(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
26 April 2001 (26.04.2001)

PCT

(10) International Publication Number
WO 01/29766 A2

(51) International Patent Classification[7]: G06N 3/063

(21) International Application Number: PCT/GB00/03957

(22) International Filing Date: 16 October 2000 (16.10.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
9924310.7 15 October 1999 (15.10.1999) GB

(71) Applicant *(for all designated States except US)*: **THE VICTORIA UNIVERSITY OF MANCHESTER** [GB/GB]; Oxford Road, Manchester M13 9PL (GB).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: **FURBER, Stephen, Byrun** [GB/GB]; 1A Gorsey Road, Wilmslow, Cheshire SK9 5DU (GB).

(74) Agent: **ROBERTS, Peter, David**; Marks & Clerk, 83-85 Mosley Street, Manchester M2 3LG (GB).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *Without international search report and to be republished upon receipt of that report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 01/29766 A2

(54) Title: NEURAL NETWORK COMPONENT

(57) Abstract: A neural network component, the component comprising a plurality of inputs, at least one processing element, at least one output, and a digital memory storing values at addresses respectively corresponding to the at least one processing element, wherein the at least one processing element is arranged to receive a value from the digital memory in response to an input signal, and is instructed to execute one of a plurality of operations by the value that is received from the digital memory.

NEURAL NETWORK COMPONENT

The present invention relates to a neural network component.

Neural networks are biologically-inspired computation architectures for processing information. They are increasingly used to solve problems that are difficult to solve with conventional algorithmic programs running on conventional stored-program computers. These typically are pattern-matching problems such as stock market forecasting, image recognition and speech recognition. Some neural network applications are commercially significant. For example, the touch-pads of many lap-top personal computers use a neural network to improve reliability (neural systems are relatively noise-insensitive).

Neurons are generally multiple input, single output devices. The strength of the output signal from a neuron is a function of the weighted sum of that neuron's inputs and may be represented by the following equation:

$$Y_i = f_i\left(\sum_j W_{ij} \cdot X_j - T_i\right)$$

Where X_j are inputs to the neuron (possibly from other neurons), W_{ij} are weights applied to the inputs, the sum $\sum_j W_{ij} \cdot X_j$ is the activation level of the neuron (an internal measurement of the state of the neuron), T_i is the threshold of the neuron, f_i is an activation function (this is usually non-linear), and Y_i is the output of ith neuron. An output will be generated by the neuron when the activation level exceeds the threshold.

A weight associated with a given input may be positive, in which case a signal received at that input will cause the activation level to increase. A positive weight may therefore be considered to be an excitory input. In some instances a weight associated with a given input may be negative, in which case a signal received at that

input will cause the activation level to decrease. A negative weight may therefore be considered to be an inhibitory input.

Connections between neurons are reinforced or weakened by adjusting the values of the weights. For example, a weight associated with a particular input of a given neuron may be increased each time a signal is received at that input. A recurring input signal (i.e. a signal received several times at the same input) will gradually increase the weight associated with that input. A signal received at that input will then cause a larger increase of the activation level of the neuron.

The activation function is usually the same for all neurons and is fixed; often a sigmoid function is used.

The activity of the *i*th neuron in known neural networks is limited to being a monotonic function of its inputs X_j, determined by the values of the weights W_{ij}. This restricts the ability of known neural networks to emulate more complex, non-monotonic behaviours.

It is an object of the present invention to provide a component of a neural network which overcomes the above disadvantage.

According to a first aspect of the invention there is provided a neural network component, the component comprising a plurality of inputs, at least one processing element, at least one output, and a digital memory storing values at addresses respectively corresponding to the at least one processing element, wherein the at least one processing element is arranged to receive a value from the digital memory in response to an input signal, and is instructed to execute one of a plurality of operations by the value that is received from the digital memory.

The invention contrasts with prior art neural networks, where a value received by a processing element is always added to an activation level of the processing element (the value is always a weight). The processing element of the invention is capable of executing a plurality of operations, and the received value must therefore act as an instruction in order for the processing element to choose one of the instructions for execution.

The activation level of a processing element is an internal value indicative of the excitation of that processing element.

Suitably, the operation is the addition of the value to an activation level of the at least one processing element.

Suitably, the operation is a decay of an activation level of the at least one processing element to zero, by increasing the activation level if it is negative or decreasing the activation level if it is positive.

Suitably, the rate of decay is dependent upon the value received from the digital memory.

Preferably, the decay is controlled by a clock.

Suitably, the operation comprises generation of an output by the at least one processing element, and then resetting an activation level of the at least one processing element to a predetermined minimum level.

Suitably, the operation comprises resetting an activation level of the at least one processing element to a predetermined minimum level without an output being generated by the at least one processing element.

Suitably, the operation comprises modifying an activation level of the at least one processing element in accordance with an automatic learning algorithm.

Suitably, the operation comprises modifying the value held in the digital memory in accordance with an automatic learning algorithm.

Suitably, the automatic learning algorithm comprises a Hebbian learning algorithm.

Suitably, the output generated by the neural network component comprises a number representative of the address of the processing element from which the output emanated.

