hình (b) có các luật (NL; PL), (NM; PM), (NS; PS), (ZE; PM), (PS; PS), (PM; NM), (PL;NL).

+ *Rút luật mờ dựa trên việc chọn trực tiếp*



Hình 6.8. Các khoảng chia tương ứng với các giá trị vào-ra

So với kỹ thuật trên kỹ thuật này là một kỹ thuật lựa chọn một cách trực tiếp, vì vậy sẽ đơn giản và trực quan hơn; được giới thiệu vào năm 1992 bởi Wang và Mendel. Giả sử có dữ liệu luyện: *(x1, x2; y)(1), (x1, x2; y)(2), ... với x1, x2* là đầu vào, y đầu ra. Quá trình xác định tập luật mờ từ

các cập dữ liệu học và dùng các luật này xây dựng một ánh xạ *f: (x1, x2)—>y* tiến hành qua 3 bước:

Bước 1: *chia không gian nhập và xuất thành những vùng mờ.* Giả sử có các khoảng của x2vày như sau *[x1-,x1+], [x2-, x2+] và [y-, y+],* chia những khoảng này thành *N* vùng mờ, mỗi vùng được xác định bằng một hàm thành viên. Ví dụ: *X*| được chia ra 5 vùng mờ, *X2* được chia ra 7 vùng mờ, *y* được chia ra 5 vùng mờ.

Bước 2: *xây dựng luật mờ từ dữ liệu luyện.*

Đầu tiên, xác định giá trị mờ của các giá trị x_1, x_2 và *y* trong dữ liệu luyện trong các khoảng chia tương ứng dùng hàm thành viên. Ví dụ: $x_1(1)$ có giá trị 0.8 với *B1* và 0,2 với *B2* và bằng 0 trên tất cả các khoảng khác. Tiếp theo, chọn những khoảng tương ứng với giá trị lớn nhất. Ví dụ: B_1 với $x_1(1)$, CE với $x_2(2)$. Cuối cùng là rút ra luật từ các giá trị có được từ trên.

Bước 3: xác định trọng số mỗi luật.

Để giải quyết mâu thuẩn giữa các luật có thể xảy ra (cùng mệnh đề IF nhưng khác giá trị trong mệnh đề THEN), cũng như làm giảm số luật ; chọn những luật nào có trọng sô cao nhất trong tập luật mâu thuẩn. Cách tính trọng *số* như sau:

$$D(rule) = \mu_A(x_1)\mu_B(x_2)\mu_C(y) \qquad (6.27)$$

Hai kỹ thuật rút luật từ mẫu luyện trên được sử dụng thông dụng trong việc cài đặt một hệ neuro-fuzzy. Việc học từ mẫu luvện được tiến hành nhiều lần cho đến khi đạt được sai số trung bình bình phương mong muốn.

6.6.2. Mạng mờ- nơ ron

Fuzzy-nơ ron Network là mô hình mạng nơ ron, trong mô hình này, các thông số và thành phần của mạng nơ ron được mờ hóa trong quá trình tính toán (luyện mạng). Các thông số thành phần của mạng nơ ron có thể là các giá trị đầu vào, trọng số hay giá trị đầu ra. Việc kết hợp này làm cho hệ thống trở nên linh hoạt hơn và tăng khả năng dự đoán của hệ thống. Thông thường, mạng nơ ron được dùng trong mô hình này là mô hình mạng lan truyền ngược. Trong mô hình mạng mơ- nơ ron , kỹ thuật làm mờ các thông số và thành phần mạng nơ ron dựa trên mô hình Sugeno trong logic mờ. Mạng mơ-nơ ron *loại 1* thường được dùng trong các bài toán phân lớp, phân lớp các giá trị nhập mờ vào trong các lớp rõ. *Loại 2, 3 và.4* dùng cho các dạng sau:

Bảng 6.3. Các dạng mạng mờ- nơ ron

Mạng mơ-nơ ron	Giá trị nhập	Trọng số	Giá trị xuất
Loại 1	mờ	Rõ	rõ
Loại 2	mờ	Rõ	mờ
Lo ại 3	mờ	mờ	mờ
Loại 4	rõ	mờ	mờ
Loại 5	rõ	Rõ	mờ
Loại 6	rõ	mờ	rõ
Loại 7	mờ	mờ	rõ

Mạng mơ-nơ ron *loại 1* thường được dùng trong các bài toán phân lớp, phân lớp các giá trị nhập mờ vào trong các lớp rõ. *Loại 2, 3 và. 4* dùng cho các dạng tập luật mờ IF-THEN. Các loại còn lại có ý nghĩa trên lý thuyết nhưng không có ý nghĩa thực tế. Với sự phân loại như vậy, một mạng mơ-nơ ron thông thường có các giá trị đầu vào và trọng số được là mờ; hàm truyền được sử dụng là hàm sigmoid. Kiến trúc mạng nơ ron trong mô hình này tương tự mạng lan truyền ngược: 1 lớp vào *n* lớp ẩn 1 lớp ra để đơn giản.

Nhận xét: Mặc dù có sự phân biệt trong kỹ thuật liên kết các mạng nơ ron-mờ và các mangmờ-nơron, nhưng nhìn chung đó vẫn là sự kết hợp hỗ trợ lẫn nhau giữa hai kỹ thuật mạng nơ ron và logic mờ. Nhờ sự kết hợp như vậy, hệ thông nơ ron-mờ giải quyết được nhiều bài toán thực tế hơn so với khi tách riêng từng kỹ thuật. Một số sản phẩm trên thực tế đã sử dụng hai kỹ thuật này như để điều khiển máy điều hoà, máy photo, lò viba, máy giặt...

6.7 Hệ lai tiến hóa mờ

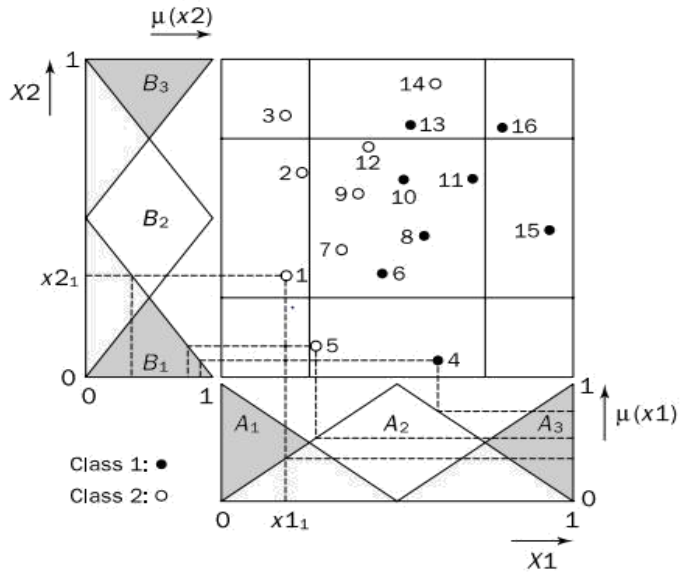
Tính toán tiến hóa cũng được sử dụng trong việc thiết kế các hệ thống mờ, đặc biệt để tạo ra các luật mờ và điều chỉnh chức năng thành viên của tập mờ.

