Chapter 1

Virtual Worlds as Fuzzy Dynamical Systems

Julie A. Dickerson and Bart Kosko

Electrical and Computer Engineering Department, Iowa State University, Ames, IA, 50010

Department of Electrical Engineering-Systems, Signal and Image Processing Institute, University of Southern California, Los Angeles, CA, 90089-2564

Abstract

Fuzzy cognitive maps (FCMs) can structure virtual worlds that change with time. A FCM links causal events, actors, values, goals, and trends in a fuzzy feedback dynamical system. A fuzzy rule defines a fuzzy patch in the input-output state-space of a system. It links commonsense knowledge with state-space geometry. A FCM connects the fuzzy rules or causal flow paths that relate events. It can guide actors in a virtual world as the actors move through a web of cause and effect and react to events and to other actors. Experts draw FCM causal pictures of the virtual world. Complex FCMs can give virtual worlds with "new" or chaotic equilibrium behavior. Simple FCMs give virtual worlds with periodic behavior. They map input states to limit-cycle equilibria. A FCM limit cycle repeats a sequence of events or a chain of actions and responses. Limit cycles can control the steady-state rhythms and patterns in a virtual world. In nested FCMs each causal concept can control its own FCM or fuzzy function approximator. Appendix A shows how an additive fuzzy system can uniformly approximate any continuous (or bounded measurable) function on a compact domain to any degree of accuracy. This gives levels of fuzzy systems that can choose goals and causal webs as well as move objects and guide actors in the webs. FCM matrices sum to give a combined FCM virtual world for any number of knowledge sources. Adaptive FCMs change their fuzzy causal web as causal patterns change and as actors act and experts state their causal knowledge. Neural learning laws change the causal rules and the limit cycles. Actors learn new patterns and reinforce old ones. In complex FCMs the user can choose the dynamical structure of the virtual world from a spectrum that ranges from mildly to wildly nonlinear. We use an adaptive FCM to model an undersea virtual world of dolphins, fish, and sharks.

1.1 Fuzzy Virtual Worlds

What is a virtual world? It is what changes in a"virtual reality" [1] or "cyberspace" [2]. A virtual world links humans and computers in a causal medium that can trick the mind or senses.

At the broadest level a virtual world is a dynamical system. It changes with time as the user or an actor moves through it. In the simplest case only the user moves in the virtual world. In general both the user and the virtual world change and they change each other.

Change in a virtual world is causal. Actors cause events to happen as they move in a virtual world. They add new patterns of cause and effect and respond to old ones. In turn the virtual world acts on the actors or on their physical or social environments. The virtual world changes their behavior and can change its own web of cause of effect. This feedback causality between actors and their virtual world makes up a complex dynamical system that can model events, actors, actions, and data as they unfold in time.

Virtual worlds are fuzzy as well as fedback. Events occur and concepts hold only to some degree. Events cause one another to some degree. In this sense virtual worlds are fuzzy causal worlds. They are fuzzy dynamical systems. A fuzzy rule defines a fuzzy patch in the input-output state-space of a system and links commonsense knowledge with state-space geometry. An additive fuzzy system approximates a function by covering its graph with fuzzy patches in the input-output state space and averaging patches that overlap.

How do we model the fuzzy feedback causality? One way is to write down the differential equations that show how the virtual "flux" or "fluid" changes in time. This gives an exact model. The Navier-Stokes equations [3] used in weather models give a fluid model of how actors move in a type of virtual world. They can show how clouds or tornadoes form and dissolve in a changing atmosphere or how an airplane flies through pockets of turbulence. The inverse kinematic equations of robotics [4] show how an actor moves through or grasps in a virtual joint space. The coupled differential equations of blood glucose and insulin [5] cast the patient as a diabetic actor awash in a virtual world of sugar and hormones. Such math models are hard to find, hard to solve, and hard to run in realtime. They paint too fine a picture of the virtual world.

Fuzzy cognitive maps (FCMs) can model the virtual world in large fuzzy chunks. They model the causal web as a fuzzy directed graph [6],[7]. The nodes and edges show how causal concepts affect one another to some degree in the fuzzy dynamical system. The "size" of the nodes gives the chunk size. In a virtual world the concept nodes can stand for events, actions, values, moods, goals, or trends. The causal edges state fuzzy rules or causal flows between concepts. In a predator-prey world survival threat increases prey runaway. The fuzzy rule states how much one node grows or falls as some other node grows or falls.

Experts draw the FCMs as causal pictures. They do not state equations. They state concept nodes and link them to other nodes. The FCM system turns each picture into a matrix of fuzzy rule weights. The system weights and adds the FCM matrices to combine any number of causal pictures. More FCMs tend to sum to a better picture of the causal web with rich tangles of feedback and fuzzy edges even if each expert gives binary (present or absent) edges. This makes it easy to add or delete actors or to change the background of a virtual world or to combine virtual worlds that are disjoint or overlap. We can also let a FCM node control its own FCM to give a nested FCM in a hierarchy of virtual worlds. The node FCM can model the complex nonlinearities between the node's input and output. It can drive the motions, sounds, actions, or goals of a virtual actor.

The FCM itself acts as a nonlinear dynamical system. Like a neural net it maps inputs to output equilibrium states. Each input digs a path through the virtual state space. In simple FCMs the path ends in a fixed point or limit cycle. In more complex FCMs the path may end in an aperiodic or "chaotic" attractor. These fixed points and attractors represent *meta-rules* of the form "If input, then attractor or fixed point." The rules are stored in the cube itself.

1.2 Additive Fuzzy Systems

A fuzzy system approximates a function by covering its graph with fuzzy patches and averaging patches that overlap. The approximation improves as the fuzzy patches grow in number and shrink in size. Figure 1.1 shows how fuzzy patches in the input-output product space $X \times Y$ cover the real function $f : X \to Y$. In Figure 1.1(a) a few large patches approximate f. In Figure 1.1(b) several smaller patches better approximate f. The approximation improves as we add more small patches but storage and complexity costs increase. This section gives the algebraic details of the fuzzy approximation.

An additive fuzzy system adds the then-parts of fired if-then rules. Other fuzzy systems combine the then-part sets with pairwise maxima. A fuzzy system has rules of the form "If input conditions hold, then output conditions hold" or "If X is $\mathbf{A}$, then Y is $\mathbf{B}$" for fuzzy sets $\mathbf{A}$ and $\mathbf{B}$. Each fuzzy rule defines a fuzzy patch or a Cartesian product $\mathbf{A} \times \mathbf{B}$ as shown in Figure 1.2. The fuzzy system covers the graph of a function with fuzzy patches and averages patches that overlap. Uncertain

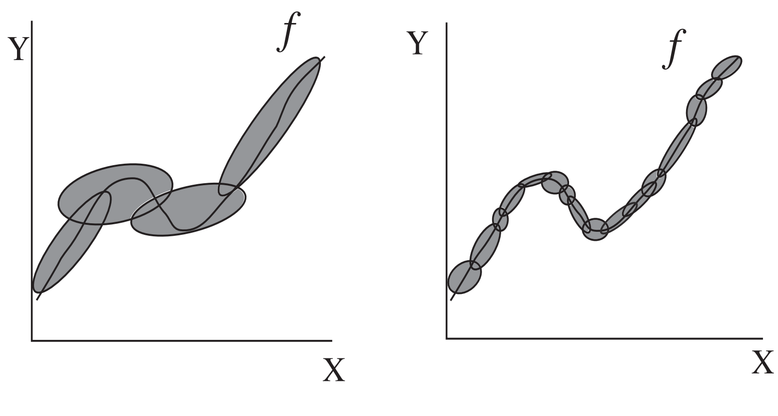

Figure 1.1 (a) Four large fuzzy patches cover part of the graph of the unknown function $f : X \to Y$. Fewer patches can decrease computation but decrease approximation accuracy. (b) More smaller fuzzy patches better cover f but at greater computational cost. Each fuzzy rule defines a patch in the product space $X \times Y$. A large but finite number of fuzzy rules or precise rules can cover the graph with arbitrary accuracy.

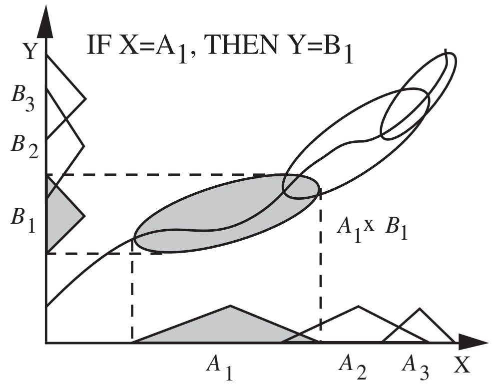

Figure 1.2 The fuzzy rule patch "If X is fuzzy set A_1, then Y is fuzzy set B_1" is the fuzzy Cartesian product $A_1 \times B_1$ in the input-output product space $X \times Y$.

fuzzy sets give a large patch or fuzzy rule. Small or more certain fuzzy sets give small patches.

Additive fuzzy systems fire all rules in parallel and average the scaled output sets B'_j to get the output fuzzy set $\mathbf{B}$ as in Figure 1.3. Correlation product inference scales each output set B_j by the degree $m_{A_j}(x)$ (or $a_j(x)$) that the rule "IF A_j, THEN B_j" fires. Most rules fire to degree 0. Defuzzification of $\mathbf{B}$ gives a number or a control signal output. Centroidal defuzzification with correlation product inference [8] gives the output value y_k at time k:

$$\begin{aligned} y_k \quad = F(x_k) \quad &= \frac{\int y m_{\mathbf{B}}(y)dy}{\int m_{\mathbf{B}}(y)dy} \qquad (1) \\ &= \frac{\sum_{j=1}^{m} Volume\,(B'_j)Centroid(B'_j)}{\sum_{j=1}^{m} Volume(B'_j)} \\ &= \frac{\sum_{j=1}^{m} c_{y_j} V_j m_{A_j}(x_k)}{\sum_{j=1}^{m} V_j m_{A_j}(x_k)} \end{aligned}$$

V_j is the volume of the jth output set B_j. We can always normalize the finite volumes V_j to unity to keep some rules from dominating others. c_{y_j} is the centroid of the jth output set. Fit value $m_{A_j}(x_k)$ scales the output set B_j. m is the number of output fuzzy sets. In practice $\mathbf{A}$ is connected. It need not be. But then we could view the rule "If X is $\mathbf{A}$, then Y is $\mathbf{A}$ " as two or more rules of the form "If X is $\mathbf{A}$, then Y is $\mathbf{B}_1$ " and "If X is $\mathbf{A}$, then Y is $\mathbf{B}_2$ " where $\mathbf{B}_1$ and $\mathbf{B}_2$ are two of the disjoint components of $\mathbf{A}$. So assume $\mathbf{B}$ is connected. Then the rule patch $\mathbf{A} \times \mathbf{B}$ is connected and a patch proper.

The additive fuzzy system computes the conditional expectation $E[Y|X = x]$ if we *view* fuzzy sets as random sets [9],[10] — if the curve $m_A : [0,1] \rightarrow X$ is a locus of two point conditional densities. Then $m_A(x)$ is the probability of $\mathbf{A}$ given that X takes on x or $m_A(x) = p(x \in A \mid X = x)$ and $m_A(x) = p(x \notin A \mid X = x)$. The conditional mean $E[Y|X]$ is the mean-squared optimal estimate of Y given the information known about X—given the information in the random or fuzzy subsets $\mathbf{A}$ of X.

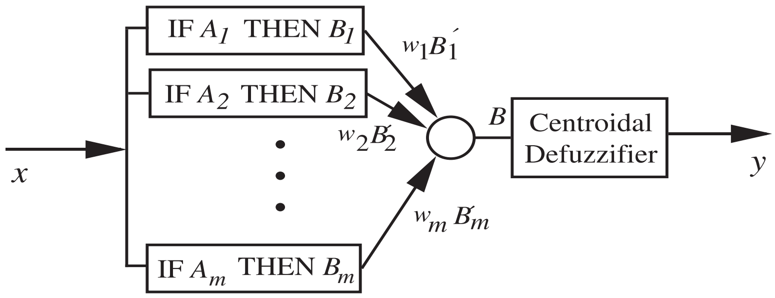

Figure 1.3 Additive fuzzy system architecture. The input x_k acts as a delta pulse (or unit bit vector) and fires each rule to some degree. The system adds the scaled output fuzzy sets. The centroid of this combined set gives the output value y_k. The system computes the conditional expectation value $E[Y|X = x_k]$.

In Appendix A we show that a fuzzy system can approximate any continuous real function defined on a compact (closed and bounded in R^n domain and show that even a bivalent expert system can uniformly approximate a bounded measurable function. The fuzzy systems have a feedforward architecture that resembles the feedforward multilayer neural systems used to approximate functions [11]. The uniform approximation of continuous functions allows us to replace each continuous fuzzy set with a finite discretization or a point in a unit hypercube [8] of high dimension.

Combining the scaled or "fired" consequent fuzzy sets $B_1', \ldots, B_m'$ in Figure 1.3 with pairwise maximum gives the envelope of the fuzzy sets and tends towards the uniform distribution. Max combination ignores overlap in the fuzzy sets B_j . Sum combination adds overlap to the peakedness of **B**. When the input changes slightly, the additive output **B** changes slightly. The max-combined output may ignore small input changes since for large sets of rules most change occurs in the overlap regions of the fuzzy sets B_j'. Here overlap problem arises since the centroid tends to stay the same for small changes in input. But the centroid smoothly tracks changes in the fuzzy-set sum (1).

We now formally derive the *standard* additive model (SAM) in (1) that we shall use in this chapter and show how an additive fuzzy system acts as a conditional mean. A general additive fuzzy system is a map $F : R^n \rightarrow R^p$. Both in practice and in uniform approximation proofs we restrict the domain to a compact subset $U \subset R^n$ but we need not. Watkins [12] has proved that an additive fuzzy system with just two rules can exactly *represent* any bounded function $f : R \rightarrow R$ even if f is not continuous. In this case the domain is the entire real line.