Suitably, the neural network component is provided with arbitration and encoding means arranged to determine the chronological order of signal pulses received from processing elements, and to represent each signal pulse as a number. This is advantageous because it overcomes the problem of pin restriction for off-chip communication, replacing for example 256 pins, if each neuron output has its own pin, by 8 pins (where 8-bit binary address encoding is used).

Suitably, the neural network component further comprises a conventional processor with read-write access to the digital memory.

Suitably, accesses to the digital memory by the conventional processor are arbitrated asynchronously with accesses required by the neural processing elements.

Suitably, a large number of neural network components are supervised by a parallel network of conventional processors.

Preferably, the digital memory is a Random Access Memory.

Suitably the component works in an asynchronous manner. Alternatively, the neural network component may work in a clocked manner.

According to a second aspect of the invention there is provided a method of operating a neural network component, the method comprising storing values in a digital memory at addresses respectively corresponding to at least one processing element, transferring a value to the at least one processing element and executing one of a plurality of operations at the processing element in response to the value that is received from the digital memory.

The method may include any of the above suitable or preferable features of the first aspect of the invention.

A specific embodiment of the invention will now be described by way of example only with reference to the accompanying Figures, in which:

Figure 1 is a schematic representation of an embodiment of a neural network component according to the invention;
Figure 2 is a schematic representation the neural network component, including arbitration and encoding means;
Figure 3 is a schematic representation of the arbitration and encoding means shown in Figure 2;
Figure 4 is a graph illustrating a Hebbian learning mechanism; and
Figures 5a to 5c are flow charts illustrating operation of the neural network component.

A neural network component as shown in Figure 1 comprises 1024 inputs 1, a Random Access Memory (RAM) 2, 256 processing elements 3 and 256 outputs 4.

Each input carries a signal from a processing element of a previous neural network component. Each input carries only 1 bit of information, in other words it is either 'on' or 'off'. Input signals take the form of pulses on the inputs.

The RAM 2 contains a set of values. A different address in the RAM is allocated to each combination of input and output (the number of addresses is 1024x256). Referring to Figure 1, in order to assist the reader and for illustration purposes only, the RAM may be considered to be a two dimensional matrix comprising columns and rows, each row containing all of the values pertaining to a particular input, and each column containing all of the values pertaining to a particular processing element. As discussed further below, a significant proportion of the RAM addresses may contain zero values.

In response to a pulse received at a given input, the RAM 2 looks up values located in the row corresponding to that input. Each value is passed to the processing element 3 corresponding to the column in which that value is located. The activation level of the processing element 3 (an internal measurement of the state of the processing element) is modified in accordance with the value that is received. Each processing element 3 acts as a neuron, and has 1024 inputs and 1 output.

In contrast to the prior art, the values stored in the RAM 2 may be weights or they may be instructions. Where the value passed to a processing element 3 is an instruction, the processing element 3 carries out an operation in accordance with the instruction, as described further below. Where the value passed to the processing element 3 is a weight, that weight is added to the activation level of the processing element 3. The operation of a processing element 'i' may be expressed mathematically as:

$$Y_i = f_i\left(\sum_j W_{ij} \cdot X_j - T_i\right)$$

Where X_j are inputs to the neuron, W_{ij} are the weights, the sum $\sum_j W_{ij} \cdot X_j$ is the activation level of the neuron, T_i is the threshold of the processing element, f_i is an activation function (this is usually non-linear), and Y_i is the output of ith processing element. A pulse is output by the processing element when the activation level exceeds the threshold (this is in effect the firing of the neuron).

In the present example, j, which designates a specific input may be between 1 and 1024, and i, which designates a specific output may be between 1 and 256.

Many prior art neural networks comprise neurons which generate analogue output values. For example, referring to the above equation, the output Y_i of a neuron would be a number having a value between 0 and 1. This output, when input to a subsequent neuron would be multiplied by a weight, and the result of the multiplication added to the activation level of that subsequent neuron.

The described embodiment of the invention operates in a different manner to analogue neural networks, in that a processing element (neuron) has only two possible outputs, i.e. 1 or 0 (or equivalently 'on' or 'off'). An output pulse is generated by a processing element whenever the activation level of that processing element exceeds its threshold. From an outside observer's point of view, a pulsed output of a processing element may be considered to be equivalent to an analogue output. For example, if a processing element were to generate 23 pulses in one second, then this could be considered to be equivalent to a notional analogue value of 0.23. The generation of pulses by the processing elements is referred to hereafter as 'pulse rate encoding'.

At first sight it might appear that pulse rate encoding is very computationally intensive when compared to the use of analogue signals. For example, a weight allocated to a given input of a neuron may be multiplied by the analogue signal 0.23 in a single calculation, and then added to the activation level of the neuron. In

contrast to this, when using pulse rate encoding the weight must be added to the activation level of the neuron 23 times (i.e. once per pulse), requiring 23 separate calculations. It is noted however that there is often no output from a given neuron. Where pulse rate encoding is used, no calculations are carried out in this eventuality since no input pulses are received. In contrast to this, in known analogue neural networks, a zero value of output is effectively an output value that happens to be zero. The weights of the neurons are multiplied by the zero output value, and the result (also zero) is added to the activation levels of the neurons. This calculation, which has no effect on the activation levels of the neurons occurs very frequently and is computationally very wasteful. The use of pulse rate encoding avoids this wasteful multiplication by zero.