Trong phần này, chúng tôi giới thiệu một ứng dụng của thuật toán di truyền để chọn một tập luật mờ IF-THEN cho một vấn đề phân loại (Ishibuchiet al., 1995).

Để áp dụng các thuật toán di truyền, chúng ta cần phải có một quần thể có tính khả thi

Với bài toán phân loại, một bộ luật mờ IF-THEN có thể được tạo ra từ dữ liệu số. Đầu tiên, sử dụng một phân vùng mờ của một không gian đầu vào.

Hình 6.6 cho ví dụ về các phân vùng mờ của một không gian đầu vào 3x3 không gian con mờ. Chấm đen và trắng ở đây biểu thị mô hình huấn luyện của lớp 1 và lớp 2, tương ứng.



Hình 6.9 : Phân vùng mờ bởi lưới mờ 3 × 3

Hình 6.6 cho biết các chu kỳ tiến hóa của một cấu trúc mạng nơ ron phân vùng có thể được xem như một bảng luật. Các giá trị ngôn ngữ đầu vào x_1 ($_1$, $_2$ và $_3$) tạo các trục ngang, các giá trị ngôn ngữ đầu vào x_2 ($_1$, $_2$ và $_3$) tạo trục dọc. Giao của hàng và cột cho kết quả luật.

Trong bảng luật, mỗi không gian con mờ có thể chỉ có một quy luật mờ IF- THEN, do đó tổng số các luật có thể được tạo ra trong một lưới là $K \times K$. Luật mờ tương ứng với K phân vùng mờ K có thể đại diện trong một dạng chung là:

Luật :

IF x_{1p} *is* A_i $i = 1, 2, \ldots, k$

AND x_{2p} *is* B_j $j = 1, 2, \ldots, k$

THEN $x_p \epsilon C_n$ $\{CF_{AiBj}^{Cn}\}$ $x_p = (x_{1p}, x_{2p}), p = 1, 2, \ldots, p;$

Trong đó, K là số khoảng mờ trong mỗi trục, là một mô hình huấn luyện trên đầu vào không gian $_1 \times _2$, trong đó P là tổng số của mô hình đào tạo, là hệ quả của luật (trong ví dụ, là một trong hai loại 1 hoặc loại 2), và là độ chắc chắn hay khả năng một mô hình trong không gian con mờ thuộc về lớp .

Để xác định hệ quả của luật và độ chắc chắn, có thể sử dụng thủ tục sau đây:

Bước 1:Phân hoạch không gian đầu vào *K*×*K* mờ, và tính độ mạnh của từng mô hìnhhuấn luyện trong mọi không gian con mờ. Mỗi lớp huấn luyện trong một không gian con mờ nhất định được đại diện bằng mô hình huấn luyện.

Hình 6.6 là một phân vùng mờ bởi lưới mờ 3x3 từ, trong một không gian con mờ, các luật xác định khi mô hình lớp huấn luyện đặc biệt xuất hiện thường xuyên hơn mô hình của lớp khác. Độ mạnh của lớp *Cn* trong không gian con mờ có thể được xác định:

$$\beta_{\mathrm{AiBj}}^{\mathrm{Cn}} = \sum_{p=1,\mathrm{xp}\,\epsilon\,\mathrm{Cn}}^{p}(\mu\mathrm{A_i}(\mathrm{x1_p}) \times \mu\mathrm{B_j}(\mathrm{x2_p})\,)\ \mathrm{x_p} = (\mathrm{x1_p}, \mathrm{x2_p}) \qquad (6.28)$$

trong đó, $\mu A_i(x_{1p})$ và $\mu B_i(x_{2p})$ là hàm thành viên của mô hình xp trong tập mờ A_i và tập $\mathrm{B_j}$, tương ứng. Trong hình 6.6, ví dụ, những thế mạnh của loại 1 và loại 2 trong không gian con mờ A_2B_1 được tính như sau:

$\beta A_2B_1\ Class1$=$\mu A_2(x_4) \times \mu B_1(x_4)$+$\mu A_2(x_6) \times \mu B_1(x_6)$+$\mu A_2(x_8) \times \mu B_1(x_8)$+$\mu A_2(x_{15})$ ×

$\beta A_2B_1\ Class1$=$\mu A_2(x_4) \times \mu B_1(x_4)$+$\mu A_2(x_6) \times \mu B_1(x_6)$+$\mu A_2(x_8) \times \mu B_1(x_8)$+$\mu A_2(x_{15}) \times \mu B_1(x_{15})$= 0.75 × 0.89 + 0.92 × 0.34 + 0.87 × 0.12 + 0.11 × 0.09 + 0.75 × 0.89 = 1.09

$\beta A_2B_1\ Class2 = \mu A_2(x_1) \times \mu B_1(x_1) + \mu A_2(x_5) \times \mu B_1(x_5) + \mu A_2(x_7) \times \mu B_1(x_7)$ = 0.42 × 0.38+ 0.54 × 0.81 + 0.65 × 0.21 = 0.73

Bước 2:Xác định các hậu quả luật và các yếu tố chắc chắn trong mỗi không gian mờcon. Khi kết quả của luật được xác định bởi các lớp mạnh, cần tìm lớp *Cm*

$$\beta_{\mathrm{AiBj}}^{\mathrm{Cm}} = max\left[\beta_{\mathrm{AiBj}}^{\mathrm{C1}}\ , \beta_{\mathrm{AiBj}}^{\mathrm{Cm}}\ , \ldots\ldots\ , \beta_{\mathrm{AiBj}}^{\mathrm{CN}}\right] \qquad (6.29)$$

Nếu một lớp được huấn luyện đặc biệt có giá trị tối đa, các hậu quả luật được xác định là . Ví dụ, trong không gian con mờ 2 1, các luật hậu quả là lớp 1.