The additive fuzzy system stores m fuzzy patches $A_j \times B_j$ or rules of the form "$IfXisA_j, thenYisB_j$" Here $A_j \subset R^n$ and $B_j \subset R^p$ multivalued or "fuzzy" sets with set functions $a_j : R^n \rightarrow [0, 1]$ and $b_j : R^p \rightarrow [0, 1]$. We also use the membership notation $m_{A_j}(x)$ and $m_{B_j}(y)$ in this chapter for the set functions. For the following derivation we use the *fit* (fuzzy unit) notation a_j and b_j for simplicity.

In practice we define the then-part set A_j by its n coordinate-projection sets $A_j^1, \ldots, A_j^n$ and thus $A_j = A_j^1 \times A_j^2 \times \ldots \times A_j^n$. How we define this fuzzy Cartesian product dictates the conjunctive (or t-norm) form of how we factor the joint set function a_j into its coordinate set functions $a_j^1, \ldots, a_j^n$. Minimum combination is the most popular form

$$a_j(x) = a_j^1(x_1) \wedge a_j^1(x_2) \wedge a_j^2(x_2) \wedge \ldots \wedge a_j^n(x_n) \tag{2}$$

for input vector $x = (x_l, \ldots, x_n)$. Product combination

$$a_j(x) = \prod_{i=1}^{n} a_j^i(x_i) \tag{3}$$

can simplify the analysis and computation of additive systems with Gaussian [13] or radial-basis [14] set functions of the form

$$a_j^i(x_i) = s_i^j \exp\left[-\frac{1}{2}\left(\frac{x_i - \bar{x}_i^j}{\sigma_i^j}\right)^2\right] \tag{4}$$

for scaling constant $0 < s_i^j \leq 1$. The choice of combination operator does not affect the structure of the standard model (1).

The first step to show the conditional-mean property is to view each scalar fuzzy set a_j^i as a random set [10]. Then $a_j^i(x_i)$ is not the degree to which $x_i \in A_j^i$ but the conditional probability $p(x_i \in A_j^i \mid X_i = x_i)$. In the same way the complement fit value $1 - a_j^i(x_i)$ is just the dual conditional probability: $p(x_i \notin A_j^i \mid X_i = x_i)$. So A_j^i is not a locus of membership degrees but a locus of two-point conditional densities.

The next step is the additive step. The m fit values $a_j^i(x_i)$ "fire" the then-part sets B_j to give the "inferred" sets B_j'. Again the result combines $a_j^i(x_i)$ and B_j in some conjunctive (t-norm) way and again it depends on how we define the Cartesian patch $A_j \times B_j$. Here min is less popular than product. The min "clip" discards all information in B_j above the fit height $a_j^i(x_i)$ and can thus change the centroid of B_j if B_j is not symmetric. Product combination or *correlation product decoding* [8] keeps all relative information in B_j and does not change its centroid:

$$B_j' = a_j^i(x) B_j \tag{5}$$

We use (5) as a default for a SAM. We can also view the inferred sets B_j' as random sets. An additive model [8] then sums these inferred sets to produce the final output set B:

$$B = \sum_{j=1}^{m} B_j' \tag{6}$$

Each rule can have a weight w_j that scales B_j' in (6). Learning can change these weights or we can use them to model frequency or "usuality" rule weights. Here we take them as unity: $w_j = 1$.

The only constraint on B or b is that it have a finite integral or *volume*:

$$0 < V = \int b(y)\, dy < \infty \tag{7}$$

This means that each input x fires at least one rule to non zero degree. Then B/V is a probability density function. Indeed it is a conditional probability since it depends on the fuzzy variable X taking on the input value x (the ratio of a joint to a marginal):

$$\frac{B}{V} = p(Y|X = x) \tag{8}$$

Note this does not require that we view the if-part sets as probability density functions. They are not. Each is a locus of continuum-many two-point conditional densities. Formally the system accepts input x_0 as a delta pulse to produce the m fit values:

$$a_j\,(x_0) = \int \delta\;(x - x_0)\;\; a_j\,(x)\;\; dx \tag{9}$$

Then the additive system output $F(x)$ equals the centroid of B:

$$\begin{aligned} F(x) &= \frac{\int yb(x,y)dy}{\int b(x,y)dy} && (10)\\ &= \int yp(Y|X = x)dy && (11)\\ &= E[Y|X = x] && (12) \end{aligned}$$

What holds for one realization of a random vector holds for them all. Hence $F = E[Y \mid X]$ as claimed. The SAM model (1) then computes the global conditional mean value $E[Y \mid X = x]$ as a convex sum of local conditional means in (26).

We now assume that the additive fuzzy system maps real vectors into scalars $F : R^n \rightarrow R$. Then put the additive assumption (6) in the centroidal output (10) to get the standard form of a additive model [8] we use in this chapter:

$$\begin{aligned} F(x) &= \frac{\int_{-\infty}^{\infty} y \sum\limits_{j=1}^{m} b'_j(y)dy}{\int_{-\infty}^{\infty} \sum\limits_{j=1}^{m} b'_j(y)dy} && (13)\\ &= \frac{\sum\limits_{j=1}^{m} \int_{-\infty}^{\infty} ya_j(x)b_j(y)dy}{\sum\limits_{j=1}^{m} \int_{-\infty}^{\infty} a_j(x)b_j(y)dy} && (14)\\ &= \frac{\sum\limits_{j=1}^{m} a_j(x)V_j \frac{\int_{-\infty}^{\infty} yb_j(y)dy}{V_j}}{\sum\limits_{j=1}^{m} a_j(x)V_j} && (15) \end{aligned}$$

$$= \frac{\sum_{j=1}^{m} a_j(x) V_j c_j}{\sum_{j=1}^{m} a_j(x) V_j} \tag{16}$$

for then-part set volumes

$$V_j = \int_{-\infty}^{\infty} b_j(y) dy \tag{17}$$

and then-part set centroids

$$c_j = \frac{\int_{-\infty}^{\infty} y b_j(y) dy}{\int_{-\infty}^{\infty} b_j(y) dy} \tag{18}$$

The model in (16) is the standard additive model or SAM and the same as (1). It holds for $F : R^n \to R^p$ as well.

The standard model (16) reduces to the Gaussian additive model of Wang and Mendel [13]

$$F(x) = \frac{\sum_{j=1}^{m} \bar{z}^j (\prod_{i=1}^{n} \mu_{A_i^j}(x_i))}{\sum_{j=1}^{m} (\prod_{i=1}^{n} \mu_{A_i^j}(x_i))} \tag{19}$$

for the Gaussian if-part set in (4) and Gaussian then-part sets with these identifications:

$$y = z \tag{20}$$

$$a_j(x) = \prod_{i=1}^{n} a_j^i(x_i) \tag{21}$$

$$= \prod_{i=1}^{n} \mu_{A_i^j}(x_i) \tag{22}$$

$$V_j = 1 \tag{23}$$

$$c_j = \bar{z}^j \tag{24}$$

The choice of product combination (2) gives (21) and (22). The unity volume follows in (23) since Wang and Mendel integrate their m then-part Gaussian sets over all of R (and thus use the scaling constant in (4) to account for the input truncation to a compact set). (24) follows because the mode of a Gaussian set equals its centroid and Wang and Mendel use the mode definition "is the point in R at which $\mu_{B_j}(z)$ achieves its maximum value." They used the Stone-Weierstrass Theorem to prove that additive Gaussian systems with all-product combination in (19) are uniform approximators of continuous maps on compact sets. This non-constructive result is a special case of the uniform approximation theorem for all additive systems. We review this general theorem and its constructive proof in Appendix A. It holds as well for Gaussian sets with min combination (2) of if-part fit values or min clipping of then-part sets B_j.

Next observe that taking the centroid of the additive B in (6) leads to a set of convex coefficients:

$$F(x) = \frac{\sum_{j=1}^{m} a_j(x)\, V_j\, c_j}{\sum_{j=1}^{m} a_j(x)\, V_j} \tag{25}$$

$$= \sum_{j=1}^{m} p_j(x)\, c_j \tag{26}$$

for the m convex coefficients (or m terms of a discrete probability density)

$$p_j(x) = \frac{a_j(x)\, V_j}{\sum_{k=1}^{m} a_k(x)\, V_k} \tag{27}$$

Wang and Mendel [13] refer to the convex sum of centroids (26) in the Gaussian case as a "fuzzy basis function expansion" even though the "basis functions" $p_j(x)$ in (27) are not orthogonal.

Feedforward fuzzy systems suffer exponential rule explosion as the number of inputs increases. Optimal rules [15] and function representation [16] offer two ways to deal with this "curse of dimensionality." Appendix B shows how supervised learning can tune the parameters of an additive fuzzy system. FCMs allow a fuzzy system to approximate nonlinear dynamical systems with a fixed number of rules [17].

1.3 Fuzzy Cognitive Maps

Fuzzy cognitive maps (FCMs) are fuzzy signed digraphs with feedback [6],[7]. An FCM is an additive fuzzy system with feedback. Nodes stand for fuzzy sets or events that occur to some degree. The nodes are causal concepts. They can model events, actions, values, goals, or lumped-parameter processes.

Directed edges stand for fuzzy rules or the partial causal flow between the concepts. The sign (+ or -) of an edge stands for causal increase or decrease. The positive edge rule in Figure 1.4a states that a survival threat increases runaway. It is a positive causal connection. The runaway response grows or falls as the threat grows or falls. The negative edge rule in Figure 1.4b states that running away from a predator decreases the survival threat. It is a negative causal connection. The survival threat grows the less the prey runs away and falls the more the prey runs away. The two rules in Figure 1.4c define a minimal feedback loop in the FCM causal web.

A FCM with n nodes has n^2 edges. The nodes $C_i(t)$ are fuzzy sets and so take values in $[0, 1]$. So a FCM state is the *fit* (fuzzy unit) vector $\mathbf{C}(t) = (C_1(t), \ldots, C_n(t))$ and thus a point in the fuzzy hypercube $I^n = [0, 1]^n$. A FCM inference is a path or point sequence in I^n. It is a fuzzy process or indexed family of fuzzy sets $\mathbf{C}(t)$. The FCM can only "forward chain" [18] to answer what-if questions. Nonlinearities do not permit reverse causality. FCMs cannot "backward chain" to answer why questions.

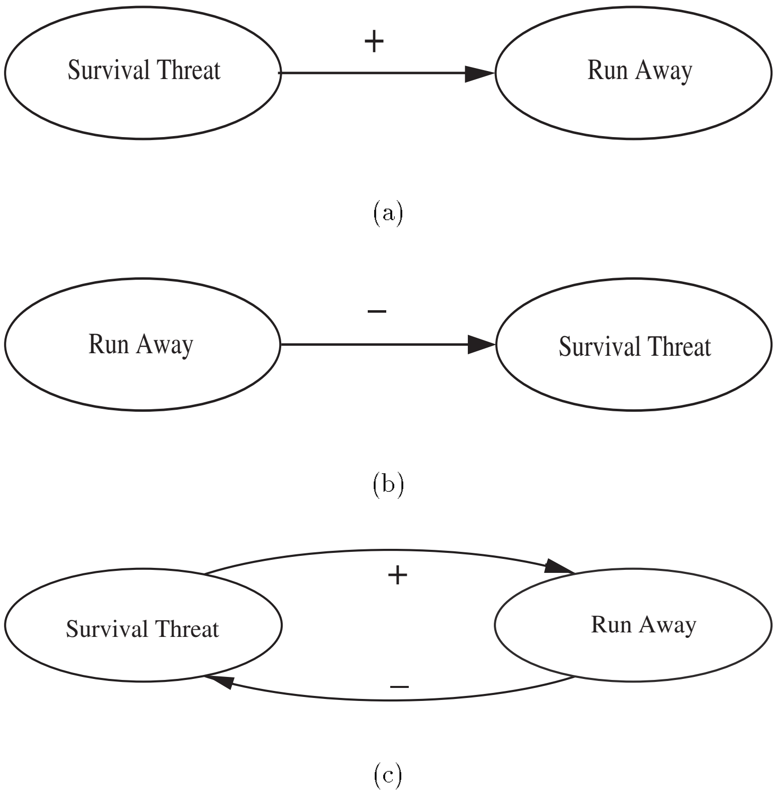

Figure 1.4 Directed edges stand for fuzzy rules or the partial causal flow between the concepts. The sign (+ or -) of an edge stands for causal increase or decrease. (a) A positive edge rule in states that a survival threat increases runaway. (b) A negative edge rule states that running away from a predator decreases the survival threat. (c) Two rules define a minimal feedback loop in the FCM causal web.

The FCM nonlinear dynamical system acts as a neural network. For each input state $\mathbf{C}(0)$ it digs a trajectory in I^n that ends in an equilibrium attractor $\mathbf{A}$. The FCM quickly converges or "settles down" to a fixed point, limit cycle, limit torus, or chaotic attractor in the fuzzy cube. Figure 1.5 shows three attractors or meta-rules for a 2-D dynamical FCM.

The output equilibrium is the answer to a causal what-if question: What if $\mathbf{C}(0)$ happens? In this sense each FCM stores a set of global rules of the form "If $\mathbf{C}(0)$, then equilibrium attractor $\mathbf{A}$."

The size of the attractor regions in the fuzzy cube governs the number of these global rules or "hidden patterns" [7]. All points in the attractor region map to the attractor. A FCM with a global fixed point has only one global rule. All input balls "roll" down its "well." FCMs can have large and small attractor regions in the fuzzy cube. The attractor types can vary in complex FCMs with highly nonlinear concepts and edges. Then one input state may lead to chaos and a more distant input state may end in a fixed point or limit cycle.