The neural network component is provided with 256 processing elements 3. Each processing element 3 acts as a neuron, and has 1024 inputs and 1 output. If individual connections were to be provided between each processing element and each subsequent neural network component, then the number of connections would rapidly become so large as to be impractical. In order to avoid this problem, as shown in Figure 2, the outputs from the processing elements are all directed into a single 8-bit output line 5. An output pulse from a given processing element is converted into an 8-bit binary number representative of that processing element, which is carried by the output line. This conversion, which may be referred to as 'addressed-event encoding' is carried out by an arbitration and encoding unit 6.

The operation of the arbitration and encoding unit 6 is illustrated schematically in Figure 3. The unit 6 has two functions: to determine the chronological order in which pulses are to be sent to the output line 5, and to convert each output into a number indicative of the processing element that generated that output. The two functions are achieved simultaneously by the unit 6. The unit 6 shown in Figure 3 is for a set of eight processing elements.

A first tier of the unit 6 comprises four arbitration and encoding blocks 10-13. An input side of a first arbitration and encoding block 10 is provided with two inputs a, b and two acknowledge outputs a_{ack}, b_{ack}. When the output of a processing element goes high, this is received at input a, a first output y of the block 10 goes high, and a second output z simultaneously outputs a 1-bit number representative of input a (in this case the binary number is '1'). The acknowledge output a_{ack} then goes high, thereby indicating to the processing element that the output has been processed by the block 10. The output of the processing element then goes low.

If input a and input b go high at approximately the same time, then a Seitz mutual exclusion element determines which input was the first to go high (Mutual exclusion elements are described at pages 260 *et seq.* of 'Introduction to VLSI systems' by Mead and Conway, published by Addison Wesley). Output from the block 10 is generated accordingly. Once the output has been generated, and an acknowledge sent back to the appropriate processing element, the later of the inputs is processed and output by the block 10.

A second tier of the unit 6 comprises a pair of arbitration and encoding blocks 14,15. Referring to a first block 14 of the pair, two inputs y, w are connected to outputs of preceding blocks 10, 11. A first output u of the block 14 goes high in response to an input signal, and a second output v simultaneously outputs a 2-bit number. The most significant bit of the 2-bit number represents the input y, and the least significant bit of the 2-bit number represents the input a of the preceding block 10. Thus, the 2-bit number is the binary number '11'.

The third and final tier of the unit 6 operates in the same manner as the previous tiers. The output from the single block 16 comprising the final tier is a 3-bit binary number, and in this case is the number '111'.

In the description of Figure 1, the output from a processing element 3 is said to be in the form of a pulse. It will be noted that in the description of Figure 3 the output from the processing element is said to be a continuous high which ends only after an acknowledge signal has been received by the processing element. This modification is required in order to allow the arbitration and encoding unit 6 to function correctly.

Referring to Figure 2, eight tiers of arbitration and encoding blocks are required to generate an 8-bit binary number output.

Referring to Figure 2, four 8-bit input lines 19 are connected to the processing elements 3 and RAM 2. An arbitration and encoding unit 20 is used to arbitrate between the four input lines 19. The arbitration and encoding unit 20 operates the manner shown schematically in Figure 2 and described above. In this case the arbitration and encoding unit includes two tiers of blocks. The arbitration and encoding unit has a 10-bit output, the extra two bits being used to distinguish between the four 8-bit input lines 19.

The 10-bit output of the arbitration and encoding unit 20 is connected to a decoder 21. The decoder converts each received 10-bit input number to an output on a corresponding 1-bit output line 1 (there are 1024 output lines). The output lines 1 correspond to the input lines 1 shown in Figure 1.

In the pulse rate encoding system used by the embodiment of the invention, communication between neural network components takes place on a time scale of the order of 1μs. This appears instantaneous compared with the temporal resolution of the neurons which fire at a rate no higher than of the order of 1kHz.

The fact that the neural network component is digital provides a degree of flexibility not seen in analogue neural network components. In particular, a large range of values may be stored in the RAM 2, and this allows some of the values to be used as

'instructions' for the processing elements 3. The term 'instruction' is intended to mean that the processing element is instructed to carry out an operation other than simply adding a value to the activation level. In known prior art neural networks, the action carried out by a neuron is always the same, i.e. to add a value to the activation level. The use of instructions allows the processing elements 3 to have much more flexibility in their operation.

The use of instructions provides a neural network having properties which are closer to biologically realistic neurons. One such property is a 'leaky' integration, in other words an activation level which decays if there is no input activity over a period of time. Another property is 'refraction', meaning that a processing element 3 ignores all inputs for a certain period following firing. This limits the processing element's maximum firing rate (i.e. the rate at which the processing element 3 can produce output pulses), thereby preventing the network from becoming unstable due to ever-increasing neuronal activity. Both refraction and leaky integration imply some dependence on real time.

Refraction and leaky integration are provided in the described embodiment of the invention using a real-time reference event, for example a 32 kHz clock (not shown in Figures 1 or 2). Pulses from the clock may transmitted as a particular binary number via the 8-bit input (for example 00000001). Where leaky integration is required, the value stored in each RAM address corresponding to this input number will instruct the processing element to decrement its activation level if the activation level is positive and increment the activation level if it is negative. Different values held in the RAM may instruct different rates of leaky integration, for example increment/decrement in response to every clock pulse, or increment/decrement in response to every second clock pulse.

