HGN AM parallelised over a group of processors

SHINAN TENG 25887319

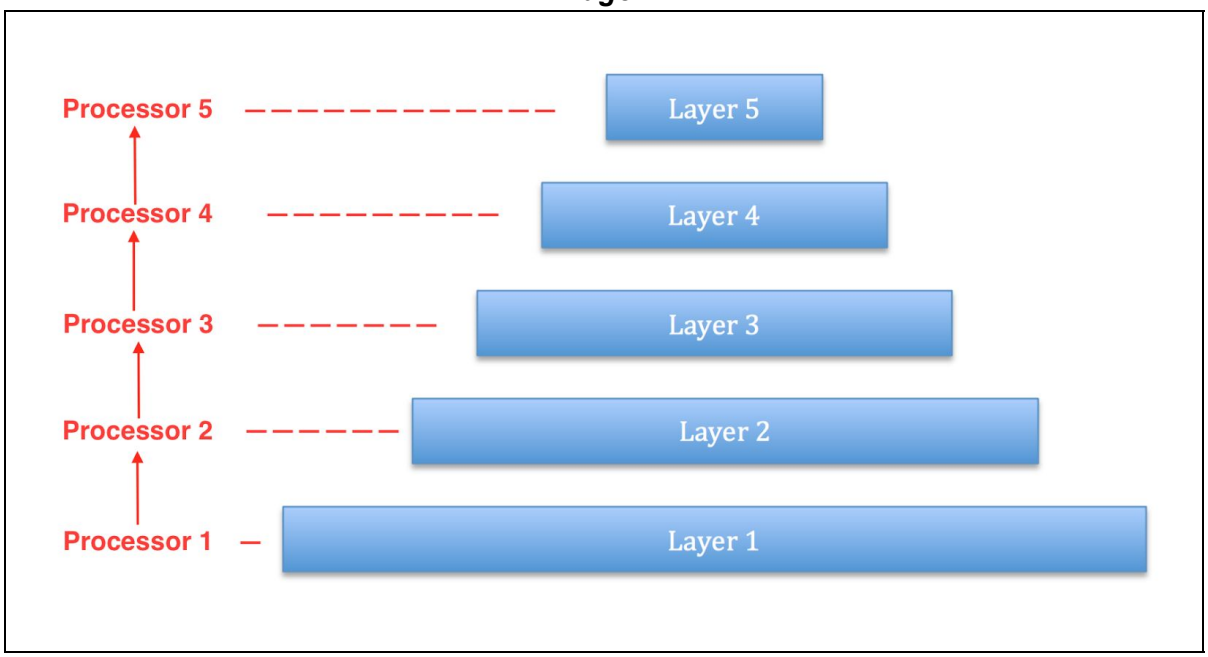
In this paper, due to the hierarchical graph neuron (HGN) uses a scalable associative memory and architecture, which only requires low maintenance floating point computations. Therefore, the HGN is very suitable for real-time calculating and can be easily deployed on a wireless sensor network. To reduce the inter-process communication and enhance the capabilities of real-time pattern recognition, we can set up a wireless sensor network as an in-networking computer, namely a batch of simple communicating sequential processes to help the system accomplish the complicated work.

How to the HGN can be parallelised over a group of processors?

We can find that in the hierarchical graph neuron, different hierarchies have different graph neurons. At the bottom of the layer, it has the largest number of graph neurons. When we move to the upper layer, it will remove the GNs at the edge, namely the upper layers have less GNs.

At first, I thought if each processor can control a single layer, but later I realized that it is not efficient. Here's the reason:

Image 1

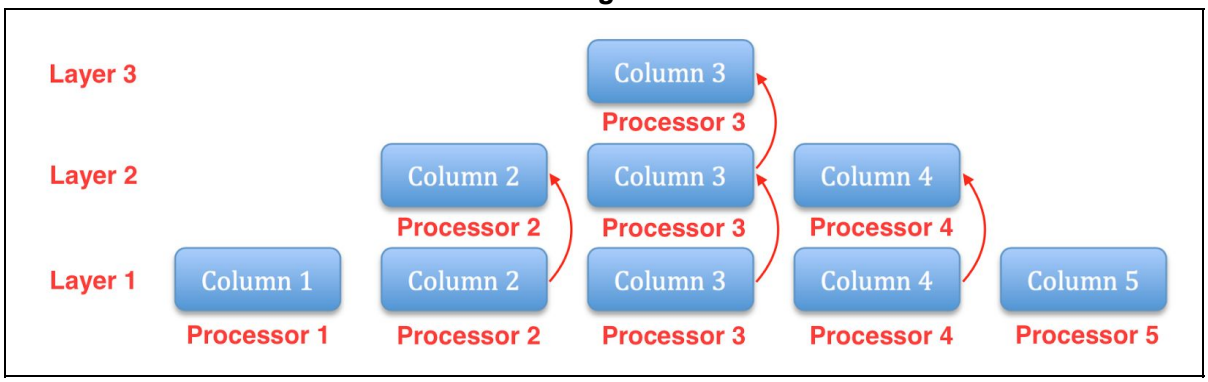


If doing like this, the processor at the bottom will accept more inputs and calculate more data, so the processor at the bottom will execute a longer time than other processors. In the case of Image 1. The execution time will like this: P1>P2>P3>P4>P5

Moreover, processor 1 will execute commands first then pass the result to the processor 2. Namely, processor 2 has to wait when processor 1 has finished its job. It's nothing different from a sequential processing. So, this proposal is useless.

Then, I thought if each processor can control a single column. This should be better than the above one. Here's the reason:

Image 2



If doing like this, every processor will have the same work load, so the execution time will almost the same and each processor doesn't have to wait for any other processor. The inter-process communication only happens when one layer passes pair value to the upper layer (e.g. Column 2 in the Layer 1 passes pair value to the Column 2 in the Layer 2). So this will reduce the inter-processing communication.

HGN Communication Schema

(Step 1) All GNs at the base layer will accept the pair value(value, command). The command contains either "store" or "recall".
(Step 2) All active GNs respond by sending a pair value(column, row) on both preceding and succeeding columns.
(Step 3) Every GN has to determine to store the bias entry or mark it as the recall.

- If bias entry exists in the memory --> GNs at the base layer read the index of the bias entry.
- If bias entry doesn't exist --> The command need to decide what to do.

(Step 4) Each active GN sends its pair value(row, bias index, command) to all the GNs in the same column (to the upper layers). Finally, the active GN should respond to the stimulator.