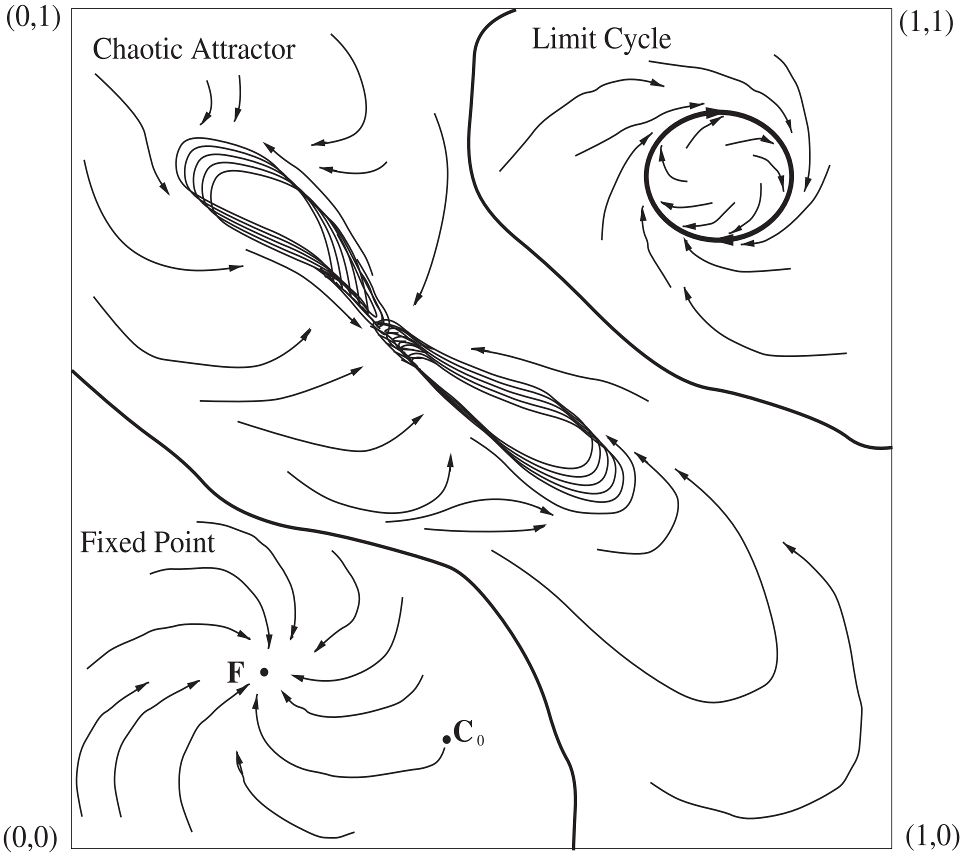

Figure 1.5 The unit square is the state space for a FCM with two nodes. The system has at most four fuzzy edge rules. In this case it has three fuzzy meta-rules of the form "If input state vector **C** then attractor **A**." The state $\mathbf{C}_0$ converges to a fixed point **F**.

1.3.1 Simple FCMs

Simple FCMs have bivalent nodes and trivalent edges. Concept values C_i take values in $\{0,1\}$. Causal edges take values in {-1,0,1}. So for a concept each simple FCM state vector is one of the 2^n vertices of the fuzzy cube I^n. The FCM trajectory hops from vertex to vertex. I^n ends in a fixed point or limit cycle at the first repeated vector.

We can draw simple FCMs from articles, editorials, or surveys. Most persons can state the sign of causal flow between nodes. The hard part is to state its degree or magnitude. We can average expert responses [7],[19] as in equation (30) below or use neural systems to learn fuzzy edge weights from data. The expert responses can initialize the causal learning or modify it as a type of forcing function.

Figure 1.6 shows a simple FCM with five concept nodes. The connection or edge matrix **E** lists the causal links between nodes:

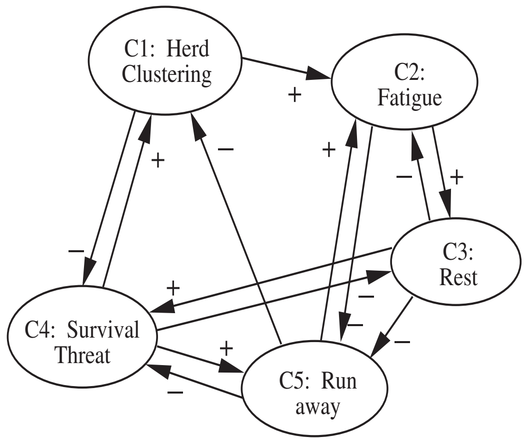

Figure 1.6 Simple FCM with five concept nodes. Edges show directed causal flow between nodes.

$$\mathbf{E} = \begin{array}{c} \\ C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \end{array} \begin{array}{c} \begin{array}{ccccc} C_1 & C_2 & C_3 & C_4 & C_1 \end{array} \\ \left[\begin{array}{rrrrr} 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & -1 \\ 0 & -1 & 0 & 1 & -1 \\ 1 & 0 & -1 & 0 & 1 \\ -1 & 1 & 0 & -1 & 0 \end{array}\right] \end{array}$$

The ith row lists the connection strength of the edges e_{ik} directed out from causal concept C_i. The ith column lists the edges e_{ki} directed into C_i. C_i causally increases C_k if $e_{ik} > 0$, decreases C_k if $e_{ik} < 0$, and has no effect if $e_{ik} = 0$. The causal concept C_4 causally increases concepts C_1 and C_5. It decreases C_3. Concepts C_1 and C_5 decrease C_4. Concept C_3 increases C_4.

1.3.2 FCM Recall

FCMs recall as the FCM dynamical system equilibrates. Simple FCM inference thresholds a matrix-vector multiplication [7],[20]. State vectors $\mathbf{C}_n$ cycle through the FCM adjacency matrix $\mathbf{E} : \mathbf{C}_1 \to \mathbf{E} \to \mathbf{C}_2 \to \mathbf{E} \to \mathbf{C}_3 \to \ldots$. The system nonlinearly transforms the weighted input to each node C_i

$$C_i\left(t_{n+1}\right) = S\left[\sum_{k=1}^{N} e_{ki}\left(t_n\right) C_k\left(t_n\right)\right] \tag{28}$$

Here $S(x)$ is a bounded signal function. For simple FCMs the sigmoid function

$$S\left(y\right) \;=\; \frac{1}{1+e^{-c(y-T)}} \tag{29}$$

with large $c > 0$ approximates a binary threshold function.

Simple threshold FCMs quickly converge to stable limit cycles or fixed points [7],[20]. These limit cycles show "hidden patterns" in the causal web of the FCM.

The FCM in Figure 1.6 gives a three-step limit cycle when input state $\mathbf{C}_1 = [0\ \ 0\ \ 0\ \ 1\ \ 0]$ fires the FCM network. Equation (28) and binary thresholding gives the four step limit cycle $\mathbf{C}_1 \to \mathbf{C}_2 \to \mathbf{C}_3 \to \mathbf{C}_4 \to \mathbf{C}_1$.

$\mathbf{C}_1 = [0\ \ 0\ \ 0\ \ 1]$

$$
\begin{aligned}
\mathbf{C}_1\mathbf{E} &= [1\ \ 0\ \ -1\ \ 0\ \ 1] \to \mathbf{C}_2 &= [1\ \ 0\ \ 0\ \ 0], \\
\mathbf{C}_2\mathbf{E} &= [-1\ \ 2\ \ 0\ \ -2\ \ 0] \to \mathbf{C}_3 &= [0\ \ 1\ \ 0\ \ 0\ \ 0], \\
\mathbf{C}_3\mathbf{E} &= [0\ \ 0\ \ 1\ \ 0\ \ -1] \to \mathbf{C}_4 &= [0\ \ 0\ \ 1\ \ 0\ \ 0], \\
\mathbf{C}_4\mathbf{E} &= [0\ \ -1\ \ 0\ \ 1\ \ -1] \to \mathbf{C}_1 &= [0\ \ 0\ \ 0\ \ 1\ \ 0].
\end{aligned}
$$

In a virtual world the limit cycle might make in order wake up, go to work, come home, then wake up again. Some complex actions such as walking break down into simple cycles of movement [21].

Each node in a simple FCM turns actions or goals on and off. Each node can control its own FCM, fuzzy control system, goal-directed animation system, force feedback, or other input-output map. The FCM can control the temporal associations or timing cycles that structure virtual worlds. These patterns establish the rhythm of the world. "Grandmother" nodes can control the time spent on each step in a FCM "avalanche" [22]. This can change the update rate and thus the timing for the network [22].

1.3.3 Augmented FCMs

FCM matrices additively combine to form new FCMs [6]. This allows combination of FCMs for different actors or environments in the virtual world. The new (augmented) FCM includes the union of the causal concepts for all the actors and the environment in the virtual world. If a FCM does not include a concept, then those rows and columns are all zero. The sum of the augmented (zero-padded) FCM matrices for each actor forms the virtual world:

$$
\mathbf{F} = \sum_{i=1}^{n} w_i \mathbf{F}_i \tag{30}
$$

The w_i are positive weights for the ith FCM $\mathbf{F}_i$. The weights state the relative value of each FCM in the virtual world and can weight any subgraph of the FCM. Figure 1.7a shows three simple FCMs. Equation (30) combines these FCMs to give the new simple FCM in Figure 1.7b that has fuzzy or multivalued edges:

$$
\mathbf{F} = \frac{1}{3}(\mathbf{F}_1 + \mathbf{F}_2 + \mathbf{F}_3) = \frac{1}{3} \cdot \begin{bmatrix} 0 & 2 & -1 & 0 & 0 & -1 \\ 0 & 0 & 2 & 3 & 1 & 0 \\ 0 & 0 & 0 & 2 & 1 & 0 \\ 2 & 0 & 0 & 0 & -1 & 0 \\ 0 & -2 & 0 & 0 & 0 & 0 \\ 1 & -1 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{31}
$$

The FCM sum (30) helps knowledge acquisition. Any number of experts can describe their FCM virtual world views and (30) will weight and combine them [19].

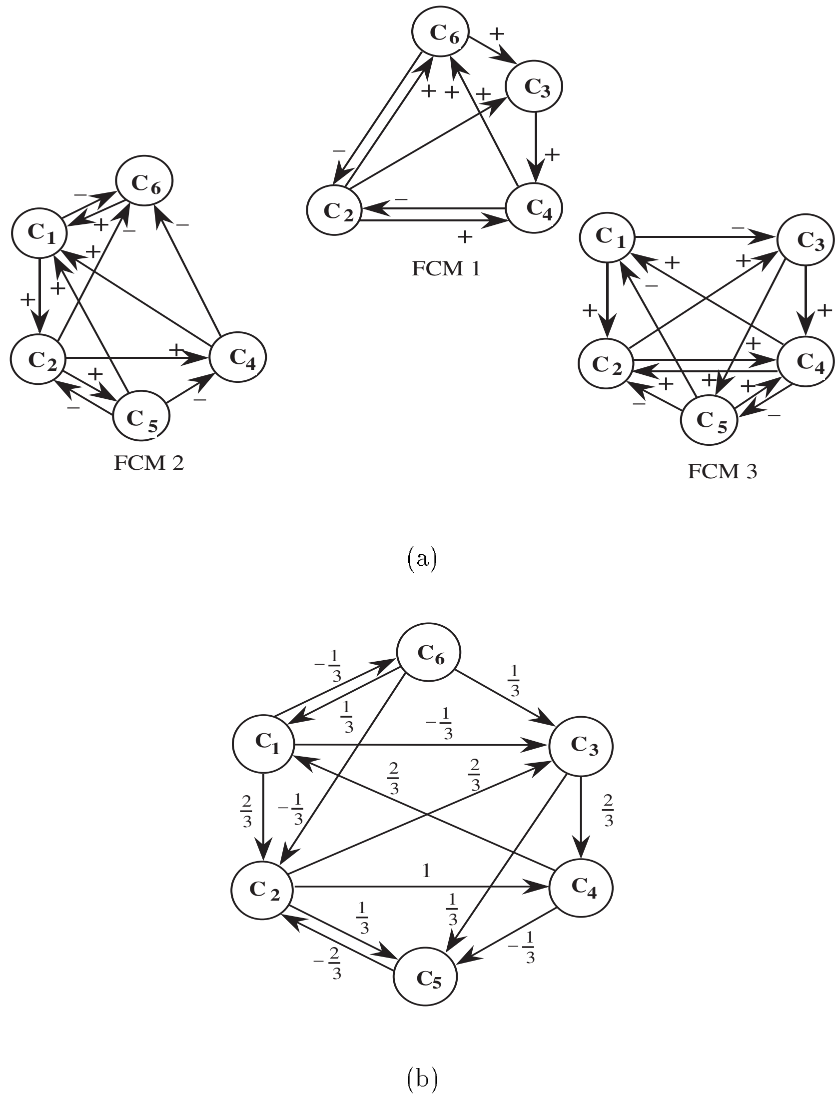

Figure 1.7 FCMs combine additively. (a) Three bivalent FCMs. (b) Augmented FCM. The augmented FCM takes the union of the causal concepts of the smaller FCMs and sums the augmented connection matrices as shown in (31)

The additive structure of combined FCMs also permits a Delphi [32] or questionaire approach to knowledge acquisition. In contrast an AI expert system [18] is a binary tree with graph search. Two or more trees need not combine to a tree. Combined FCMs tend to have feedback or closed loops and that precludes graph search with forward or backward "chaining." The strong law of large numbers [7] ensures that the knowledge estimate $\mathbf{F}$ in (30) improves with the expert sample size n if we view the experts as independent (unique) random knowledge sources with finite variance (bounded uncertainty) and identical distribution (same problem-domain focus). The sample FCM converges to the unknown population FCM as the number of experts grows.

The FCM sum (30) can lead to new limit cycles that are not found in the individual summed FCMs. The limit cycles in the FCMs shown in Figure 1.7a are given below. FCM 1 has the fixed point: (001101) and the 3 step limit cycles:

$$(000100) \rightarrow (000001) \rightarrow (001000) \rightarrow (000100)$$
$$(000101) \rightarrow (001001) \rightarrow (001100) \rightarrow (000101)$$

FCM 2 has a 3 step limit cycle:

$$(010000) \rightarrow (000110) \rightarrow (100000) \rightarrow (010000)$$

FCM 3 has one fixed point: (110100). The combined FCM has no fixed points and one 4 step limit cycle:

$$(100100) \rightarrow (110000) \rightarrow (011110) \rightarrow (101110) \rightarrow (100100).$$

This limit cycle is distinct from the limit cycles of each of the summed FCMs.