Sau đó, các yếu tố chắc chắn có thể được tính:

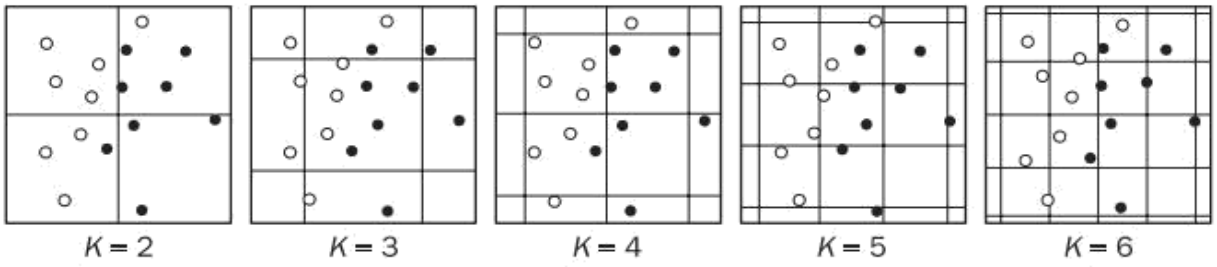
$$\mathrm{CF}_{\mathrm{AiBj}}^{\mathrm{Cm}} = \frac{\beta_{\mathrm{AiBj}}^{\mathrm{Cm}} - \beta_{\mathrm{AiBj}}}{\sum_{n=1}^{N} \beta_{\mathrm{AiBj}}^{\mathrm{Cn}}} \qquad (6.30)$$

Với :

$$\beta_{\mathrm{AiBj}} = \frac{\sum_{\substack{\mathrm{n=1} \\ \mathrm{n\neq m}}}^{\mathrm{N}} \beta_{\mathrm{AiBj}}^{\mathrm{Cn}}}{\mathrm{N}-2}$$

Ví dụ, các độ chắc chắn (Certainty Factor) của các luật ứng với không gian con mờ $_{2\,1}$ có thể được tính như sau:

$$\mathrm{CF}_{\mathrm{A2B1}}^{\mathrm{Class2}} = \frac{1.09 - 0.73}{1.09 + 0.73} = 0.20$$



Hình 6.10 : Bảng các luật mờ

Làm thế nào để giải thích các yếu tố chắc chắn ở đây?

Nếu tất cả các mô hình huấn luyện trong không gian con mờ thuộc về cùng một lớp *C*, sau đó các yếu tố chắc chắn đạt tối đa; chắc chắn mô hình mới trong không gian con sẽ thuộc về lớp . Tuy nhiên, nếu mô hình huấn luyện thuộc lớp khác nhau và các lớp này có thế mạnh tương tự, sau đó độ chắc chắn đạt tối thiểu và không đảm bảo mô hình mới sẽ thuộc về lớp . Điều này có nghĩa là các mô hình trong không gian con mờ $_{2\,1}$ dễ bị phân loại nhầm. Hơn nữa, nếu một không gian con mờ không có bất kỳ mô hình đào tạo, chúng ta không thể xác định luật nào. Trong thực tế, nếu một phân vùng mờ là quá thô, nhiều mô hình có thể được phân loại sai. Mặt khác, nếu một phân vùng mờ quá tốt, nhiều luật mờ không thể có vì thiếu họcmẫu trong không gian con mờ tương ứng. Như vậy, chọn mật độ mạng mờ là quan trọng cho phân loại mô hình đầu vào. Trong khi đó, như hình 6.10, mẫu huấn luyện không cần phân bố đều trong không gian đầu vào. Kết quả, khó chọn mật độ thích hợp cho lưới mờ. Để khắc phục những khó khăn này, sử dụng nhiều luật mờ. Một ví dụ được cho trong hình 6.10. Số lượng của các bảng phụ thuộc vào độ phức tạp của phân loại.

Luật mờ IF-THEN được tạo ra cho mỗi không gian con mờ của nhiều tập mờ loại trừ các bảng, do đó một bộ các luật hoàn chỉnh có thể được quy định như:

$$\mathrm{S_{ALL}} = \sum_{\mathrm{K=2}}^{\mathrm{L}} \mathrm{S_K}\,, \qquad \text{với} \quad \mathrm{K} = 2,3,\ldots\ldots,\mathrm{L} \tag{6.31}$$

trong đó, là tập hợp luật tương ứng với bảng luật mờ *K*. Các bộ luật tạo ra nhiều bảng luật mờ. Hình 6.10 chứa:

$$2^2 + 3^3 + 4^4 + 5^5 = 90 \text{ luật.}$$

Khi bộ luậtS_{ALL}được tạo ra, một mô hình mới, có thể phân loại theo các thủ tục sau đây:

Bước 1: Trongmỗi không gian con mờ,tính độ tương thích của mô hình cho từng lớp:

$$\alpha^{Cn}_{K\{A_iB_j\}} = \mu K\{A_i\}^{(x1)} \text{ x } \mu K\{B_j\}^{(x2)} \text{ x } CF^{Cn}_{K\{A_iB_j\}} \quad (6.32)$$
$$n = 1,2,\ldots..,N; \quad K = 2,3,\ldots..,L; \quad i = 1,2,\ldots.K; \quad j = 1,2,\ldots..,K$$

Bước 2:Xác định độ tương thích tối đa của mô hình mới cho từng lớp:

$$\alpha^{Cn} = \max\left[\alpha^{Cn}_{1\{A_iB_1\}}, \alpha^{Cn}_{K\{A_iB_2\}}, \alpha^{Cn}_{K\{A_2B_1\}}, \alpha^{Cn}_{K\{A_2B_2\}},\right.$$

$\alpha Cn = max\,[\alpha_1\{A_1B_1\}\,C_n, \alpha_1\{A_1B_2\}\,C_n, \alpha_1\{A_2B_1\}\,C_n, \alpha_1\{A_2B_2\}\,C_n,$

$\alpha_2\{A_1B_1\}\,C_n,\ldots,\alpha_2\{A_1B_K\}\,C_n, \alpha_2\{A_2B_1\}\,C_n,\ldots,\alpha_2\{A_2B_K\}\,C_n,\ldots,\alpha_2\{A_KB_1\}\,C_n,\ldots,\alpha_2\{A_kB_k\}\,C_n$ $,\ldots, \alpha_L\{A_1B_1\}\,C_n,\ldots,\alpha_L\{A_1B_K\}\,C_n,\alpha_L\{A_2B_1\}\,C_n,\ldots, \alpha_L\{A_2B_K\}\,C_n,\ldots, \alpha_L\{A_KB_1\}\,C_n,\alpha_1\{A_KB_K\}$ $C_n\,] ;\ N=1, 2,\ldots, N$

Bước 3: Xác định lớp C mà các mô hình mới có độ tính tương thích cao nhất, đó là:

$$\alpha^{Cm} = \max\left[\alpha^{C1}, \alpha^{C2}, \ldots, \alpha^{CN},\right.$$
$$\text{Assign pattern } x = (x1, x2) \text{ to class } Cm$$

Số lượng bảng luật mờ cần cho một mô hình phân loại có thể khá lớn; do đó, một bộ luật hoàn chỉnh là rất lớn. Mặt khác, các luật trong có khả năng phân loại khác nhau, do đó chỉ chọn các luật với tiềm năng cao để phân loại nhằm giảm kích thước các bộ luật. Vấn đề chọn luật mờ IF-THEN có thể xem như tổ hợp bài toán tối ưu với hai mục tiêu. Mục tiêu đầu tiên: tối đa hóa số lượng mô hình phân loại; thứ hai giảm thiểu số lượng các luật.