An instruction may be provided which determines whether or not the weight should be adjusted by automatic learning algorithms (for example Hebbian learning

algorithms). As an example of a neuron with automatic learning capabilities, the following neuron behaviour can be modelled as shown in Figure 4.

The activation level of the processing element operates within the range {-L, +T}, where –L is the refractory level and T is the threshold which, when reached, causes the processing element to output a pulse and the activation level to be reset to –L. {-L, 0} is the refractory range of the processing element: if a weight is passed to the processing element in response to an input pulse, the weight is **not** added to the activation level (the activation level is unchanged), and the weight itself is decremented. {0, F} is the weakly excited range: : if a weight is passed to the processing element in response to an input pulse it will be added to the activation level and the weight itself will be unchanged by the learning mechanism. {F, T} is the strongly excited range: if a weight is passed to the processing element in response to an input pulse it will be added to the activation level and the weight itself will be incremented.

The operation of a processing element arranged to operate in accordance with the model shown in Figure 3 is shown schematically by a flow chart in Figure 5a-c.

Referring first to Figure 5a, a pulse is received at a given input j. The value associated with that input j for a given processing element is retrieved from the RAM and passed to the processing element. The value will fall within one of three ranges: a first range corresponds to weights which are to be added to the activation level, a second range corresponds to weights which are to be added to the activation level and in addition instruct the processing element that the weight associated with that input is to be incremented, and a third range comprises instructions other than weights.

The remainder of Figure 5a deals with a value which falls within the first range. The value is added to the activation level of the processing element (the activation level will be reduced if the weight is negative). If the activation level following addition of

the value is less than the threshold T then no further action is taken. If the activation level is greater than the threshold T, then the processing element outputs a pulse and the activation level of the processing element is reduced to –L.

If the value falls within the second range then operation of the processing element as shown in the flow chart branches to Figure 5b. If the activation level of the processing element is greater than or equal to zero then the value is added to the activation level of the processing element. Following this, if the activation level is greater than F then the value saved in the RAM is incremented.

If the activation level of the processing element is less than zero then the value is not added to the activation level, and the value saved in the RAM is decremented.

If the value falls within the third range, this indicates some other form of instruction. For example, as shown in Figure 5c, the processing element may be instructed to emit a pulse irrespective of the activation level, and reset the activation level to the refractory level.

This Hebbian learning algorithm, which results in a stable neural network if appropriate values are chosen, depends on a mechanism to return the activation level from –L to 0 which is separate from the numerical weight processing, and therefore depends on the use of an 'increment towards zero over time' instruction which may be implemented as described above.

It will be understood by those skilled in the art that the processing elements may be capable of executing other instructions, which may be defined by values stored in the RAM value memory. For example, the activation level may be reset to the refractory level without firing the neuron. This models a strongly inhibitory input.

Referring again to Figure 1, the neural network component may include a conventional processor (not shown) with read-write access to the RAM value store 4. In addition, the processor may have the capability to monitor and generate input and output to neural activity streams, using suitable interface peripherals incorporated into the neural network component.

Processor accesses to the RAM value store 4 and the neural activity streams can be arbitrated asynchronously into the fully operational network with minimal interference with the neural processing provided that the network is not operating near to saturation at the time. One monitor processor could supervise a number of neural network components, but a very large system may require several such processors. An implementation based on 1 Gbit DRAM technology might have a monitor processor on each chip. Thus, the neural network may be implemented as a very large number of neural network components supervised by a parallel network of conventional processors. The ease with which the architecture connects to conventional machines is a merit of the invention. The control processor greatly extends the flexibility of the architecture and provides a close coupling between the neural processing and a conventional, sequential machine.

If communication is implemented using clocked logic the temporal firing information will be quantised. If asynchronous logic is used this quantisation is avoided. The use of asynchronous logic is therefore advantageous.

Although in the illustrated embodiment one processing element corresponds to one neuron, it will be appreciated that a processing element may be responsible for performing the calculations for more than one neuron.

Claims

1. A neural network component, the component comprising a plurality of inputs, at least one processing element, at least one output, and a digital memory storing values at addresses respectively corresponding to the at least one processing element, wherein the at least one processing element is arranged to receive a value from the digital memory in response to an input signal, and is instructed to execute one of a plurality of operations by the value that is received from the digital memory.

2. A neural network component according to claim 1, wherein the operation is the addition of the value to an activation level of the at least one processing element.

3. A neural network component according to claim 1, wherein the operation is a decay of an activation level of the at least one processing element to zero, by increasing the activation level if it is negative or decreasing the activation level if it is positive.

4. A neural network component according to claim 3, wherein the rate of decay is dependent upon the value received from the digital memory.

5. A neural network component according to claim 3 or 4, wherein the decay is controlled by a clock.

6. A neural network component according to claim 1, wherein the operation comprises generation of an output by the at least one processing element, and then resetting an activation level of the at least one processing element to a predetermined minimum level.

7. A neural network component according to claim 1, wherein the operation comprises resetting an activation level of the at least one processing element to a

predetermined minimum level without an output being generated by the at least one processing element.