1.3.4 Nested FCMs

FCMs can bring goals and intentions to virtual worlds as they define dynamic physical and social environments. This can give the "common representation" needed for a virtual world [23]. The FCM can combine simple actions to model "intelligent" behavior [21],[24]. Each node in turn can control its own simple FCM in a *nested* FCM. Complex actions such as walking emerge from networks of simple reflexes. Nested simple FCMs can mimic this process as a net of finite state machines with binary limit cycles.

The output of a simple FCM is a binary limit cycle that describes actions or goalsKos88a. This holds even if the binary concept nodes change state asynchronously. Each output turns a function on or off as in a robotic neural net [21]. This output can control smaller FCMs or fuzzy control systems. These systems can drive visual, auditory, or tactile outputs of the virtual world. The FCM can control the temporal associations or timing cycles that structure virtual worlds. The FCM state vector drives the motion of each character as in a frame in a cartoon. Simple equations of motion can move each actor between the states.

FCM nesting extends to any number of fuzzy sets for the inputs. A concept can divide into smaller fuzzy sets or subconcepts. The edges or rules link the sets. This leads to a discrete multivalued output for each node. Enough nodes allow this system to approximate any continuous function [11] for signal functions of the form (29). The subconcepts Q_{ij} partition the fuzzy concept $\mathbf{C}_j$

$$C_j = \bigcup_{i=1}^{N_j} Q_{ij} \tag{32}$$

Figure 1.8 shows the concept of a SURVIVAL THREAT divided into subconcepts. Each subconcept is the degree of threat.

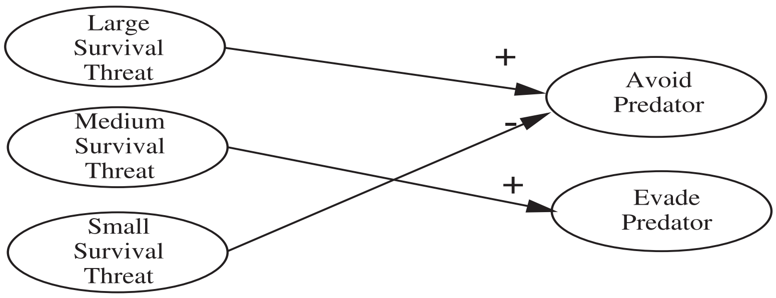

Figure 1.8 Subconcepts map to other concepts. This gives a more varied response.

The FCM edges or rules map one subconcept to another. These subconcept mappings form a fuzzy system or set of fuzzy if-then rules that map inputs to outputs. Each mapping is a fuzzy rule or state-space patch that links fuzzy sets. The patches cover the graph of some function in the input-output state space. The fuzzy system then averages the patches that overlap to give an approximation of a continuous function [9]. Figure 1.8 shows how subconcepts can map to different responses in the FCM. This gives a more varied response to changes in the virtual world.

1.4 Virtual Undersea World

Figure 1.9 shows a simple FCM for a virtual dolphin. It lists a causal web of goals and actions in the life of a dolphin [25]. The connection matrix $\mathbf{E}_D$ states these causal relations in numbers:

$$\mathbf{E}_D = \begin{array}{c} \\ D_1 \\ D_2 \\ D_3 \\ D_4 \\ D_5 \\ D_6 \\ D_7 \\ D_8 \\ D_9 \\ D_{10} \end{array} \begin{array}{c} \begin{array}{cccccccccc} D_1 & D_2 & D_3 & D_4 & D_5 & D_6 & D_7 & D_8 & D_9 & D_{10} \end{array} \\ \left[\begin{array}{cccccccccc} 0 & -1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 0 & -1 \\ 1 & 0 & -1 & 0 & 0 & -1 & -1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 1 & -1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & -1 & 0 & 1 \\ -1 & -1 & 1 & 0 & -1 & -1 & -1 & -1 & -1 & 0 \end{array}\right] \end{array}$$

The ith row lists the connection strength of the edges e_{ik} directed out from causal concept D_i and the ith column lists the edges e_{ki} directed into D_i . Row 9 shows how the concept SURVIVAL THREAT changes the other concepts. Column 9 shows the concepts that change SURVIVAL THREAT.

We can model the effect of a survival threat on the dolphin FCM as a sustained input to D_9. This means $D_9 = 1$ for all time t_k. $\mathbf{C}_0$ is the initial input state of the dolphin FCM:

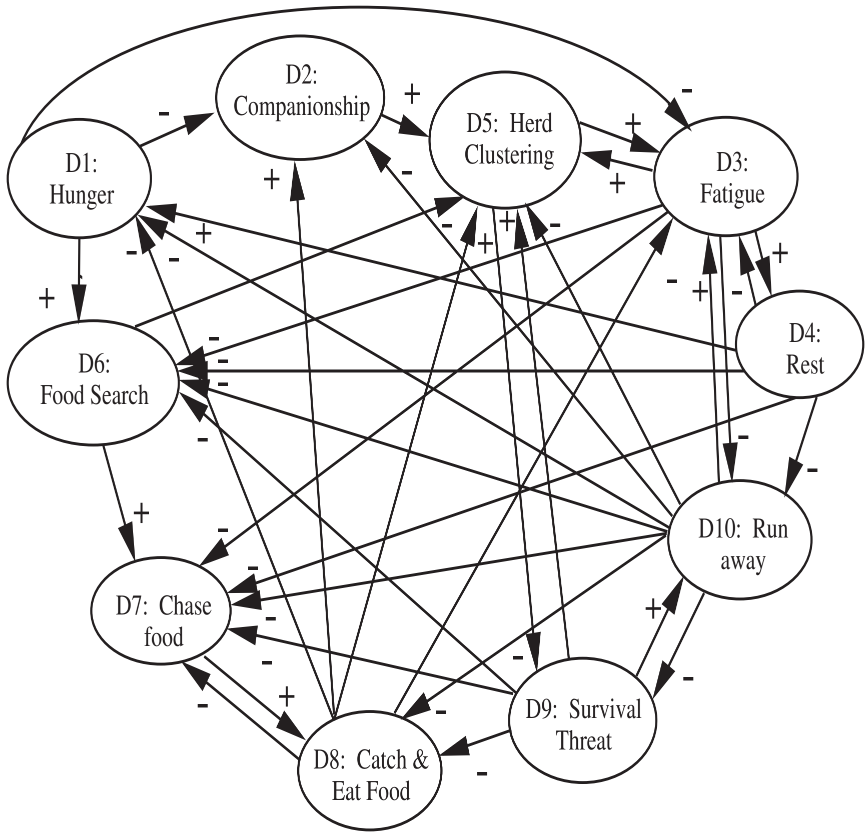

Figure 1.9 Trivalent fuzzy cognitive map for the control of a dolphin actor in a fuzzy virtual world. The rules or edges connect causal concepts in a signed connection matrix.

$$\mathbf{C}_0 \; = \; [\; 0\; 0\; 0\; 0\; 0\; 0\; 0\; 0\; 1\; 0].$$

Then

$$\mathbf{C}_0\mathbf{E}_D \; = [0\; 0\; 0\; 0\; 1\; -1\; -1\; -1\; 0\; 1] \rightarrow \mathbf{C}_1 \; = [0\; 0\; 0\; 0\; 1\; 0\; 0\; 0\; 1\; 1].$$

The arrow stands for a threshold operation with 1/2 as the threshold value. $\mathbf{C}_1$ keeps D9 on since we want to study the effect of a sustained threat. $\mathbf{C}_1$ shows that when threatened the dolphins cluster in a herd and flee the threat. The negative rules in the ninth row of $\mathbf{E}_D$ show that a threat to survival turns off other actions. The FCM converges to the limit cycle $\mathbf{C}_1 \rightarrow \mathbf{C}_2 \rightarrow \mathbf{C}_3 \rightarrow \mathbf{C}_4 \rightarrow \mathbf{C}_5 \rightarrow \mathbf{C}_1 \ldots$ if the threat lasts:

$$\mathbf{C}_1\mathbf{E}_D = [-1\; -1\; 2\; 0\; 0\; -2\; -2\; -2\; -2\; 1] \; \rightarrow \; \mathbf{C}_2 = [\; 0\; 0\; 1\; 0\; 0\; 0\; 0\; 0\; 1\; 1],$$

$$\mathbf{C}_2\mathbf{E}_D = [-1\ \ -1\ \ 1\ \ 1\ \ 1\ \ -3\ \ -3\ \ -2\ \ -1\ \ 0] \ \rightarrow\ \mathbf{C}_3 = [0\ \ 0\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0],$$

$$\mathbf{C}_3\mathbf{E}_D = [1\ \ 0\ \ 0\ \ 1\ \ 2\ \ -3\ \ -3\ \ -1\ \ -1\ \ -1] \ \rightarrow\ \mathbf{C}_4 = [\ 1\ \ 0\ \ 0\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0],$$

$$\mathbf{C}_4\mathbf{E}_D = [1\ \ -1\ \ -1\ \ 0\ \ 1\ \ -1\ \ -2\ \ -1\ \ -1\ \ 0] \ \rightarrow\ \mathbf{C}_5 = [\ 1\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0],$$

$$\mathbf{C}_5\mathbf{E}_D = [0\ \ -1\ \ 0\ \ 0\ \ 1\ \ 0\ \ -1\ \ -1\ \ -1\ \ 1] \ \rightarrow\ \mathbf{C}_1 = [0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 1\ \ 1],$$

Flight causes fatigue ($\mathbf{C}_2$). The dolphin herd stops and rests staying close together ($\mathbf{C}_3$). All the activity causes hunger ($\mathbf{C}_4$,$\mathbf{C}_5$). If the threat persists, they again try to flee ($\mathbf{C}_1$). A threat surpresses hunger. This limit cycle shows a "hidden" global pattern in the causal virtual world.

The FCM converges to the new limit cycle $\mathbf{C}_6 \rightarrow \mathbf{C}_7 \rightarrow \mathbf{C}_8 \rightarrow \mathbf{C}_9 \rightarrow \mathbf{C}_{10} \rightarrow \mathbf{C}_{11} \rightarrow \mathbf{C}_{12} \rightarrow \mathbf{C}_{13} \rightarrow \mathbf{C}_6 \rightarrow \ldots$ when the shark gives up the chase or eats a dolphin and the threat ends ($D_9 = 0$):
$\mathbf{C}_6 = [0\ \ 0\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0]$,

$$\begin{aligned}
\mathbf{C}_7\mathbf{E}_D &= [1\ \ 0\ \ 0\ \ 1\ \ 1\ \ -2\ \ -2\ \ 0\ \ -1\ \ -2] \ \rightarrow\ \mathbf{C}_7 \ = \ [\ 1\ \ 0\ \ 0\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0],\\
\mathbf{C}_8\mathbf{E}_D &= [1\ \ -1\ \ -1\ \ 0\ \ 0\ \ 0\ \ -1\ \ 0\ \ -1\ \ -1] \ \rightarrow\ \mathbf{C}_8 \ = \ [1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0],\\
\mathbf{C}_9\mathbf{E}_D &= [0\ \ -1\ \ -1\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0] \ \rightarrow\ \mathbf{C}_9 \ = \ [\ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0],\\
\mathbf{C}_{10}\mathbf{E}_D &= [0\ \ 0\ \ 0\ \ 0\ \ -1\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0] \ \rightarrow\ \mathbf{C}_{10} \ = \ [0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0],\\
\mathbf{C}_{11}\mathbf{E}_D &= [0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0] \ \rightarrow\ \mathbf{C}_{11} \ = \ [0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0],\\
\mathbf{C}_{12}\mathbf{E}_D &= [-1\ \ 1\ \ -1\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0] \ \rightarrow\ \mathbf{C}_{12} \ = \ [0\ \ 1\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0],\\
\mathbf{C}_{13}\mathbf{E}_D &= [0\ \ 0\ \ 1\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ -1\ \ 0] \ \rightarrow\ \mathbf{C}_{13} \ = \ [0\ \ 0\ \ 1\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0],\\
\mathbf{C}_{14}\mathbf{E}_D &= [0\ \ 0\ \ 1\ \ 1\ \ 1\ \ -1\ \ -1\ \ 0\ \ -1\ \ -1] \ \rightarrow\ \mathbf{C}_6 \ = \ [0\ \ 0\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0],
\end{aligned}$$

The dolphin herd rests from the previous chase ($\mathbf{C}_6, \mathbf{C}_7$). Then they begin a hunt of their own ($\mathbf{C}_9, \mathbf{C}_{10}$). They eat ($\mathbf{C}_{11}$) and then they socialize and rest ($\mathbf{C}_{12}, \mathbf{C}_{13}, \mathbf{C}_6$). This makes them hungry and the feeding cycle repeats.

1.4.1 Augmented Virtual World

Figure 1.10 shows an augmented FCM for an undersea virtual world. It combines fish school, shark, and dolphin herd FCMs with: $\mathbf{F} = \mathbf{F}_{fish} + \mathbf{F}_{shark} + \mathbf{F}_{dolphin}$. The new links among these FCMs are those of predator and prey where the larger eats the smaller. The actors chase, flee, and eat one another. A hungry shark chases the dolphins and that leads to the limit cycle ($\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3, \mathbf{C}_4$) above. Augmenting the FCM matrices gives a large but sparse FCM since the actors respond to each other in few ways. Figure 1.11 shows the connection matrix for the augmented FCM in Figure 1.10. The augmented FCM moves the actors in the virtual world. The binary output states of this FCM move the actors. Each FCM state maps to equations or function approximations for movement.

