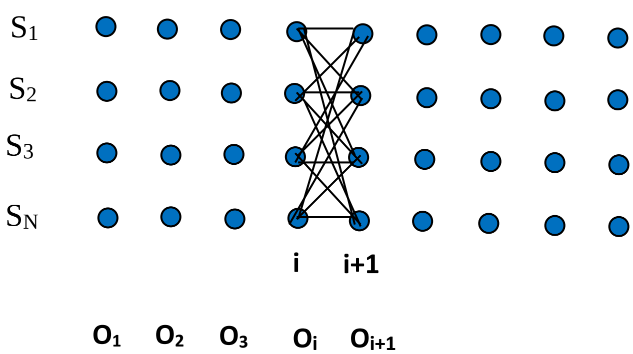
Parallel implementation of the modified Viterbi algorithm[1]

Final project in the course Parallel and Distributed Computation, 2014 Fall Semester

1. Definitions and the algorithm.

Let's consider a graph with nodes that are ordered in vertical time slices (trellis). Each node on time slice **i** is connected to each node next time slices **i+1.**



Definitions:

Each horizontal level j of nodes represents some state S_j of the model. This model is associated with one of the N given states S_1 S_2 S_3 S_4 … S_N for each time slice. If the model was in state S_i for some time slice and have changed to the state S_j for the next time slice, we will call a value $t_{i,j}$ as "transition".

For each state S_j are defined a pair of values (a_j, b_j) and a function f(j, i) which is called "emission function"

$$\mathbf{f(j, i) = a_j \exp(-(o_i - b_j)^2)}$$

Let's call $d_{m,i}$ a "state value" for state S_m at time slice i.

The set of values $o_1, o_2, o_3, \dots, o_K$ is called the observation sequence.

[1] Viterbi algorithm, http://en.wikipedia.org/wiki/Viterbi_algorithm, 1.01.14

Algorithm:

Suppose that the observation sequence $o_1, o_2, o_3, \dots, o_K$ is known for K time slices.

1. Assign $d_{m,i}$ to 1 for each state **m** on first time slice

$$d_{m,1} = 1 \text{ for each } m = 1, 2, \dots, N$$

2. Suppose that $d_{m,i}$ is known for each **m** on some time slice **i**.
3. Calculate $d_{m,i+1}$ for the next time slice **i+1** as follows:

$$d_{m,i+1} = \max[d_{m,1}*t_{1,m}*f(1, i),\ \ d_{m,2}*t_{2,m}*f(2, i),\ \ d_{m,3}*t_{3,m}*f(3, i), \dots, d_{m,N}*t_{N,m}*f(N, i)]$$

4. Remember the state on time slice **i** that supplied the maximum value of $d_{m,i+1}$ for each m on time slice **i+1**.
5. Repeat steps 3 and 4 till the very end of the trellis.
6. Choose the maximum value of $d_{m,K}$ on the last time slice K.
7. Make back propagation from the last time slice to the first time slice using the saved information on step 4. The found sequence of states in forward direction is called the State Transition Path.

NOTE:

If for some time slice **i** the observation is equal to zero than the back propagation have to be performed back from this time slice. Perform steps 1 – 7 starting from the time slice **i+1.**

2. Requirements:

Implement the Viterbi algorithm using heterogeneous environment MPI, OpenMP, CUDA. The purpose of the application is to find the State Transition Path for the given sequence of observations, transition matrix $T = \{t_{ij}\}$ and emission function for each state.

Grade Policy:

60 points for the proper parallel implementation of the Modified Viterbi algorithm with two components MPI+*OpenMP* or *OpenMP+ CUDA* or *MPI +CUDA*. The last configuration will properly run under LINUX OS or on the single machine on MS Windows environment.

10 points for implementation of *MPI+OpenMP+CUDA*. This configuration will properly run under LINUX OS or on the single machine on MS Windows environment.

10 points for the documentation of your solution – clear explanation what and how the problem was parallelized, what is a rational of choosing the specific architecture.

5 points for the code quality – modularity, generality, self-explanatory, organization.

10 points for the Load Balancing.

5 points for the complexity evaluation.

Additional Bonus for the project grade

5 points for implementation under LINUX OS

5 points for your own proposal (have to be approved by lecturer)

5 points for the solution that takes into consideration that the observation sequence is a sequence of vectors (ask the lecturer).

3. Input data and Output of the project

You will be supplied with the following data:

- 1000x1000 Transition Matrix
- 1000x2 Matrix for coefficients for the emission function
- 30000 records of the observations

The output of the application is the found State Transition Path.