8. A neural network component according to claim 1, wherein the operation comprises modifying an activation level of the at least one processing element in accordance with an automatic learning algorithm.

9. A neural network component according to claim 1 or claim 8, wherein the operation comprises modifying the value held in the digital memory in accordance with an automatic learning algorithm.

10. A neural network component according to claim 8 or claim 9, wherein the automatic learning algorithm comprises a Hebbian learning algorithm.

11. A neural network component according to any preceding claim, wherein the output generated by the neural network component comprises a number representative of the address of the processing element from which the output emanated.

12. A neural network component according to claim 11, wherein the neural network component is provided with arbitration and encoding means arranged to determine the chronological order of signal pulses received from processing elements, and to represent each signal pulse as a number.

13. A neural network component according to any preceding claim, further comprising a conventional processor with read-write access to the digital memory.

14. A neural network component according to claim 13, wherein accesses to the digital memory by the conventional processor are arbitrated asynchronously with accesses required by the neural processing elements.

15. A neural network component according to claim 13 or 14, wherein a large number of neural network components are supervised by a parallel network of conventional processors.

16. A neural network component according to any preceding claim, wherein the digital memory is a Random Access Memory.

17. A neural network component according to any preceding claim, wherein the component works in an asynchronous manner.

18. A method of operating a neural network component, the method comprising storing values in a digital memory at addresses respectively corresponding to at least one processing element, transferring a value to the at least one processing element and executing one of a plurality of operations at the processing element in response to the value that is received from the digital memory.

19. A neural network component substantially as hereinbefore described with reference to the accompanying figures.

20. A method of operating a neural network component substantially as hereinbefore described with reference to the accompanying figures.