We used a simple update equation for position:

$$p\,(t_{n+1}) \ = \ p\,(t_n) + (t_{n+1} - t_n) \cdot v\,(t_n) \tag{33}$$

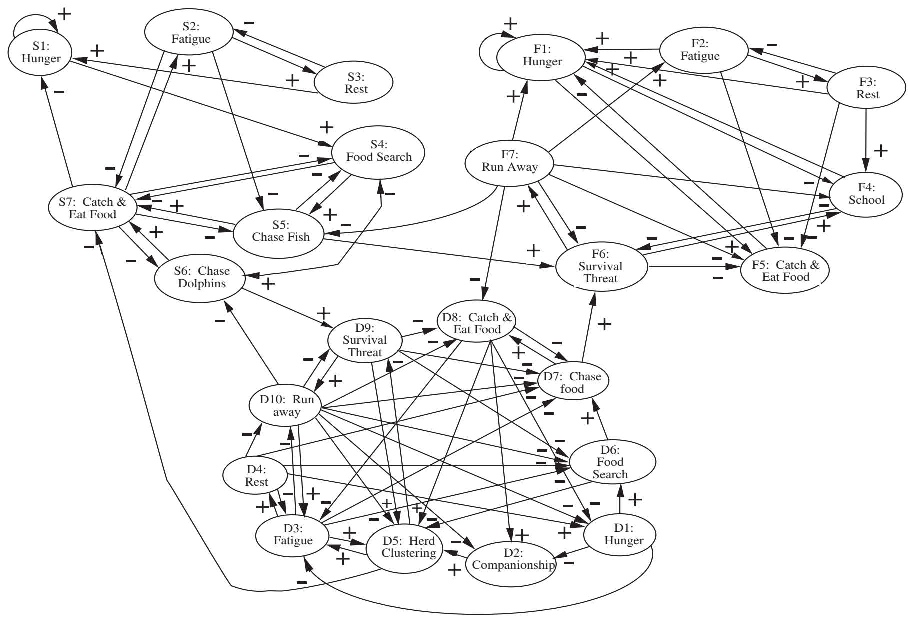

Figure 1.10 Augmented FCM for different actors in a virtual world. The actors interact through linked common causal concepts such as chasing food and avoiding a threat.

The velocity $v(t)$ does not change at time step Δt. The FCM finds the direction and magnitude of movement. The magnitude of the velocity depends on the FCM state. If the FCM state is "run away," then the velocity is FAST. If the FCM state is "rest," then the velocity is SLOW. The prey choose the direction that maximizes the distance from the predator. The predator chases the prey. When a predator searches for food it swims at random [26]. Each state moves the actors through the sea.

The FCM in Figure 1.10 encodes limit cycles between the actors. For example, if we start with a hungry shark and We set the causal link between concept S4: FOOD SEARCH and S6: CHASE DOLPHINS equal to zero to look at shark interactions with the fish school. Then the first state $\mathbf{C}_1$ is

$$\mathbf{C}_1 = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]$$

This vector gives a 7-step limit cycle after four transition steps:

$$\begin{aligned}\mathbf{C}_1\mathbf{E}_A &= [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0] \rightarrow \\ \mathbf{C}_2 &= [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0],\end{aligned}$$

$$\begin{aligned}\mathbf{C}_2\mathbf{E}_A &= [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ -1\ 0\ 0\ 0\ 0\ 0\ 0\ 0] \rightarrow \\ \mathbf{C}_3 &= [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0],\end{aligned}$$

$$\begin{aligned}\mathbf{C}_3\mathbf{E}_A &= [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0] \rightarrow \\ \mathbf{C}_4 &= [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0],\end{aligned}$$

	Dolphin										*Shark*							*Fish*						
	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}	S_1	S_2	S_3	S_4	S_5	S_6	S_7	F_1	F_2	F_3	F_4	F_5	F_6	F_7
D_1	0	−1	−1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D_2	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D_3	0	0	0	1	1	−1	−1	0	0	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D_4	1	0	−1	0	0	−1	−1	0	0	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D_5	0	0	1	0	0	0	0	0	−1	0	0	0	0	0	0	0	−1	0	0	0	0	0	0	0
D_6	0	0	0	0	−1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D_7	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
D_8	−1	1	−1	0	1	0	−1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D_9	0	0	0	0	1	−1	−1	−1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D_{10}	−1	−1	1	0	−1	−1	−1	−1	−1	0	0	0	0	0	0	−1	0	0	0	0	0	0	0	0
S_1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
S_2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	−1	0	−1	0	0	0	0	0	0	0
S_3	0	0	0	0	0	0	0	0	0	0	1	−1	0	0	0	0	0	0	0	0	0	0	0	0
S_4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	−1	0	0	0	0	0	0	0
S_5	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	1	0	0	0	0	0	1	0
S_6	0	0	0	0	0	0	0	0	1	0	0	0	0	−1	0	0	1	0	0	0	0	0	0	0
S_7	0	0	0	0	0	0	0	0	0	0	−1	1	0	−1	−1	−1	0	0	0	0	0	0	0	0
F_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	−1	1	0	0
F_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	−1	0	0
F_3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−1	0	1	−1	0	0
F_4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	−1	0
F_5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	0	0	0	0
F_6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	−1	0	1
F_7	0	0	0	0	0	0	0	−1	0	0	0	0	0	0	−1	0	0	1	1	0	−1	−1	−1	0

Figure 1.11 Augmented FCM connection matrix for the dolphin herd, fish school, and shark. Figure 1.10 shows the nodes and edges. The lines show the FCMs of the actors. The sparse region outside the lines shows the interaction space of the FCMs.

$$\mathbf{C}_4\mathbf{E}_A = [0\ \ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ \ -1\ 1\ 1] \rightarrow$$
$$\mathbf{C}_5 = [0\ \ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1],$$

$$\mathbf{C}_5\mathbf{E}_A = [0\ \ 0\ 0\ 0\ 0\ 0\ 0\ -1\ 0\ 0\ 0\ 1\ 0\ 0\ -2\ -1\ 0\ 2\ 1\ 0\ 0\ -2\ -2\ 1] \rightarrow$$
$$\mathbf{C}_6 = [0\ \ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 1],$$

$$\mathbf{C}_6\mathbf{E}_A = [0\ \ 0\ 0\ 0\ 0\ 0\ 0\ -1\ 0\ 0\ 0\ 0\ 1\ 0\ -2\ 0\ -1\ 3\ 1\ 1\ -2\ -1\ -1\ 0] \rightarrow$$
$$\mathbf{C}_7 = [0\ \ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 0],$$

$$\mathbf{C}_7\mathbf{E}_A = [0\ \ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ -1\ 0\ 0\ 0\ 0\ 0\ 3\ -1\ 1\ 0\ -1\ 0\ 0] \rightarrow$$
$$\mathbf{C}_8 = [0\ \ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0],$$

$$\mathbf{C}_8\mathbf{E}_A = [0\ \ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 2\ -1\ 0\ 0\ 0\ 0\ 0] \rightarrow$$
$$\mathbf{C}_9 = [0\ \ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0],$$

$$\mathbf{C}_9\mathbf{E}_A = [0\ \ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ -1\ 1\ 0\ 0\ -1\ 1\ 0\ 0] \rightarrow$$
$$\mathbf{C}_{10} = [0\ \ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0],$$

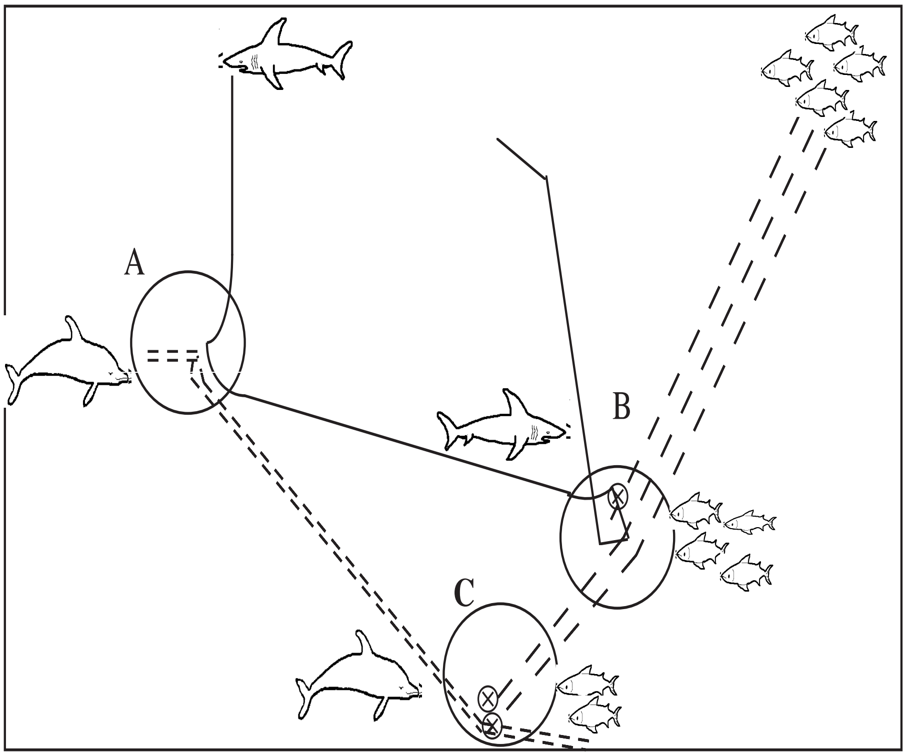

Figure 1.12 FCMs control the virtual world. The augmented FCM controls the actions of the actors. In event A the hungry shark forces the dolphin herd to run away. Each dashed line stands for a dolphin swim path. In event B the shark finds the fish and eats some. Each dashed line stands for the path of a fish in the school. The cross shows the shark eating a fish. In event C the fish run into the dolphins and suffer more losses. The solid lines are the dolphin paths. The dashes are the fish swim paths. The cross shows a dolphin eating a fish.

$$\begin{aligned} \mathbf{C}_{10}\mathbf{E}_A &= [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ -1\ 1\ 1\ 0] \rightarrow \\ \mathbf{C}_{11} &= [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0], \end{aligned}$$

$$\begin{aligned} \mathbf{C}_{11}\mathbf{E}_A &= [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ -1\ 0\ 0\ 1\ -1\ 1\ 1] \rightarrow \\ \mathbf{C}_5 &= [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1]. \end{aligned}$$

In this limit cycle a shark searches for food ($\mathbf{C}_1$,$\mathbf{C}_2$,$\mathbf{C}_3$). The shark finds some fish ($\mathbf{C}_4$), chases the fish ($\mathbf{C}_5$), and then eats some of the fish ($\mathbf{C}_6$). To avoid the shark most fish run away and then regroup as a school ($\mathbf{C}_5$,$\mathbf{C}_6$,$\mathbf{C}_7$). Then the fish rest and eat while the shark rests ($\mathbf{C}_8$,$\mathbf{C}_9$). In time the shark gets hungry again and searches for fish ($\mathbf{C}_{10}$,$\mathbf{C}_{11}$).

The result is a complex dance among the actors as they move in a 2-D ocean. Figure 1.12 shows these movements. The forcing function is a hungry shark ($\mathbf{C}_{11} = 1$). The shark encounters the dolphins who cluster and then flee the shark. The shark chases but cannot keep up. The shark still searches for food and finds the

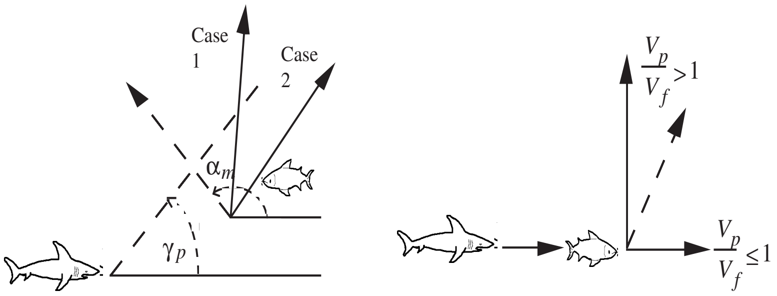

Figure 1.13 Fish change their behavior as the degree of threat changes. (a) The fish minimize time within the sighting angle of the predator. Case 1 shows the angle of escape when the fish swim faster. Case 2 shows the desired angle when the predator swims faster. (b) The fish maximize the distance between themselves and the predator to evade the predator. The fish swim straight ahead when the fish swim faster than the predator. The fish swim away at an angle if the predator swims faster.

fish. It catches a fish and then rests with its hunger sated. Meanwhile the hungry dolphins search for food and eat more fish. Each actor responds to the actions of the other.