Trong các thuật toán di truyền, mỗi giải pháp được coi là một cá thể; do đó giải pháp cần đại diện cho một tập hợp có tính khả thi của luật IF- THEN như là một NST có chiều dài cố định. Mỗi gen trong một NST như vậy đại diện cho một luật mờ trong *SAll*

$$SAll = 2^2+3^3+4^4+5^5+6^6$$

Mục tiêu của chúng ta là thiết lập một tập luật mờ *S* bằng cách chọn luật thích hợp từ

bộ luật . Nếu một luật đặc biệt thuộc về *S*, bit tương ứng trong NST thừa nhận giá trị 1, nếu nó không thuộc về *S*, bit giả định giá trị -1. Luật Dummy được đại diện bởi số không.

Luật giả

Một luật giả được tạo ra khi các kết quả của luật này không xác định. Điều này xảy ra khi một không gian con mờ không có mô hình huấn luyện. Luật giả không ảnh hưởng đến hiệu suất của hệ phân loại, do đó có thể được loại trừ khỏi luật *S*.

Làm thế nào để quyết định luật mờ thuộc về cai trị đặt S?

Trong quần thể khởi tạo, quyết định này dựa trên 50% cơ hội. Nói cách khác, mỗi luật mờ có xác suất bằng 0,5 và nhận giá trị 1 trong mỗi nhiễm sắc thể, đại diện trong quần thể khởi tạo. Một thuật toán di truyền cơ bản để chọn luật IF-THEN gồm các bước:

Bước 1: Tạo ngẫu nhiên một quần thể gồm nhiều nhiễm sắc thể.Quy mô quần thể cóthể tương đối nhỏ, ví dụ 10 hoặc 20 nhiễm sắc thể. Mỗi gen trong một NST tương ứng với một luật mờ IF-THEN thiết lập bởi . Các gen tương ứng với luật giả nhận giá trị 0, tất cả các gen khác được phân chia ngẫu nhiên hoặc là 1 hoặc -1.

Bước 2: Tính toán hiệu suất của mỗi NST trong quần thể hiện tại.Vấn đề chọn luật mờcó hai mục tiêu: để tối đa hóa tính chính xác việc phân loại mô hình và để giảm thiểu kích thước của một luật. Điều này có thể đạt được bằng cách huấn luyện khi cho hai trọng số tương ứng, W_P và W_N trong các chức năng huấn luyện:

$$\mathrm{f(S)} = Wp\frac{\mathrm{Ps}}{\mathrm{P_{ALL}}} - W_N\frac{\mathrm{Ns}}{\mathrm{N_{ALL}}} \qquad (6.33)$$

trong đó -P_s là số mẫu được phân loại thành công,

-P_{ALL} là tổng số mẫu được mô tả cho hệ phân loại, N_s và

- N_{ALL} là số lượng luật mờ IF-THEN trong tập *S* và S_{ALL} tương ứng.

Độ chính xác phân loại quan trọng hơn so với kích thước của một luật. Điều này được thể hiện bằng cách gán các trọng số: $0 < WN \leq WP$

Giá trị tiêu biểu cho W_N và$W_P$là 1 và 10, tương ứng. Như vậy, ta có:

$$f(S) = 10\frac{Ps}{P_{ALL}} - \frac{Ns}{N_{ALL}} \qquad (6.34)$$

Bước 3: Chọn một cặp NST cho lai ghép. NST mẹ được chọn với một xác suất kết hợp với việc huấn luyện phù hợp

Bước 4: Tạo một cặp NST con bằng cách áp dụng toán tử lai chéo tiêu chuẩn.NST mẹ bị bỏ qua vào ngẫu nhiên điểm giao nhau được lựa chọn.

Bước 5: Thực hiện các đột biến trên mỗi gen của con.Các đột biến có xác suất khoảng 0.01.Các đột biến được thực hiện bằng cách nhân giá trị gen với -1.

Bước 6: Đặt cácNSTcon được tạo ra trong quần thể mới.

Bước 7: Lặp lại bước 3 cho đến khi kích thước của quần thể mới bằng cácquymô dânsố ban đầu, sau đó thay thế quần thể ban đầu với các quần thể mới.

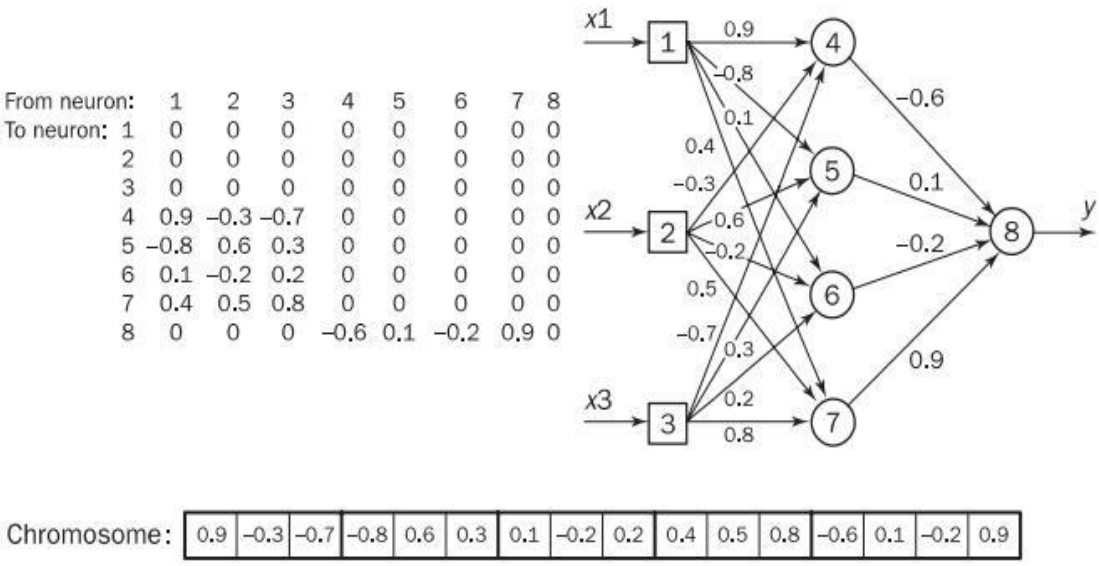
Bước 8: Đến bước 2, và lặp lại quá trình này cho đến khi một số quy định của thế hệ(thường là vài trăm) được xem xét.

Các thuật toán trên có thể làm giảm đáng kể số lượng luật mờ IF-THEN để phân loại chính xác. Trong thực tế, một số mô phỏng cho thấy số lượng các luật có thể được giảm xuống ít hơn 2 phần trăm của các bộ ban đầu tạo ra các luật. Một lá cắt giảm như vậy một hệ thống phân loại tương đối mờ với vài luật quan trọng, có thể sau được kiểm tra kỹ lưỡng bởi các chuyên gia của con người. Điều này cho phép sử dụng hệ thống tiến hóa mờ như một công cụ thu thập tri thức cho khám phá tri thức mới trong cơ sở dữ liệu phức tạp.