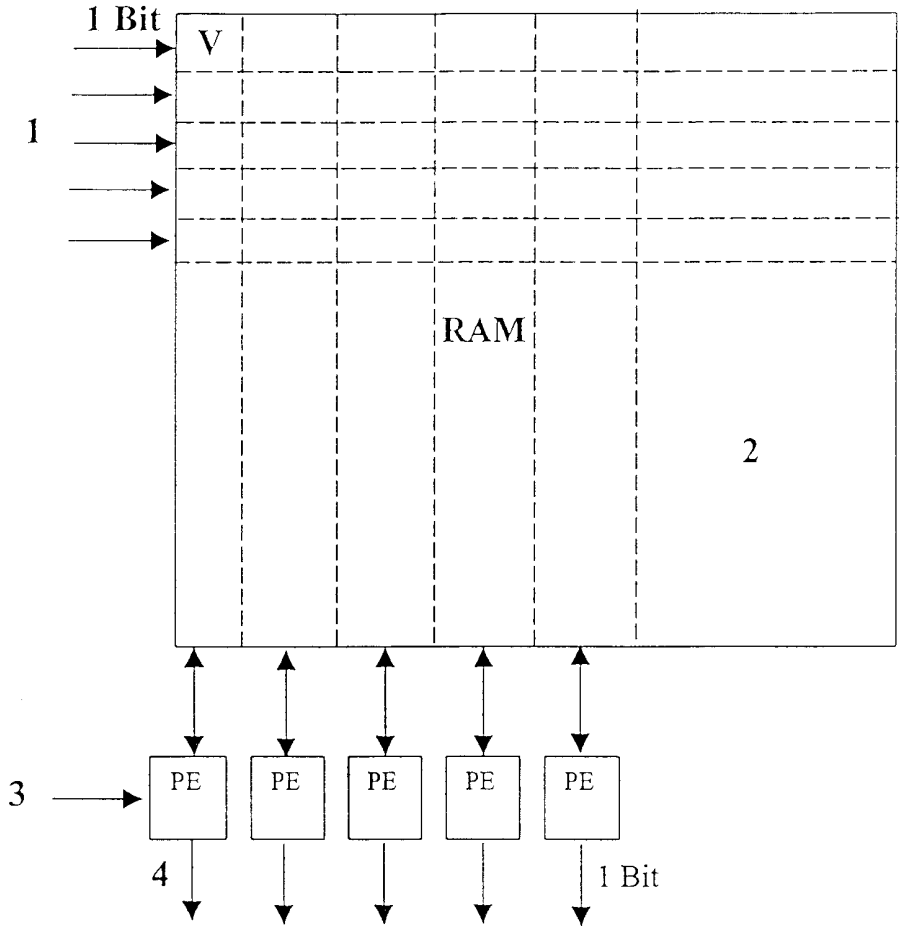

FIG. 1

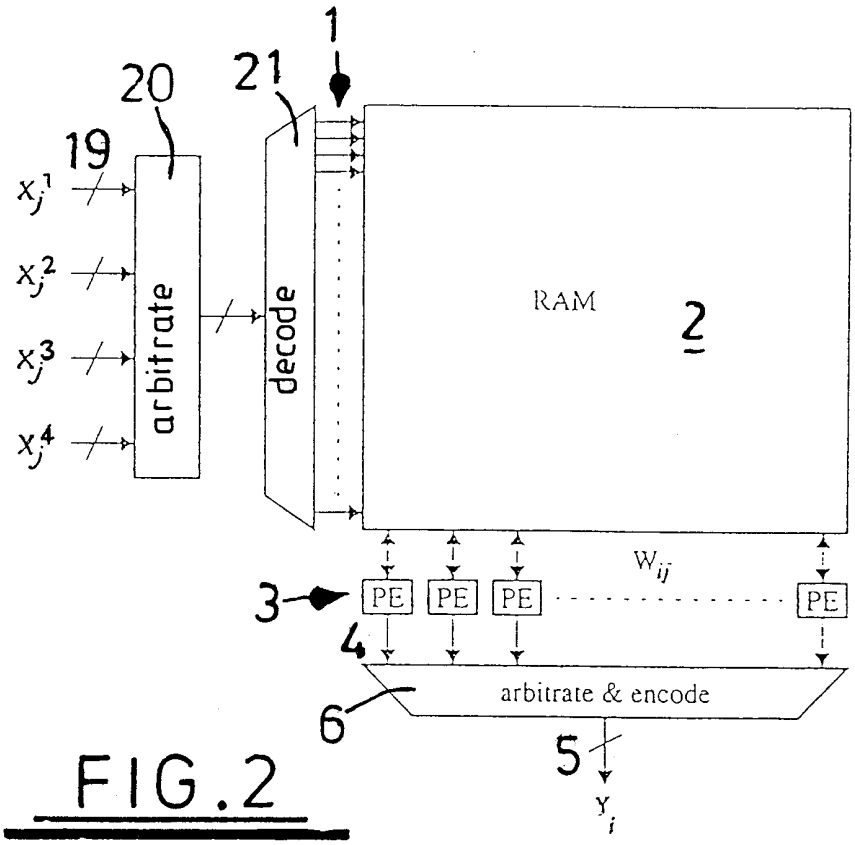
1
20
21
19
x_j^1
x_j^2
x_j^3
x_j^4
arbitrate
decode
RAM
2
W_{ij}
3
PE
PE
PE
PE
4
arbitrate & encode
6
5
Y_i

FIG.2

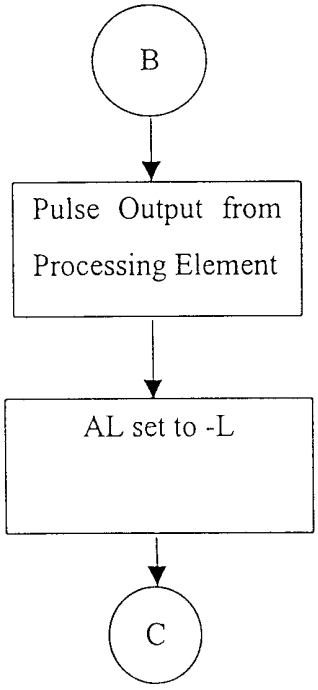
B
Pulse Output from Processing Element
AL set to -L
C