1.4.2 Nested FCMs for Fish Schools

In a simple FCM the threat response concepts link as a rule: SURVIVAL THREAT implies RUN AWAY. Fish change their behavior as the degree of threat changes. This rule does not model the effects of different threats. For that we need a nested FCM or a fuzzy function approximator that links the threat degree to different responses. The size of the threat is a function of the size, speed, and attack angle of the predator [27]. A small threat leads to avoidance behavior. Figure 1.13a shows how fish avoid a predator. The fish move in direction α to maximize their distance from the predator[28]:

$$\cot\alpha = \cot\alpha_m + \frac{V_p}{V_f \sin\alpha_m} \tag{34}$$

V_p and V_f are the velocities of the predator and the fish. α_m is the angle that minimizes the time in terms of the predator's sighting angle γ_p:

$$\tan\alpha_m = -\cot\gamma_p \tag{35}$$

A large threat causes the fish to evade the predator. The fish try to maximize the minimum distance from the predator D_p [28]:

$$D_p^2 = [(X_o - V_p\,t) + V_f\,t\,cos\,\alpha]^2 + (V_f\ t\,sin\,\alpha)^2 \tag{36}$$

X_0 is the initial distance between predator and prey. α is the escape angle of the prey. V_p and $V_f$are the velocities of the predator and the fish. Figure 1.13b shows how fish evade a predator. A fuzzy system can approximate these responses using hand-picked rules or a neural-fuzzy learning [29]. These threat responses cause the "fountain effect" and the "burst effect" in fish schools [27] as each fish tries to

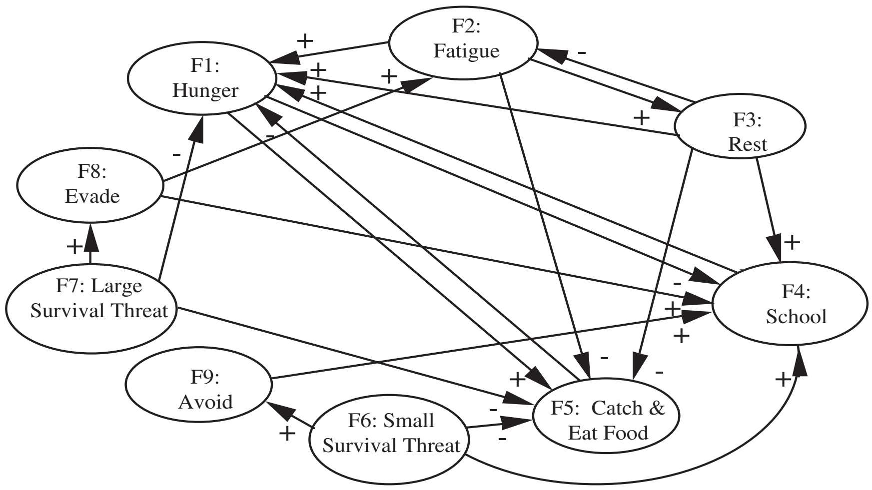

Figure 1.14 Example of a nested FCM. The concept of a survival threat divides into two subconcepts that each map to a different survival tactic.

increase its chances of survival. The fountain effect occurs when a predator moves towards a fish school and the school splits and flows around the predator. The school re-forms behind the predator. In the burst effect the school expands in the form of a sphere to evade the predator.

A small survival threat may be a slow-moving predator that either has not seen or decided to attack the fish. A large survival threat may be a fast predator such as a barracuda or shark that swims towards the center of the school. If we insert this new sub-FCM into the Fish FCM in Figure 1.10, we get the FCM in Figure 1.14. Different limit cycles appear for different degrees of threat. For a small threat (F_6) the fish avoid the predator (F_9) as they move out of the line-of-sight of the predator. Large threats (F_7) cause the fish to scatter quickly to evade the predator F_8. This leads to fatigue and rest (F_2 and F_3).

1.5 Adaptive Fuzzy Cognitive Maps

An adaptive FCM changes its causal web in time. The causal web learns from data. The causal edges or rules change in sign and magnitude. The additive scheme is a type of causal learning since it changes the FCM edge strengths. In general an edge e_{ij} changes with some first-order learning law:

$$\dot{e}_{ij} = f_{ij}(E, C) + g_{ij}(t) \tag{37}$$

Here g_{ij} is a forcing function. Data fires the concept nodes and in time this leaves a causal pattern in the edge. Causal learning is local in f_{ij}. It depends on just its own value and on the node signals that it connects:

$$\dot{e}_{ij} = f_{ij}\left(e_{ij}, C_i, C_j, \dot{C}_i, \dot{C}_j\right) + g_{ij}(t) \tag{38}$$

Correlation or Hebbian learning can encode some limit cycles in the FCMs or temporal associative memories (TAMs) [7]. It adds pairwise correlation matrices in (37). This method can only store a few patterns. Differential Hebbian learning encodes changes in a concept in equation (38). Both types of learning are local and light in computation.

To encode binary limit cycles in connection matrix $\mathbf{E}$ the TAM method sums the weighted correlation matrices between successive states [7]. To encode the limit cycle $\mathbf{C}_1 \rightarrow \mathbf{C}_2 \rightarrow \mathbf{C}_3 \rightarrow \mathbf{C}_1$ we first convert each binary state $\mathbf{C}_i$ into a bipolar state vector $\mathbf{X}_i$ by replacing each 0 with a -1. Then $\mathbf{E}$ is the weighted sum

$$\mathbf{E} = q_1 \mathbf{X}_1^T \mathbf{X}_2 + q_2 \mathbf{X}_2^T \mathbf{X}_3 + \ldots + q_{n-1} \mathbf{X}_{n-1}^T \mathbf{X}_n + q_n \mathbf{X}_n^T \mathbf{X}_1 \tag{39}$$

The length of the limit cycle should be less than the number of concepts. Else crosstalk can occur. Proper weighting of each correlation matrix pair can improve the encoding [30] and thus increase the FCM storage capacity. Correlation learning is a form of the unsupervised signal Hebbian learning law in neural networks [8]:

$$\dot{e}_{ij} = -e_{ij} + C_i(x_i) C_j(x_j) \tag{40}$$

A virtual world can encode an event sequence with (39) or (40). A simple chase cycle might be $\mathbf{C}_1 \rightarrow \mathbf{C}_2 \rightarrow \mathbf{C}_3$:

$$\mathbf{C}_1 = [1\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 1]$$

$$\mathbf{C}_2 = [1\ \ 0\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0]$$

$$\mathbf{C}_3 = [1\ \ 0\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 1\ \ 1]$$

Then (39) gives the FCM connection matrix $\mathbf{E}$ when $q_i = 1$ for all i:

$$\mathbf{E} = \begin{array}{c|rrrrrrrrrr|}
 & D_1 & D_2 & D_3 & D_4 & D_5 & D_6 & D_7 & D_8 & D_9 & D_{10} \\ \hline
D_1 & 3 & -3 & 1 & -1 & 1 & -3 & -3 & -3 & 1 & 1 \\
D_2 & -3 & 3 & -1 & 1 & -1 & 3 & 3 & 3 & -1 & -1 \\
D_3 & 1 & -1 & -1 & 1 & 3 & -1 & -1 & -1 & 3 & -1 \\
D_4 & -1 & 1 & -3 & -1 & 1 & 1 & 1 & 1 & 1 & 1 \\
D_5 & 1 & -1 & -1 & -3 & -1 & -1 & -1 & -1 & -1 & 3 \\
D_6 & -3 & 3 & -1 & 1 & -1 & 3 & 3 & 3 & -1 & -1 \\
D_7 & -3 & 3 & -1 & 1 & -1 & 3 & 3 & 3 & -1 & -1 \\
D_8 & -3 & 3 & -1 & 1 & -1 & 3 & 3 & 3 & -1 & -1 \\
D_9 & 1 & -1 & -1 & 3 & -1 & -1 & -1 & -1 & -1 & 3 \\
D_1 0 & 1 & -1 & 3 & 1 & -1 & -1 & -1 & -1 & -1 & -1 \\ \hline
\end{array}$$

Then

$$\begin{aligned}
\mathbf{C}_1\mathbf{E} &= [5 - 5\ 3\ 1\ 3 - 5 - 5 - 5\ 3 - 1] \rightarrow \mathbf{C}_2 = [1\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 0], \\
\mathbf{C}_2\mathbf{E} &= [5 - 5 - 5 - 7\ 3 - 5 - 5 - 5\ 3\ 7] \rightarrow \mathbf{C}_3 = [1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1], \\
\mathbf{C}_3\mathbf{E} &= [6 - 6\ 2 - 6 - 2 - 6 - 6 - 6 - 2\ 6] \rightarrow \mathbf{C}_1 = [1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1].
\end{aligned}$$

Correlation encoding treats negative and zero causal edges the same. It can encode "spurious" causal implications between concepts such as $e_{6,2} = 3$. This means searching for food causes a desire to socialize. Correlation encoding is a poor model of inferred causality. It says two concepts cause each other if they are on at the same time. Differential Hebbian learning encodes causal changes to avoid spurious causality. The concepts must move in the same or opposite directions to infer a causal link. They must come on and turn off at the same time or one must come on as the other turns off. Just being on does not lead to a new causal link. The patterns of turning on or off must correlate positively or negatively.

The differential Hebbian learning law [7] correlates concept changes or velocities:

$$\dot{e}_{ij} \;=\; -e_{ij} \;+\; \dot{C}_i\left(x_i\right)\dot{C}_j\left(x_j\right) \tag{41}$$

So $\dot{C}_i(x_i)\dot{C}_j(x_j) > 0$ iff concepts C_i and C_j move in the same direction. $\dot{C}_i(x_i)\dot{C}_j(x_j) < 0$ iff concepts C_i and C_j move in opposite directions. In this sense (41) learns patterns of causal change. The first-order structure of (41) implies that $e_{ij}(t)$ is an exponentially weighted average of paired (or lagged) changes. The most recent changes have the most weight. The *discrete* change $\Delta C_i(t) = C_i(t) - C_i(t-1)$ lies in {-1,0,1}. The discrete differential Hebbian learning can take the form

$$e_{ij}\left(t+1\right) = \begin{cases} e_{ij}\left(t\right) \;+\; c_t\left[\Delta C_i\left(x_i\right)\Delta C_j\left(x_j\right) \;-\; e_{ij}\left(t\right)\right] & if \quad \Delta C_i\left(x_i\right) \neq 0 \\ e_{ij}\left(t\right) & if \quad \Delta C_i\left(x_i\right) = 0 \end{cases} \tag{42}$$

Here c_t is a learning coefficient that decreases in time [20]. The sequence of learning coefficients $\{c_t\}$ should decrease slowly [8] in the sense of

$$\sum_{t=1}^{\infty} c_t = \infty$$

but not too slowly in the sense that

$$\sum_{t=1}^{\infty} c_t^2 < \infty.$$

In practice $c_t \approx \frac{1}{t}$. $\Delta C_i \Delta C_j > 0$ iff concepts C_i and C_j move in the same direction. $\Delta C_i \Delta C_j < 0$ iff concepts C_i and C_j move in opposite directions. **E** changes only if a concept changes. The changed edge slowly "forgets" the old causal changes in favor of the new ones. This causal law can learn higher-order causal relations if it correlates multiple cause changes with effect changes.

We used differential Hebbian learning to encode a feeding sequence and a chase sequence in a FCM. The concepts in the ith row learn only when $\Delta C_i(x_i)$ equals 1 or -1. We used

$$c_t\left(t_k\right) = 0.1\left[1 - \frac{t_k}{1.1N}\right].$$

The training data came from the rest, eat, play and the chase sequences in Section 1.4. This gave the $\mathbf{E}_D$:

	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}
D_1	−0.25	0.00	0.00	−0.24	−0.24	0.76	−0.51	0.00	0.00	0.00
D_2	0.00	−0.49	0.49	−0.51	0.00	0.00	0.00	0.00	0.00	0.00
D_3	−0.26	0.00	−0.25	1.00	0.75	0.00	0.00	0.00	0.00	0.00
D_4	1.00	0.00	−0.25	−0.25	−0.25	−0.50	0.00	0.00	0.00	0.00
D_5	0.51	−0.16	0.49	−0.34	−0.51	−0.33	0.00	0.00	0.00	−0.16
D_6	0.00	0.00	0.00	0.00	0.00	−0.49	1.00	−0.51	0.00	0.00
D_7	0.00	−0.51	0.00	0.00	−0.51	0.00	−0.49	1.00	0.00	0.00
D_8	0.00	1.00	−0.33	0.00	0.67	0.00	0.00	−0.67	0.00	0.00
D_9	0.00	0.00	−1.00	0.00	1.00	0.00	0.00	0.00	0.00	1.00
D_{10}	0.00	0.00	1.00	−0.51	−1.00	0.00	0.00	0.00	0.00	−0.49

This learned edge matrix $\mathbf{E}_D$ resembles the FCM matrix in Figure 1.9. The causal links it lacks between $\mathbf{D}_10$ and ($\mathbf{D}_6$,$\mathbf{D}_7$,$\mathbf{D}_8$) were not in the training set. The diagonal links terms for self-inhibition of each concept. This occurs since each concept is on for one cycle before the matrix transitions to the next state. The hunger input $\mathbf{CL}_0 = [\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]$ with a threshold of 0.51 now leads to the limit cycle:

$$\mathbf{CL}_0\mathbf{E}_D = [-0.25\ 0.00\ 0.00\ -0.24\ -0.24\ 0.76\ -0.51\ 0.00\ 0.00\ 0.00] \rightarrow$$

$$\mathbf{CL}_1 = [0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0],$$

$$\mathbf{CL}_1\mathbf{E}_D = [0.00\ 0.00\ 0.00\ 0.00\ 0.00\ -0.49\ 1.00\ -0.51\ 0.00\ 0.00] \rightarrow$$

$$\mathbf{CL}_2 = [0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0],$$

$$\mathbf{CL}_2\mathbf{E}_D = [0.00\ -0.51\ 0.00\ 0.00\ -0.51\ 0.00\ -0.49\ 1.00\ 0.00\ 0.00] \rightarrow$$

$$\mathbf{CL}_3 = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0],$$

$$\mathbf{CL}_3\mathbf{E}_D = [0.00\ 1.00\ -0.33\ 0.00\ 0.67\ 0.00\ 0.00\ -0.67\ 0.00\ 0.00] \rightarrow$$

$$\mathbf{CL}_4 = [0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0],$$

$$\mathbf{CL}_4\mathbf{E}_D = [0.51\ -0.65\ 0.98\ -0.85\ -0.51\ -0.33\ 0.00\ 0.00\ 0.00\ -0.16] \rightarrow$$

$$\mathbf{CL}_5 = [0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0],$$

$$\mathbf{CL}_5\mathbf{E}_D = [-0.26\ 0.00\ -0.25\ 1.00\ 0.75\ 0.00\ 0.00\ 0.00\ 0.00\ 0.00] \rightarrow$$

$$\mathbf{CL}_6 = [0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0],$$

$$\mathbf{CL}_6\mathbf{E}_D = [1.51\ -0.16\ 0.25\ -0.59\ -0.76\ -0.83\ 0.00\ 0.00\ 0.00\ -0.16] \rightarrow$$

$$\mathbf{CL}_1 = [1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0],$$

Figure 1.15(a) shows the hand-designed limit cycle from the previous section. Figure 1.15(b) shows the limit cycle from FCM found with differential Hebbian learning. The DHL limit cycle is one step shorter. Both FCMs have just one limit cycle and the null fixed point in the space of 2^{10} binary state vectors. The value of $\mathbf{E}_{D5}$ does not change over 2 intervals. The learning law in (42) learns only if there is a change in the node.