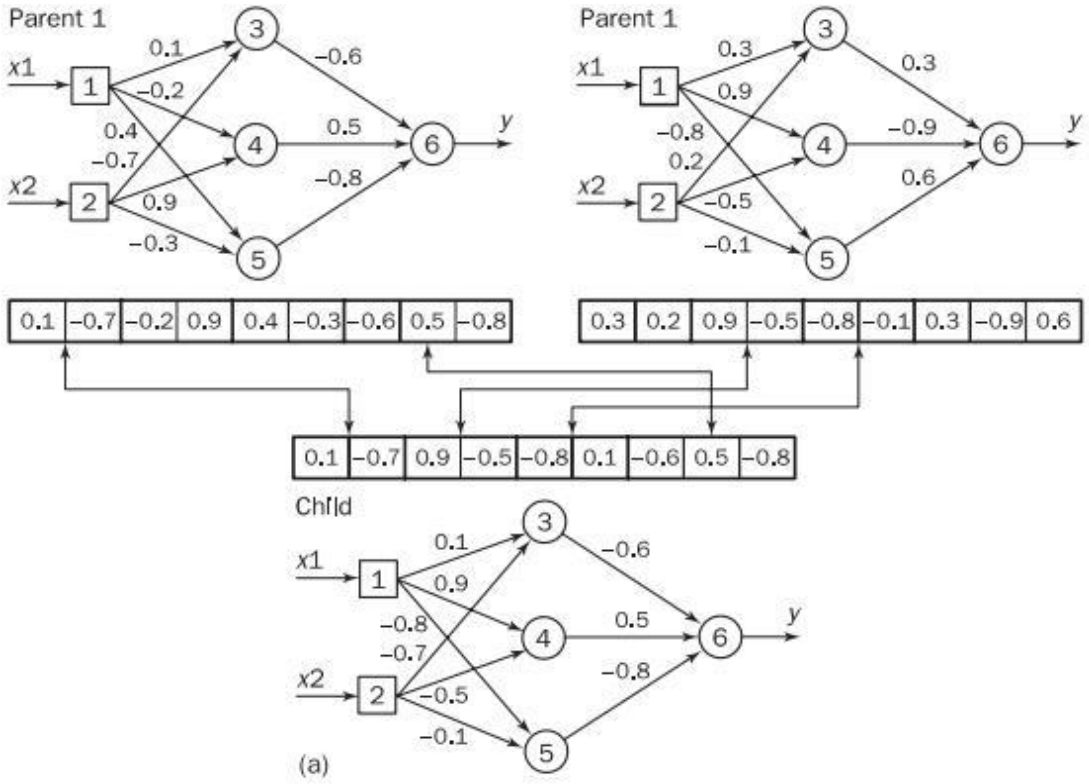
6.8 Hệ lai tiến hóa nơ ron

Mặc dù các mạng nơ ron được sử dụng để giải quyết nhiều vấn đề, nhưng chúng vẫn có một số hạn chế. Ví dụ, các thuật toán học lan truyền ngược thường được sử dụng vì nó là linh hoạt và dễ xử lý, nhưng nó có một nhược điểm: nó không đảm bảo giải pháp tối ưu toàn cục. Trong các ứng dụng, thuật toán lan truyền ngược có thể hội tụ về một tập hợp các giá trị tối ưu trọng cục bộ. Thuật toán di truyền là một kỹ thuật tối ưu hóa hiệu quả (cho toàn cục) và lựa chọn cấu trúc liên kết phù hợp.

Kỹ thuật tối ưu hóa trọng của mạng nơ ron bằng giải pháp tiến hóa (Montana và Davis, 1989; Whitley và Hanson , 1989; Ichikawa và Sawa, 1992), cụ thể là giải thuật di truyền có thể tóm tắt như sau. Để sử dụng các thuật toán di truyền, đầu tiên cần mô tả tên một NST (tiếng Anh: Chromosome). Các topology kết nối của mạng nơ ron thể hiện bởi một ma trận vuông kết nối (hình 6.11). Mỗi cột trong ma trận xác định loại kết nối từ một tế bào thần kinh (cột) khác (hàng), trong đó 0 biểu thị không có kết nối và 1 biểu thị kết nối mà trọng số được xác định thông qua quá trình học. Để chuyển đổi ma trận kết nối vào một nhiễm sắc thể, chỉ cần kết chuỗi các hàng của ma trận với nhau (hình 6.11).



Hình 6.11 : Tập mã hóa nhiễm sắc thể



Hình 6.12 : Lai ghép từ bố mẹ (Parent1 lai Parent1) tạo con (Child)

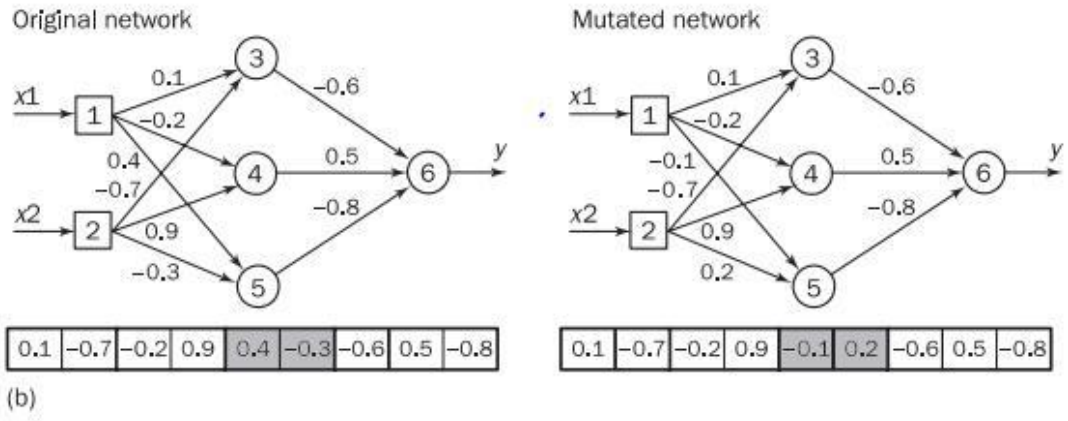
Giả sử, muốn tối ưu trọng cho mạng nơ ron nhiều lớp truyền thẳng, trọng ban đầu trong mạng được chọn ngẫu nhiên trong khoảng (-1, 1). Mô hình mạng nơ ron có thể biểu diễn bằng

một ma trận vuông, trong đó một số thực trong ma trận tương ứng với trọng số của một nơ ron, và số 0 trang ma trận (có nghĩa là không có kết nối giữa hai nơ ron) (hình 6.11). Tổng cộng, có 16 liên kết giữa các nơ ron (16 phần tử của ma trận có giá trị số; các phần tử còn lại đều bằng

Một nhiễm sắc thể là một bộ gen, một tập hợp của các số được đại diện bởi một 1 gen gồm 16 NTS, trong đó, mỗi gen tương ứng với một liên kết có trọng duy nhất trong mạng. Vì vậy, nếu xếp chuỗi các hàng của ma trận với nhau, bỏ qua số không, chúng ta có một NST (xem: Chromosome, Hình 6.11).