FIG.5c

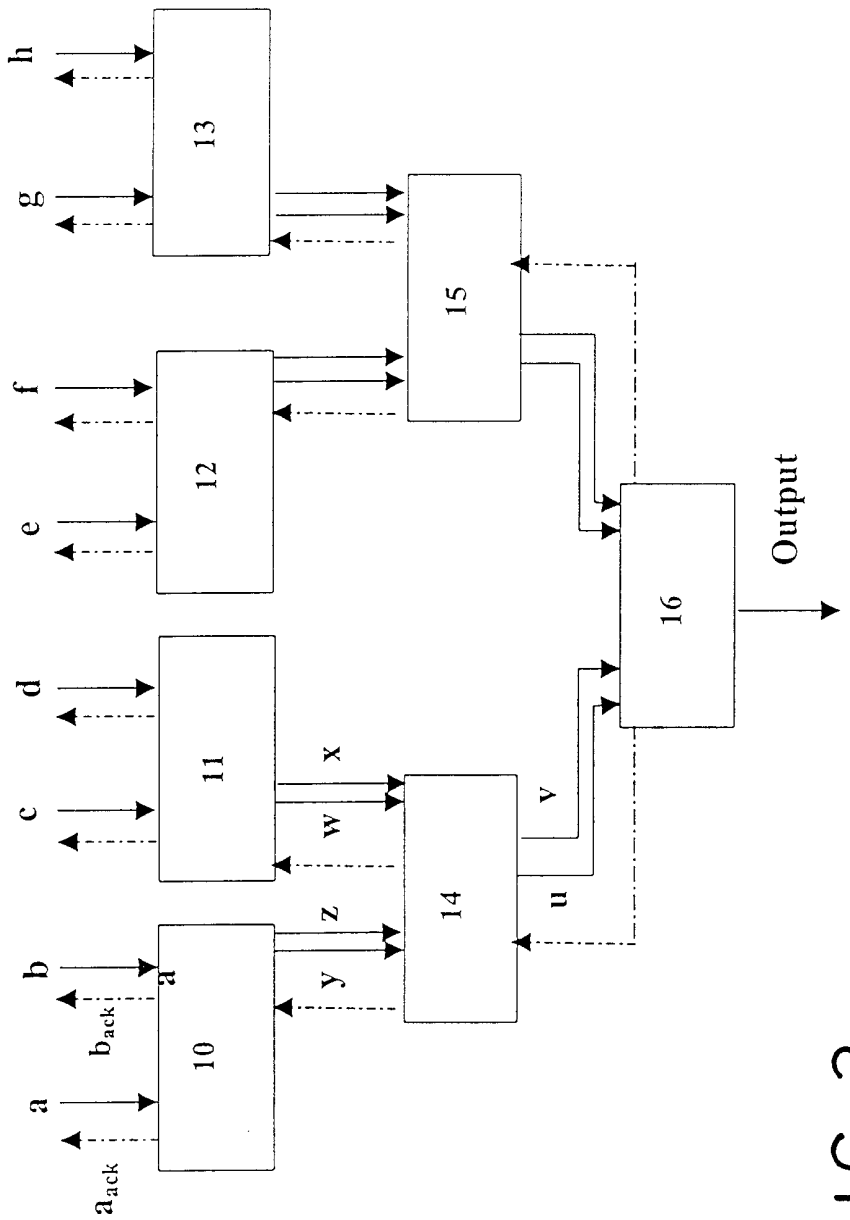

FIG.3

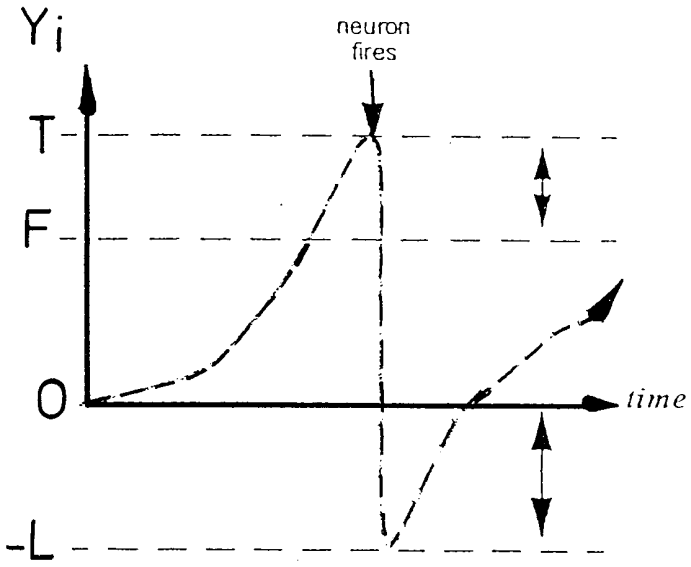

FIG. 4

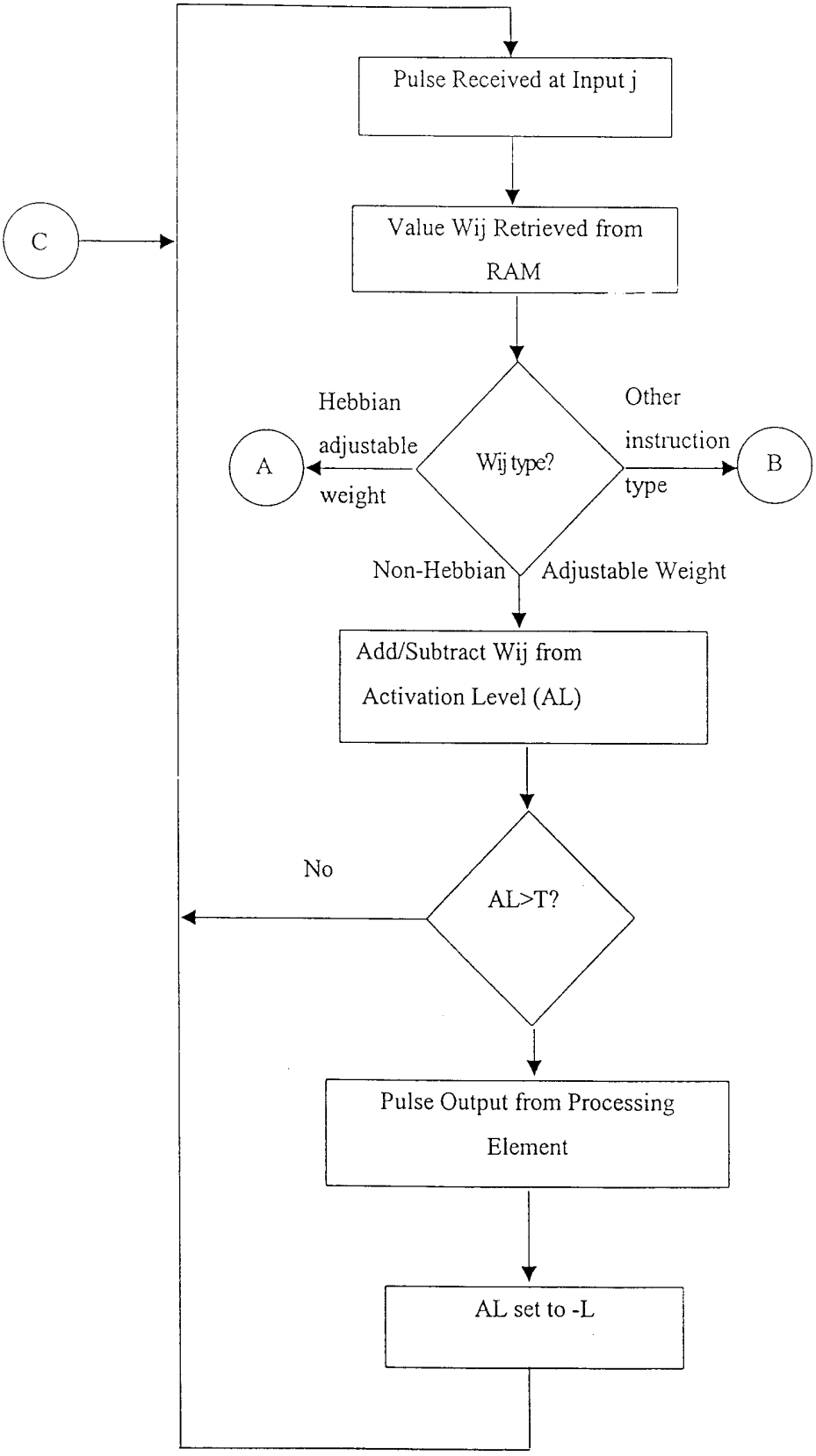
Pulse Received at Input j
C
Value Wij Retrieved from RAM
Hebbian adjustable weight
A
Wij type?
Other instruction type
B
Non-Hebbian
Adjustable Weight
Add/Subtract Wij from Activation Level (AL)
No
AL>T?
Pulse Output from Processing Element
AL set to -L