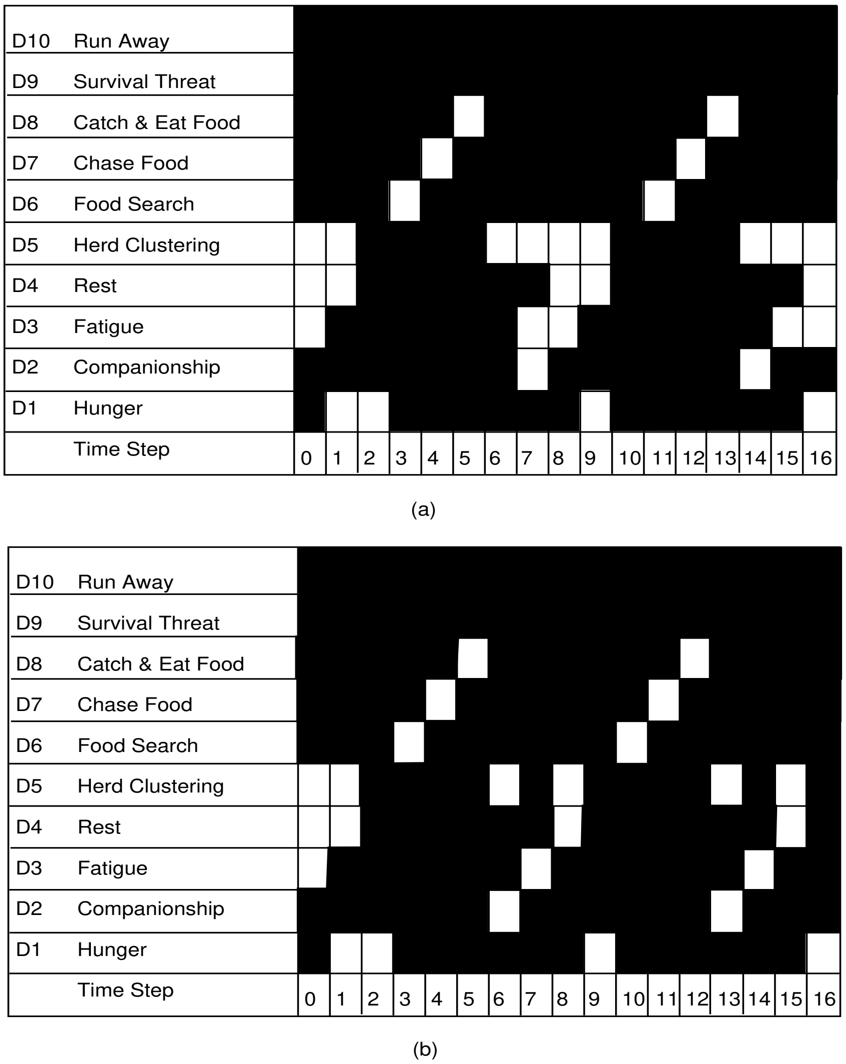

Figure 1.15 Limit cycle comparison between the hand-designed system and the FCM found with differential Hebbian learning. Each column is a binary state vector. (a) Rest, feed, play, rest limit cycle for the FCM in Figure 1.9. (b) Limit cycle for the FCM found with (42).

1.6 Conclusions

Fuzzy cognitive maps can model the causal web of a virtual world. The FCM can control its local and global nonlinear behavior. The local fuzzy rules or edges and the fuzzy concepts they connect model the causal links within and between events. The global FCM nonlinear dynamics give the virtual world an "arrow of time." A user can change these dynamics at will and thus change the causal processes in the virtual world. FCMs let experts and users choose a causal web by drawing causal pictures instead of by stating equations.

FCMs can also help visualize data. They show how variables relate to one another in the causal web. The FCM output states can guide a cartoon of the virtual world as shown in Figure 1.16. This cartoon shows the dolphin chase, rest, eat sequence described earlier. The cartoon animates the FCM dynamics as the system trajectory moves through the FCM state space. This can apply to models in economics, medicine, history, and politics [31] where the social and causal web can change in complex ways that may arise from changing the sign or magnitude of a single FCM causal rule or edge.

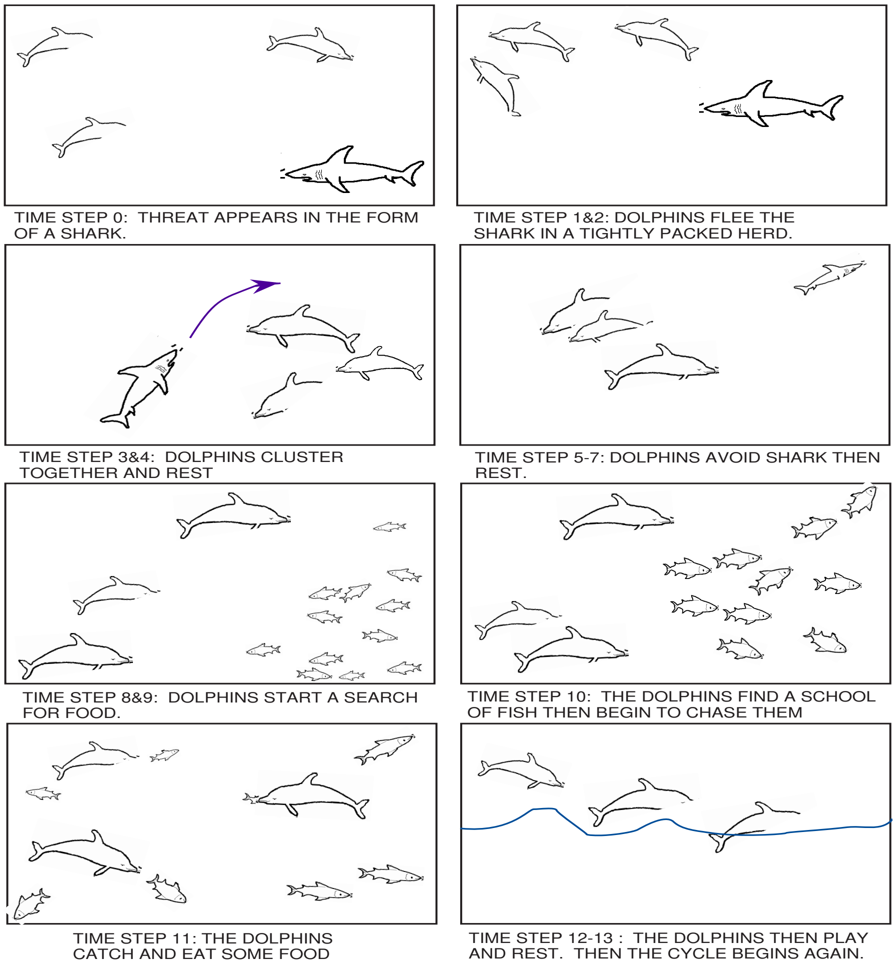

Figure 1.16 The FCM output states can guide a cartoon of the virtual world. This cartoon shows the dolphin chase, rest, eat sequence described in section 3. The cartoon animates the FCM dynamics as the system trajectory moves through the FCM state space.

The additive structure of combined FCMs permits a Delphi [32] or questionnaire approach to knowledge acquisition. These new causal webs can change an adaptive FCM that learns its causal web as neural-like learning laws process time-series data. Experts can add their FCM matrices to the adaptive FCM to initialize or guide the learning. Such a causal web can learn the user's values and action habits and perhaps can test them or train them.

More complex FCMs have more complex dynamics and can model more complex virtual worlds. Each concept node can fire on its own time scale and fire in its own nonlinear way. The causal edge flows or rules can have their own time scales too and may increase or decrease the causal flow through them in nonlinear ways. This behavior does not fit in a simple FCM with threshold concepts and constant edge weights.

A FCM can model these complex virtual worlds if it uses more nonlinear math to change its nodes and edges. The price paid may be a chaotic virtual world with unknown equilibrium behavior. Some users may want this to add novelty to their virtual world or to make it more exciting. A user might choose a virtual world that is mildly nonlinear and has periodic equilibria. At the other extreme the user might choose a virtual world that is so wildly nonlinear it has only aperiodic equilibria. Think of a virtual game of tennis or raquetball where the gravitational potential changes at will or at random.

Fuzziness and nonlinearity are design parameters for a virtual world. They may give a better model of a real process.

REFERENCES

[1] M. Krueger, *Artificial Reality II*, Second ed: Addison-Wesley, 1991.

[2] W. Gibson, *Neuromancer*, New York: Ace Books, 1984.

[3] R. A. Brown, *Fluid Mechanics of the Atmosphere*, New York: Academic Press, 1991.

[4] J. J. Craig, *Introduction to Robotics*, Reading, MA: Addison-Wesley, 1986.

[5] E. Ackerman, L. Gatewood, J. Rosevear and G. Molnar, "Blood Glucose Regulation and Diabetes," in *Concepts and Models of Biomathematics*, F. Heinmets, Ed.: Marcel Dekker, 1969.

[6] B. Kosko, "Fuzzy Cognitive Maps," *International Journal Man-Machine Studies*, Vol. 24, No. , pp. 65-75, 1986.

[7] B. Kosko, "Hidden Patterns in Combined and Adaptive Knowledge Networks," *International Journal of Approximate Reasoning*, Vol. 2, No. , pp. 337-393, 1988.

[8] B. Kosko, *Neural Networks and Fuzzy Systems* Englewood Cliffs: Prentice Hall, 1992.

[9] B. Kosko, "Fuzzy Systems as Universal Approximators," *IEEE Transactions on Computers*, Vol. 43, No. 11, November, pp. 1329-1333, 1994.

[10] H. T. Nguyen, "On Random Sets and Belief Functions," *Journal of Mathematical Analysis and Applications*, Vol. 65, No. 1-2, pp. 531-542, 1978.

[11] K. Hornik, M. Stinchcombe and H. White, "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks*, Vol. 2, No. , pp. 359 - 366, 1989.

[12] F. A. Watkins, "Fuzzy Engineering,"*Ph.D. Thesis*, University of California at Irvine, 1994.

[13] L. Wang and J. M. Mendel, "Fuzzy Basis Functions, Universal Approximation, and Orthogonal Least-Squares Learning," *IEEE Transactions on Neural Networks*, Vol. 3, No. 5, September, pp. 807 - 814, 1992.

[14] D. F. Specht, "A General Regression Neural Network," *IEEE Transactions on Neural Networks*, Vol. 2, No. 6, November, pp. 569-576, 1991.

[15] B. Kosko, "Optimal Fuzzy Rules Cover Extrema," *International Journal of Intelligent Systems*, Vol. 10, No. 2, pp. 249-255, 1995.

[16] F. A. Watkins, "The Representation Problem for Additive Fuzzy Systems," *Proceedings of the the 1995 IEEE International Conference on Fuzzy Systems (IEEE FUZZ-95)*,Vol. I, pp. 117-122,1995.

[17] J. A. Dickerson and B. Kosko, "Virtual Worlds as Fuzzy Cognitive Maps," *Presence*, Vol. 3, No. 2, Spring, pp. 173-189, 1994.

[18] P. H. Winston, *Artificial Intelligence*, Second ed. Reading, MA: Addison-Wesley, 1984.

[19] W. R. Taber and M. Siegel, "Estimation of Expert Weights with Fuzzy Cognitive Maps," *Proceedings of the 1st IEEE International Conference on Neural Networks (ICNN-87)*, San Diego,Vol. II, pp. 319-325,1987.

[20] B. Kosko, "Bidirectional Associative Memories," *IEEE Transactions Systems, Man, and Cybernetics*, Vol. 18, No. 1, pp. 49-60, 1988.

[21] R. A. Brooks, "A Robot that Walks: Emergent Behaviors from a Carefully Evolved Network," *Neural Computation*, Vol. 1, No. 2, pp. 253-262, 1989.

[22] S. Grossberg, *Studies of Mind and Brain*, Boston: Reidel, 1982.

[23] N. I. Badler, B. L. Webber, J. Kalita and J. Esakov, "Animation from Instructions," in *Making Them Move: Mechanics, Control, and Animation of Articulated Figures*, N. I. Badler, B. A. Barsky and D. Zeltzer, Eds. San Mateo, CA: Morgan Kaufmann, pp. 51-98, 1991.

[24] J. H. Connell, *Minimalist Mobile Robotics: A Colony-style Architecture for an Artificial Creature* Academic Press, Harcourt Brace Jovanovich, 1990.

[25] S. H. Shane, "Comparison of Bottlenose Dolphin Behavior in Texas and Florida, with a Critique of Methods for Studying Dolphin Behavior," in *The Bottlenose Dolphin*, S. Leatherwood and R. R. Reeves, Eds.: Academic Press, pp. 541-558, 1990.

[26] B. O. Koopman, *Search and Screening*, New York: Pergamon Press, 1980.

[27] B. L. Partridge, "The Structure and Function of Fish Schools," *Scientific American*, Vol. 246, No. 6, pp. 114-123, 1982.

[28] D. Weihs and W. P. W., "Optimal Avoidance and Evasion Tactics in Predator-Prey Interactions," *Journal of Theoretical Biology*, Vol. 106, No. , pp. 189-206, 1984.

[29] J. A. Dickerson, "Fuzzy Function Approximation with Ellipsoidal Rules," *Ph.D. Thesis*, University of Southern California, 1993.

[30] Y. F. Wang, J. B. Cruz and J. H. Mulligan, "Guaranteed Recall of All Training Pairs for Bidirectional Associative Memory," *IEEE Transactions on Neural Networks*, Vol. 2, No. 6, pp. 559-567, 1991.

[31] W. R. Taber, "Knowledge Processing with Fuzzy Cognitive Maps," *Expert Systems with Applications*, Vol. 2, No. 1, pp. 83-87, 1991.