*Bước tiếp th*eo, chọn các toán tử di truyền: toán tử lai ghép và đột biến. Toán tử lai từhai NST là bố mẹ, tạo đứa trẻ có di truyền từ bố mẹ. Mỗi gen trong NSTcủa trẻ đại diện bởi các gen tương ứng của bố mẹ được lựa chọn ngẫu nhiên (Hình 6.12).

Hình 6.13 cho ví dụ về đột biến.



Hình 6.13 : Mạng nơ ron tối ưu hóa: (b) đột biến

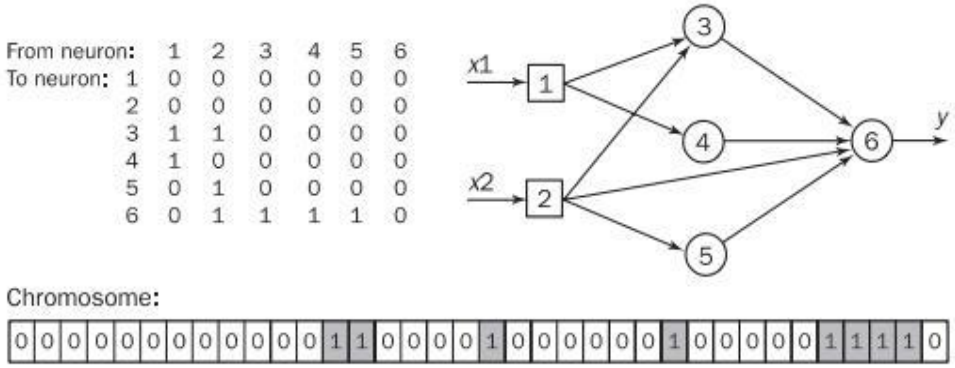
Xác định quy mô (số lượng) các quần thể (dân số), tức là số lượng các mạng với trọng lượng khác nhau, kiểu lai và đột biến xác suất và số lượng của các thế hệ.

Đến nay, chúng ta giả định rằng cấu trúc của mạng cố định, và tiến hóa chỉ được sử dụng để tối ưu hóa trọng lượng trong các mạng cung cấp. Tuy nhiên, các kiến trúc của mạng (tức là số lượng nơ ron và họ mối liên kết, tức là trọng số) sẽ quyết định sự thành công hay thất bại của các ứng dụng. Các thuật toán di truyền cũng có thể giúp lựa chọn mạng kiến trúc (tức số lượng nơ ron trong mạng).

Ý tưởng cơ bản để phát triển một kiến trúc mạng phù hợp là tìm kiếm di truyền trong một quần thể (Miller et al., 1989; Schaffer et al., 1992). Trước tiên, chọn phương pháp *mã hóa một kiến*

trúc mạng thành một nhiễm sắc thể. Có nhiều cách khác nhau để mã hóa cấu trúc của mạng. Điều quan trọng là quyết định bao nhiêu thông tin là cần thiết cho các mạng đại diện.

Cho một tập các ví dụ huấn luyện và một chuỗi nhị phân cho kiến trúc mạng; một GA cơ bản có thể được mô tả bởi những điều sau đây bước sau:



Hình 6.14 : Mạng mã hóa trực tiếp

Bước 1: Chọn kích thước của một số nhiễm sắc thể, lai tạo, đột biến và xác định số lượng các bước huấn luyện.

Bước 2: Xác định chức năng huấn luyện để đo hiệu suất của một NST riêng lẻ thíchhợp. Nói chung, sự thích hợp của mạng không nên chỉ dựa vào độ chính xác, mà còn về tốc độ học, kích thước và độ phức tạp của nó. Tuy nhiên, hiệu suất của mạng quan trọng hơn so với kích thước của nó, do đó các chức năng vẫn có thể được xác định bởi tổng các bình phương lỗi.

Bước 3: Ngẫu nhiên tạo ra một số ban đầu của nhiễm sắc thể.

Bước 4: Giải mã một NST riêng lẻ thành một mạng nơ ron. Tính tổng các lỗi bìnhphương và xác định tập thể dục của mạng.

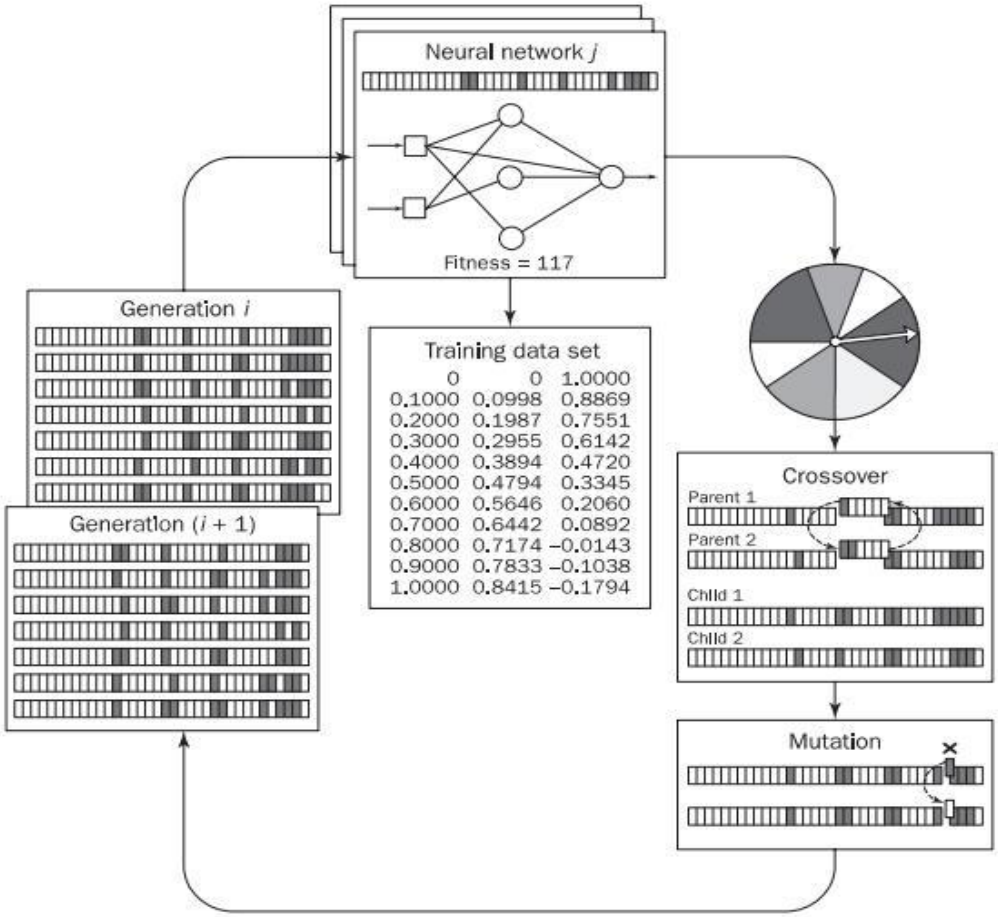
Bước 5: Lặp lại bước 4 cho đến khi tất cả cá thể trong dân số đã được xem xét.

