Content-addressable search engine

Parallel HGN Scheme

Parallelism within the HGN schema

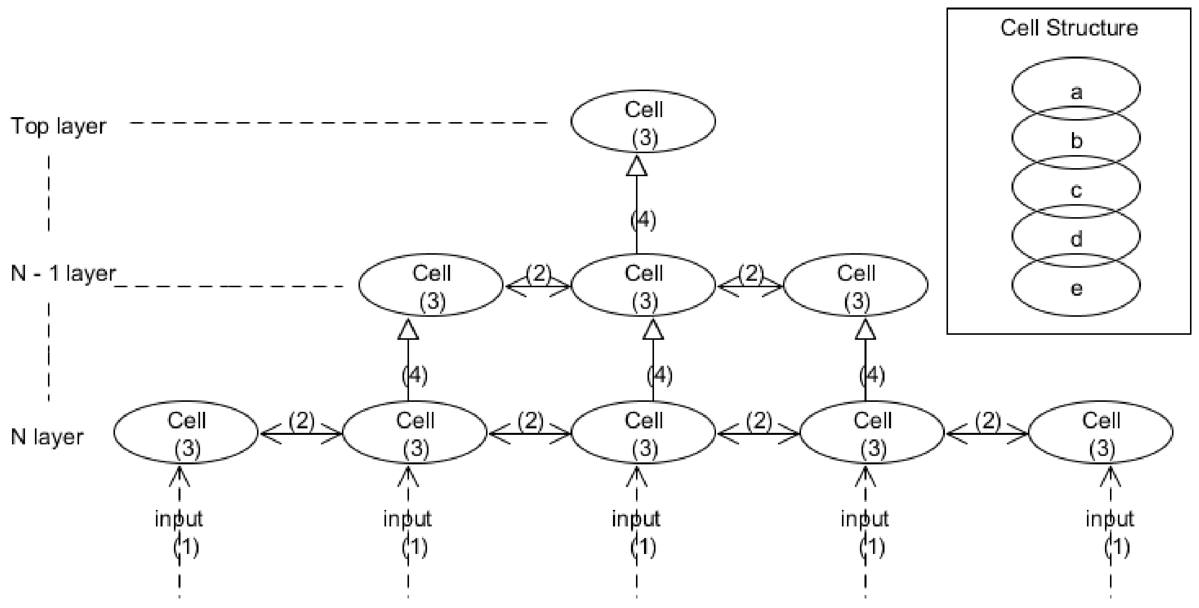

Figure 1: HGN Communication Schema (Pattern size = 5)

As I see, the premise of parallelism is relative independency. Firstly, the tasks to be parallelized cannot depend each other. For example, it is impossible to execute task A and B simultaneously if task A requires the output of task B. Furthermore, it becomes complex if task A and B require the same resource. Another finding is in many times loops are the points to parallelize. However, it is inappropriate to separate a task into too small pieces. As the increasing of processors, the communication cost will increase as well.

Figure 1 shows the topology and communication schema of HGN. It assumes the pattern size N is 5. So the total number of layers is 3 (N-2). There are 5 (N) columns in the base layer. Each column comprises of 5 GNs. There are generally 4 steps in the communication schema.

(1) The input stream is assigned to the cells in base layer respectively. The matched GN in the cells will be active and its position is recorded. These tasks are individual and do not depend on each other. So they are parallelizable. On the other hand, each GN in one cell is independent. However, it would make the task pieces too trivial.

(2) The active GN in one cell sends message to its neighbors. These can be supposed as normal message passing in parallel computing.

(3) Each active GN updates its bias entries using message received from neighbors. These are individual tasks as well.

(4) Each active GN (except them in bordered columns) sends their bias entry index and position to the corresponding column in upper layer. This means the tasks on upper layers will depend on the lower layers' output. In other words, it is less possible to parallel tasks according to layers.

The algorithm then repeats these steps to upper layers until the top one. It can be regarded as a loop. However, since the dependency between layers, it is inappropriate to separate this loop tasks.

Based on above analysis, I would recommend to parallelize the tasks according to the columns instead of layers. It can utilize Geometric Parallel Schema which maps the tasks a pipeline of processors. A possible parallel flow-char is showed in Figure 2.

Flow-chart for Parallel HGN

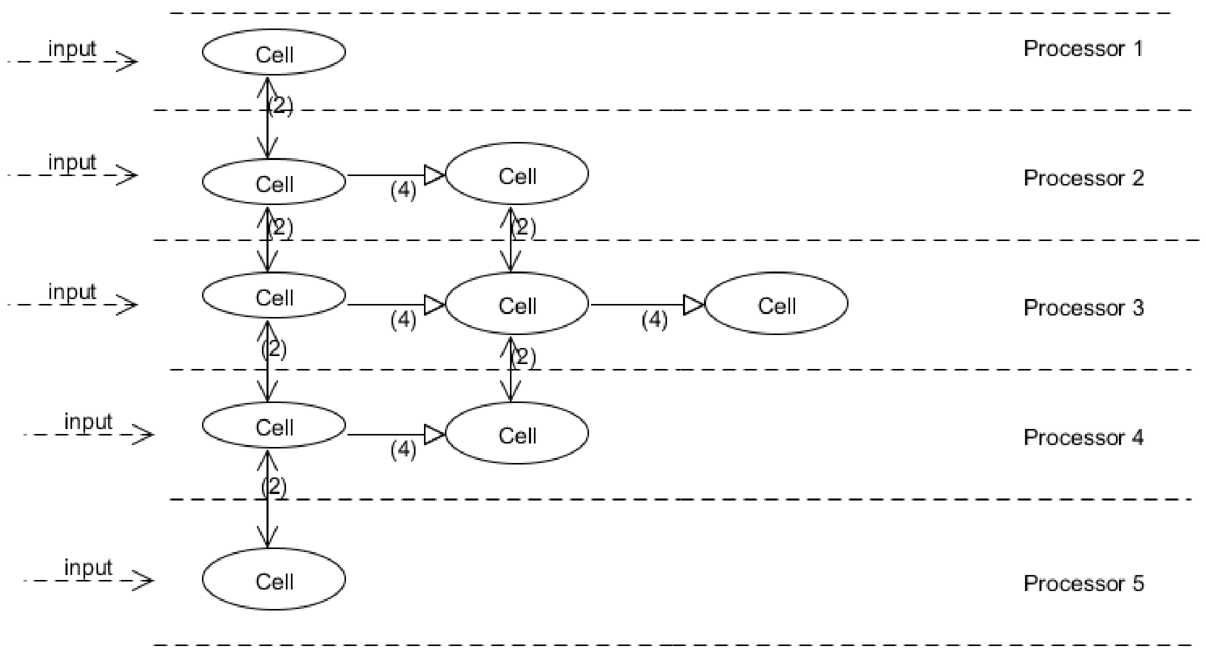

Figure 2: Parallel HGN Flow-chart

The parallel HGN work flow can be like this:

(1) The program allocates a number of processors based on the pattern size

(2) Initialize and generate the GNs and cell structures in each processors. The cell numbers varies on the processor rank. The first and last processors host only one cell each.

(3) Scatter the input over the processors

(4) The cell activate and record matched GNs. After that, the processors send message to neighbor processors (rank -1 and rank + 1). And then start receive messages as well.

(5) Each cell updates bias entries using received message.

(6) Then the cell in upper layer starts to work using the result of lower layers

(7) Repeat 4 – 6 until the last cell in the middle processor

It is obvious that the tasks on each processor is skewed. The execution time finally depends on the middle processor and bordered processors do not contribute much to the whole process.