[32] J. P. Martino, *Technological Forecasting for Decisionmaking*, American Elsevier, 1972.

[33] J. A. Dickerson and B. Kosko, "Fuzzy Function Approximation with Supervised Ellipsoidal Learning," *Proceedings of the World Conference on Neural Networks (WCNN '93)*, Portland, OR,Vol. II, pp. 9-17,1993.

[34] J. A. Dickerson and B. Kosko, "Fuzzy Function Learning with Covariance Ellipsoids," *Proceedings of the IEEE International Conference on Neural Networks (IEEE ICNN-93)*, San Francisco, pp. 1162-1167,1993.

[35] J. A. Dickerson and B. Kosko, "Fuzzy Function Approximation with Ellipsoidal Rules," *IEEE Transactions on Systems*, Man, and Cybernetics, No. August, pp. To Appear, 1996.

[36] B. Kosko, "Stochastic Competitive Learning," *IEEE Transactions on Neural Networks*, Vol. 2, No. 5, pp. 522-529, 1991.

[37] H. M. Kim and B. Kosko, "Fuzzy Prediction and Filtering in Impulsive Noise," *Fuzzy Sets and Systems*, Vol. 77, No. 1, pp. 15-33, 1996.

A Proof of the Fuzzy Approximation Theorem

Fuzzy Approximation Theorem An additive fuzzy system uniformly approximates f: $X \to Y$ if X is compact and f is continuous.

Proof:

Pick any small constant $\epsilon > 0$. We must show that $|F(x) - f(x)| < \epsilon$ for all $x \in X$. X is a compact subset of R_n. $F(x)$ is the centroidal output (1) of the additive fuzzy system F. Continuity of f on compact X gives uniform continuity. So there is a fixed distance δ such that, for all x and z in X, $|f(x) - f(z)| < \epsilon/4$ if $|x - z| < \delta$. (Replace δ by δ/n for any L^p space with $p > 1$.) We can construct a set of open cubes $M_1, \cdots, M_m$ that cover X and that have ordered overlap in their n coordinates so that each cube corner lies at the midpoint c_j of its neighbors M_j. Pick symmetric output fuzzy sets B_j centered on $f(c_j)$. So the centroid of B_j is $f(c_j)$.

Pick $u \in X$. Then by construction u lies in at most 2^j overlapping open cubes M_j. Pick any w in the same set of cubes. If $u \in M_j$ and $w \in M_k$, then for all $v \in M_j \cap M_k$: $|u - v| < \delta$ and $|v - w| < \delta$. Uniform continuity implies that $|f(u) - f(w)| \leq |f(u) - f(v)| + |f(v) - f(w)| < \frac{\epsilon}{2}$. So for cube centers c_j and c_k, $|f(c_j) - f(c_k)| < \frac{\epsilon}{2}$.

Pick $x \in X$. Then x too lies in at most 2^j open cubes with centers c_j and $|f(c_j) - f(x)| < \frac{\epsilon}{2}$. Along the kth coordinate of the range space R^p the kth component of the additive system centroid $F(x)$ lies on or between the kth components of the centroids of the B_j sets. So, since $|f(c_j) - f(c_k)| < \frac{\epsilon}{2}$ for all $f(c_j)$, $|F(x) - f(c_j)| < \frac{\epsilon}{2}$. Then

$$|F(x) - f(x)| \leq |F(x) - f(c_j)| + |f(c_j) - f(x)| < \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon$$

Q.E.D.

B Learning in SAMs: Unsupervised Clustering and Supervised Gradient Descent

A fuzzy system learns if and only if its rule patches move or change shape in the input-output product space $X \times Y$. Learning can change the centers or widths of triangle or trapezoidal sets. These changing sets then change the shape or position of the Cartesian rule patches built out of them. The mean-value theorem and the calculus of variations show [15] that optimal lone rules cover the extrema or bumps of the approximand. Good learning schemes [33, 34, 35] tend to quickly move rules patches to these bumps and then move extra rule patches between them as the rule budget allows. Hybrid schemes use unsupervised clustering to learn the first set of fuzzy rule patches in position and number and to initialize gradient descent in supervised learning.

Learning changes system parameters with data. Unsupervised learning amounts to blind clustering in the system product space $X \times Y$ to learn and tune the m fuzzy rules or the sets that compose them. Then k quantization vectors $q_j \in X \times Y$ move in the product space to filter or approximate the stream of incoming data pairs $(x(t), y(t))$ or the concatenated data points $z(t) = [x(t)|y(t)]^T$. The simplest form of such *product space clustering* [8] centers a rule patch at each data point and thus puts $k = m$. In general both the data and the quantizing vectors greatly outnumber the rules and so $k >> m$.

A natural way to grow and tune rules is to identify a rule patch with the uncertainty ellipsoid [33, 34, 35] that forms around each quantizing vector q_j from the inverse of its positive definite covariance matrix K_j. Then sparse or noisy data grows a patch larger and thus a less certain rule than does denser or less noisy data. Unsupervised competitive learning [8] can learn these ellipsoidal rules in three steps:

$$\|z(t) - q_j(t)\| = \min(\|z(t) - q_1(t)\|, \ldots, \|z(t) - q_k(t)\|) \tag{B.1}$$

$$q_i(t+1) = \begin{cases} q_j(t) + \mu_t[z(t) - q_j(t)] & \text{if } i = j \\ q_i(t) & \text{if } i \neq j \end{cases} \tag{B.2}$$

$$K_i(t+1) = \begin{cases} K_j(t) + v_t[(z(t) - q_j(t))^T(z(t) - q_j(t)) - K_j(t)] & \text{if } i = j \\ K_i(t) & \text{if } i \neq j \end{cases} \tag{B.3}$$

for the Euclidean norm $\|z\|^2 = z_1^2 + \cdots + z_{n+p}^2$.

The first step (B.1) is the competitive step [36]. It picks the nearest quantizing vector q_j to the incoming data vector $z(t)$ and ignores the rest. Some schemes may count nearby vectors as lying in the winning subset. We used just one winner per datum. This correlation matching approximates the competitive dynamics of nonlinear neural networks. The second step updates the winning quantization or "synaptic" vector and drives it toward the centroid of the sampled data pattern class [36]. The third step updates the covariance matrix of the winning quantization vector. We initialize the quantization vector with sample data ($q_i(0) = z(i)$) to avoid skewed groupings and to initialize the covariance matrix with small positive numbers on its diagonal to keep it positive definite. Projection schemes [33, 34, 35] can then convert the ellipsoids into fuzzy sets along each coordinate of the input-output space. Other schemes can use the unfactored joint set function directly[37]. Supervised learning can also tune the eigenvalue parameters of the rule ellipsoids.

The sequences of learning coefficients $\{\mu_t\}$ and $\{v_t\}$ should decrease slowly [8] in the sense of $\sum_{t=1}^{\infty} \mu_t = \infty$ but not too slowly in the sense of $\sum_{t=1}^{\infty} \mu_t^2 < \infty$. In practice $\mu_t \approx \frac{1}{t}$. The covariance coefficients obey a like constraint as in our choice of $v_t = 0.2[1 - \frac{t}{1.2N}]$ where N is the total number of data points. The supervised learning schemes below also use a similar sequence of decreasing learning coefficients.

Supervised learning changes SAM parameters with error data. The error at each time t is the desired system output minus the actual SAM output: $\varepsilon_t = d_t - F(x_t)$. Unsupervised learning uses the blind data point $z(t)$ instead of the desired or labeled value d_t. The teacher or supervisor supervises the learning process by giving the desired value d_t at each training time t. Most supervised learning schemes perform stochastic gradient descent on the squared error and do so through iterated use of the chain rule of differential calculus.

Supervised gradient descent can learn or tune SAM systems [34, 35], by changing the rule weights w_j in (B.4), the then-part volumes V_j, the then-part centroids c_j, or parameters of the if-part set functions a_j. The rule weight w_j enters the ratio form of the general SAM system

$$F(x) = \frac{\sum_{j=1}^{m} w_j \; a_j(x) \; V_j \; c_j}{\sum_{j=1}^{m} w_j \; a_j(x) \; V_j} \tag{B.4}$$

in the same way as does the then-part volume V_j in the SAM Theorem. Both cancel from (B.4) if they have the same value–if $w_1 = \cdots = w_m > 0$ or if $V_1 = \cdots = V_m > 0$. So both have the same learning law if we replace the nonzero weight w_j with the nonzero volume V_j [35]:

$$\begin{aligned} w_j(t+1) &= w_j(t) - \mu_t \; \frac{\partial E_t}{\partial w_j} && \text{(B.5)} \\ &= w_j(t) - \mu_t \; \frac{\partial E_t}{\partial F} \frac{\partial F}{\partial w_j} && \text{(B.6)} \\ &= w_j(t) + \mu_t \; \varepsilon_t \; \frac{p_j(x_t)}{w_j(t)}[c_j - F(x_t)] && \text{(B.7)} \end{aligned}$$

for instantaneous squared error $E_t = \frac{1}{2}(d_t - F(x_t))^2$ with desired-minus-actual error $\varepsilon_t = d_t - F(x_t)$. We include the rule weights here for completeness. Our fuzzy systems were unweighted and thus used $w_1 = \cdots = w_m > 0$. The volumes then change in the same way if they are independent of the weights (which they may not be in some ellipsoidal learning schemes):

$$\begin{aligned} V_j(t+1) &= V_j(t) - \mu_t \; \frac{\partial E_t}{\partial V_j} && \text{(B.8)} \\ &= V_j(t) + \mu_t \; \varepsilon_t \; \frac{p_j(x_t)}{V_j(t)}[c_j - F(x_t)] && \text{(B.9)} \end{aligned}$$

The learning law (B.7) follows since $\frac{\partial E_t}{\partial F} = -\varepsilon$ and since

$$\frac{\partial F}{\partial w_j} = \frac{a_j(x) \; V_j \; c_j \sum_{i=1}^{m} w_i \; a_i(x) \; V_i - a_j(x) \; V_j \sum_{i=1}^{m} w_i \; a_i(x) \; V_i \; c_i}{(\sum_{i=1}^{m} w_i \; a_i(x) \; V_i)^2} \tag{B.10}$$

$$= \frac{w_j \; a_j(x) \; V_j}{w_j \sum_{i=1}^{m} w_i \; a_i(x) \; V_i} \left[\frac{c_j \sum_{i=1}^{m} w_i \; a_i(x) \; V_i}{\sum_{i=1}^{m} w_i \; a_i(x) \; V_i} - \frac{\sum_{i=1}^{m} w_i \; a_i(x) \; V_i \; c_i}{\sum_{i=1}^{m} w_i \; a_i(x) \; V_i} \right] \tag{B.11}$$

$$= \frac{p_j(x)}{w_j}[c_j - F(x)] \tag{B.12}$$

from the SAM Theorem.

The centroid c_j in the SAM Theorem has the simplest learning law:

$$c_j(t+1) = c_j(t) - \mu_t \frac{\partial E_t}{\partial F} \frac{\partial F}{\partial c_j} \quad \text{(B.13)}$$

$$= c_j(t) + \mu_t \, \varepsilon_t \, p_j(x_t). \quad \text{(B.14)}$$

So the terms w_j, V_j, and c_j do not change when $p_j \approx 0$ and thus when the$j$th if-part set barely fires: $a_j(x_t) \approx 0$.

Tuning the if-part sets involves more computation since the update law contains an extra partial derivative. Suppose that the if-part set function a_j is a function of l parameters: $a_j = a_j(m_j^1, \ldots, m_j^l)$. Then we can update each parameter with

$$m_j^k(t+1) = m_j^k(t) - \mu_t \frac{\partial E_t}{\partial F} \frac{\partial F}{\partial a_j} \frac{\partial a_j}{\partial m_j^k} \quad \text{(B.15)}$$

$$= m_j^k(t) + \mu_t \, \varepsilon_t \, \frac{p_j(x_t)}{a_j(x_t)} [c_j - F(x_t)] \frac{\partial a_j}{\partial m_j^k}. \quad \text{(B.16)}$$

Exponential if-part set functions can reduce the learning complexity. They have the form $a_j = e^{f_j(m_j^1, \ldots, m_j^l)}$ and obey $\frac{\partial a_j}{\partial m_j^k} = a_j \frac{\partial f_j(m_j^1, \ldots, m_j^l)}{\partial m_j^k}$. Then the parameter update (B.15) simplifies to

$$m_j^k(t+1) = m_j^k(t) + \mu_t \, \varepsilon_t \, p_j(x_t)[c_j - F(x_t)] \frac{\partial f_j}{\partial m_j^k} \quad . \quad \text{(B.17)}$$

This can arise for independent exponential or Gaussian sets $a_j(x) = \prod_{i=1}^{n} exp\{f_j^i(x_i)\} = exp\{\sum_{i=1}^{n} f_j^i(x_i)\} = exp\{f_j(x)\}$. The exponential set function

$$a_j(x) = exp\{\sum_{i=1}^{n} u_j^i (v_j^i - x_i)\} \quad \text{(B.18)}$$

has partial derivatives $\frac{\partial f_j}{\partial u_j^k} = v_j^k - x_k(t)$ and $\frac{\partial f_j}{\partial v_j^k} = u_j^k$.

The Gaussian set function

$$a_j(x) = exp\{-\frac{1}{2} \sum_{i=1}^{n} (\frac{x_i - m_j^i}{\sigma_j^i})^2\} \quad \text{(B.19)}$$

has mean partial derivative $\frac{\partial f_j}{\partial m_j^k} = \frac{x_k - m_j^k}{(\sigma_j^k)^2}$ and variance partial derivative $\frac{\partial f_j}{\partial \sigma_j^k} = \frac{(x_k - m_j^k)^2}{(\sigma_j^k)^3}$. Such Gaussian set functions reduce the SAM model to Specht's [14] radial basis function network. We can use the smooth update law (B.17) to update non-differentiable triangles or trapezoids or other sets by viewing their centers and widths as the Gaussian means and variances.