Bước 6: Chọn một cặp NST cho giao phối, với xác suất thích hợp

Bước 7: Tạo một cặp NSTcon bằng cách áp dụng các gen lai và đột biến. Một nhà điều hành lai ngẫu nhiên, đơn giản là hoán đổi các hàng tương ứng giữa cha mẹ, tạo ra con cái. Đột biến một hoặc hai bit trong các NST với xác xuất thấp (khoảng 0.005)

Bước 8: Đặt cácNSTcon được tạo ra trong quần thể mới.

Bước 9: Lặp lại bước 6 cho đến khi kích thước của quần thểNSTmới bằng kích thướccủa quần thể ban đầu; sau đó thay thế quần thể ban đầu bằng NST mới.



Hình 6.15 : Vòng đời tiến hóa của 1 mạng nơ ron

Bước 10: Quay về bước 4 và lặp lại quá trình này cho đến khi một số quy định của cácthế hệ đạt được. Chu trình tiến hóa phát triển của mạng nơ ron được trình bày trong Hình 6.15. Ngoài việc học mạng nơ ron, tiến hóa tính toán cũng được sử dụng để tối ưu hóa các phép lai và chọn biến đầu vào.

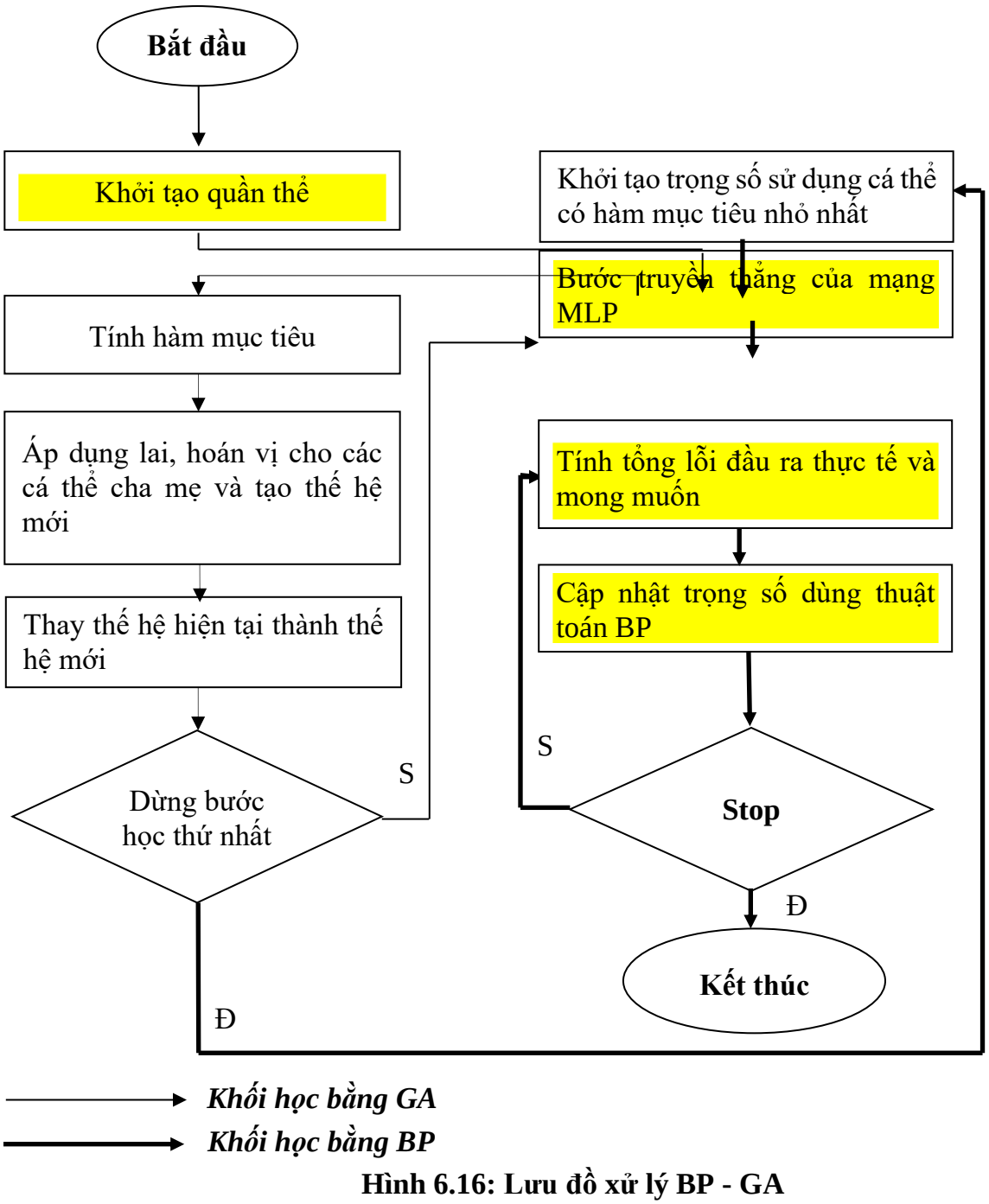
VÍ DỤ VỀ HỌC BẰNG CÁCH TÍCH HỢP ANN VÀ GA

Mặc dù mạng NN được cho là công cụ tốt với các hệ nhận dạng mẫu ảnh nhưng chi phí huấn luyện dùng mạng NN là rất tốn kém.

Trong chương trước, một trong các ứng dụng quan trọng của GA là được dùng để tối ưu hóa mạng NN. Gần đây một số tác giả đã đề xuất giải pháp kết hợp mạng NN và GA để tối ưu thao tác huấn luyện cho hệ thống. Ý tưởng của việc sử dụng GA là tìm trọng số tối ưu để đưa

vào mạng NN. Thuật toán huấn luyện [?] sử dụng GA kết hợp tối ưu quá trình huấn luyện mạng NN bằng thuật toán BP được mô tả như dưới đây.

Thuật toán được mô tả trong hình 6.16 gồm 02 giai đoạn học. Giai đoạn thứ nhất sử dụng GA với bước truyền thẳng để tăng tốc cả quá trình học. GA thực hiện tìm kiếm toàn cục và tìm kiếm tập trọng số khởi tạo gần tối ưu cho giai đoạn học thứ hai, ở đó, mỗi cá thể được sử dụng để mã hóa các trọng số của mạng NN.