FIG. 5a

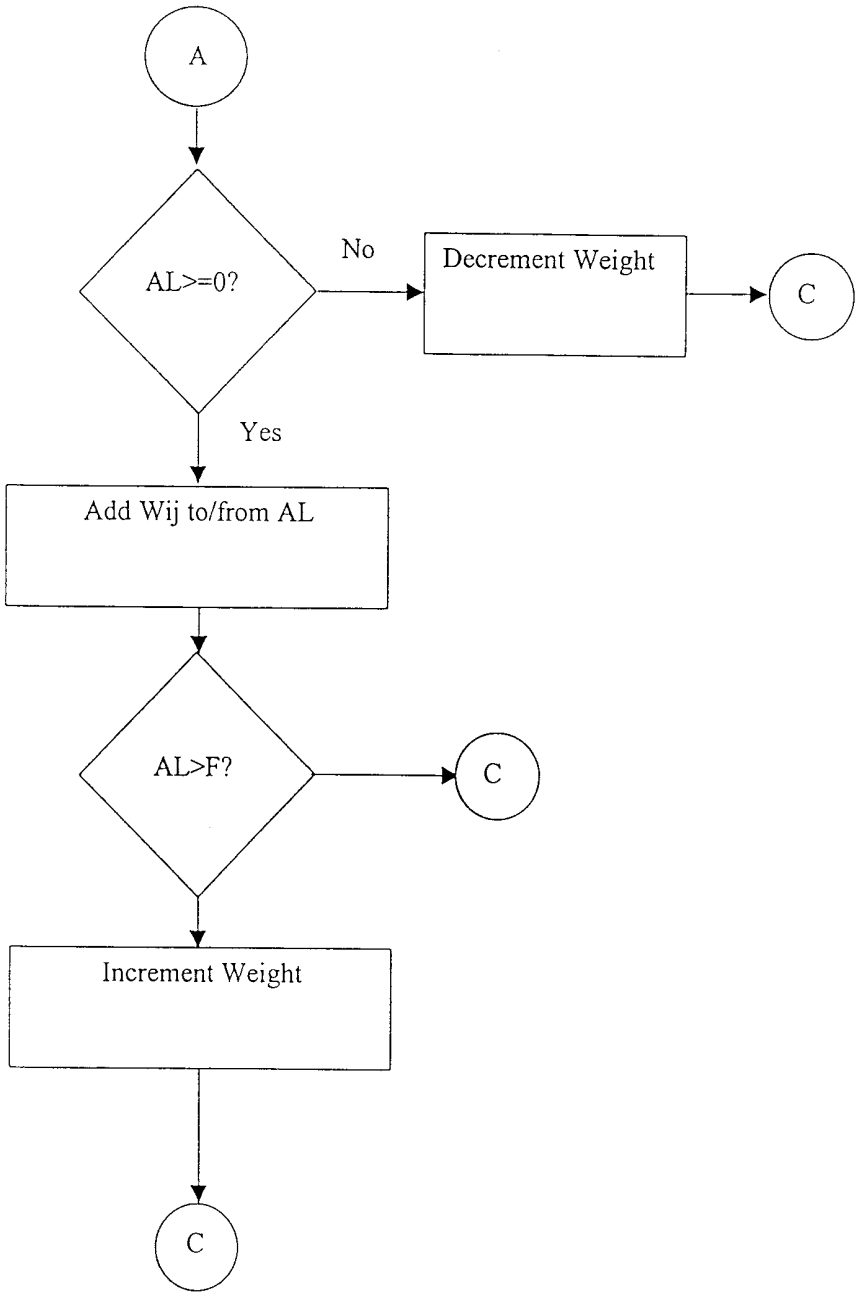
A
AL>=0?
No
Decrement Weight
C
Yes
Add Wij to/from AL
AL>F?
C
Increment Weight
C

FIG.5b