Hình 6.16: Lưu đồ xử lý BP - GA

Cơ sở để GA tiến hành việc trên là sử dụng hàm mục tiêu (hay hàm hợp lý) là hàm lỗi của mạng NN tương ứng: tổng bình phương lỗi tiến tới nhỏ nhất:

$$E = \frac{1}{2}\sum_{i=1}^{n}(d_i - y_i)^2$$

do đó, nó trở thành bài toán tối ưu không giám sát để tìm tập các tham số quyết định bằng cách tối thiểu hàm mục tiêu hay chính là tìm giá trị nhỏ nhất của hàm trên. Giá trị tốt nhất của hàm mục tiêu (Fitness Function) cho GA trong một quần thể định nghĩa là giá trị nhỏ nhất của mục tiêu trong quần thể hiện tại

Giai đoạn thứ hai sử dụng thuật toán BP để huấn luyện mạng NN. Kết thúc giai đoạn thứ nhất, bộ trọng số tốt nhất tương ứng với cá thể ưu việt trong quần thể được lựa chọn làm trọng số khởi tạo cho thuật toán BP. Nó chính là bộ tham số cho phép xác định điểm gần cực tiểu nhất của hàm mục tiêu.

Với sự kết hợp này, thuật toán BP sẽ cần phải thay đổi một số yếu tố:

- Thuật toán không tự khởi tạo trọng số mà nhận các trọng số từ GA;
- Thành phần quán tính được loại bỏ để tăng tốc độ hội tụ và loại bỏ dao động.

Thuật toán BP – GA bao gồm hai giai đoạn học được mô tả bằng các bước như sau:

Giai đoạn 1: Giai đoạn học bằng GA

Thuật toán: Khởi tạo các nhiễm sắc thể ngẫu nhiên cho thế hệ hiện tại, khởi tạo các tham số làm việc và đặt nhiễm sắc thể đầu tiên là nhiễm sắc thể tốt nhất best_chromosome.

Bước 1. Lặp từ *i=1* đến kích thước quần thể:

Khởi tạo sub_total_fitness bằng 0 và sub_best_chromosome là rỗng

Bước 2. Lặp từ *j=1* đến độ dài của nhiễm sắc thể, thực hiện các công việc sau:

- Thực hiện thủ tục truyền thẳng cho mạng 3 lớp sử dụng hàm tương tác dạng sigmoid:

$f(u) = \frac{1}{1+e^{-\lambda.u}}$

- Tính hàm mục tiêu (2.13): $E = \frac{1}{2}\sum_{i=1}^{n}(d_i - y_i)^2$

- Tính tổng sai số total_fitness bằng cách tích lũy sub_total_fitness

Bước 3. Lưu best_chromosome vào sub_best_chromosome

Bước 4. So sánh các sub_best_chromosome với nhau và đặt sub_best_chromosome lớn nhất là best_chromosome.

Bước 5. Lặp từ *i=0* đến kích thước quần thể/2, thực hiện các thủ tục sau:

- Khởi tạo sub_total_fitness bằng 0 và sub_best_chromosome là rỗng
- Lặp từ *j=1* tới độ dài nhiễm sắc thể:

* Chọn các nhiễm sắc thể cha mẹ sử dụng phương pháp chọn theo bánh xe Roulette

* Áp dụng các phép lai và đột biến

- Lặp từ *k=1* đến độ dài nhiễm sắc thể:
* Thực hiện thủ tục truyền thẳng chomạng
* Tính các giá trị hàm mục tiêu cho các nhiễm sắc thể cha mẹ.
- Tính sub_total_fitness bằng cách tích lũy giá trị hàm mục tiêu của mỗi nhiễm sắc thể.
- Lưu best_chromosome vào sub_best_chromosome
Bước 6. Thay thế hệ cũ bằng thế hệ mới nếu thỏa mãn điều kiện dừng.

Giai đoạn 2: Giai đoạn học bằng thuật toán BP

- Đặt best_chromosome làm giá trị khởi tạo vector trọng số của mạng MLP, khởi tạo cấu trúc mạng MLP;
- Tính đầu ra thực sự của mạng MLP truyền thẳng;
- Tính lỗi giữa đầu ra mong muốn và đầu ra thực sự;
- Cập nhật trọng số mạng MLP sử dụng thuật toán BP.
-Trọng số tìm được bởi BP sẽ được lưu lại, đây chính là trọng số dùng để so sánh với giá trị tìm được ở pha nhận dạng mẫu.

TÀI LIỆU THAM KHẢO

TIẾNG ANH

[1]. Rajendra Arvind Akerkar and Priti Srinivas Sajja (2010). *"Knowledge-Based Systems"*. Jones and Bartlett Publisher.

[2]. Michael Negnevitsky. *Artificial Intelligence. A Guide to Intelligent Systems.* Addison Wesley. Pearson Education. 2002.

[3]. Stuart Rusell, Peter Norvig. *Artificial Intelligence- A modern Approach. 2003*

[4]. Li-Xin Wang. *A Course in Fuzzy Systems and Control*. Prentic-Hall International, Inc. 1997.

[6]. Guanrong Chen, Trung Tat Pham. *Fuzzy set, Fuzzy logic and Fuzzy Control Systems*. CRC Press. 2000.

[7]. Dawn E. Holmes, Lakhmi C. Jain. *Innovations in Machine Learning. Theory and Applications.* Springer. 2006

[8]. Chin-Teng Lin, C.S. George Lee. *Nơ ron Fuzzy Systems, A Neuro-Fuzzy Synergism to Intelligent Systems.* Prentic Hall International, Inc. 2002.

[9]. Geogre F. Luger. *Artificial Intelligence, Structures and Strategies for Complex Problem Solving.* Addison – Wesley Publishing Company, Inc. 2002.

TIẾNG VIỆT

[10]. Hoàng Kiếm, Đỗ Phúc, Đỗ Văn Nhơn. *Các hệ cơ sở trí thức.* Nhà xuất bản Đại học Quốc gia TP. Hồ Chí Minh. 2007.

[10]. Hoàng Kiếm. Đinh Nguyễn Anh Dũng. *Trí tuệ nhân tạo.* Nhà xuất bản Đại học Quốc gia TP. Hồ Chí Minh. 2005.

[11]. Nguyễn Quang Hoan. *Trí tuệ nhân tạo.* Học viện Công nghệ Bưu chính Viễn thông. 2007.

[12]. Huỳnh Thái Hoàng. *Hệ thống điều khiển thông minh.* Nhà xuất bản Đại hoc Quốc gia TP. Hồ Chí Minh. 2006.

[13]. Nguyễn Như Phong. *Tính toán mềm và ứng dụng.* Nhà xuất bản Khoa học và Giáo dục. 2007

[14]. Nguyễn Đình Thúc. *Lập trình tiến hóa.* Nhà xuất bản Giáo duc. 2001.

[15]. Đỗ Trung Tuấn. *Hệ chuyên gia.* Nhà xuất bản Giáo dục. 1999

[16]. Nguyễn Trọng Thuần. *Điều khiển logic và ứng dụng. Nhà xuất bản Khoa học và Kỹ thuật.* 2009.

[17]. B. Bouchon Meunier, Hồ Thuần, Đặng Thanh Hà. *Logic mờ và ứng dụng.* Nhà xuất bản Đại học Quốc gia Hà Nội. 2007.