**Using Machine Learning to generate minimum spanning tree for circuit interconnect network**

**Team Members:** Pritam Bhattacharyya (GTID: 902989158)

Shruti Sunil Bhave (GTID: 902988298)

Shruti Mishra (GTID: 903039268)

**Brief overview of the project idea:**

The performance of a fabricated system is heavily dependent on multiple factors associated with the quality of interconnects that connect the various devices of the system on the die. Among the various factors, the total interconnect length and its associated RC delays play a very important role in determining the overall performance of the circuit. Understandably, the generation of optimal routing plan of interconnects is regarded as a major cause of concern in the chip designing process.

In our project, we aim to explore the possibilities of obtaining the minimal spanning tree from a given set of nets using machine learning algorithms. Machine learning is extensively used in the industry to perform predictive analytics and modelling. For our project, we will focus on a subset of machine learning algorithms called supervised learning, where the system builds an algorithm to map the inputs to outputs, given a substantial set of input-output pairs.

Our perception in choosing this project is that since routing is a mathematical task at the core, the system can be trained to build an optimal routing solution given a set of nets as input. Training in the context of machine learning means supplying the system with a large corpus of input-output pairs. We will be using Cadence Encounter to generate this corpus of inputs (set of nets) and outputs (the corresponding minimum spanning tree).

Then, we plan to use the previously generated input-output pairs to train the system using artificial neural network concepts. During the training phase, using these pairs, the system will build a general rule that will map the inputs to the outputs. When the system successfully builds this general rule, it is said to be trained.

After the system is trained, we expect that it should be able to apply the rule that it built during the training phase to come up with an optimal output (the minimum spanning tree) for any random input (set of nets).

**Implementation Scope:**

In our project, as we described in the above section, we will be implementing the supervised learning program that will take a random set of nets as its input and give the corresponding minimum spanning tree as the output based on a substantial database of input-output pairs.

However, please note that we plan to use standard machine-learning libraries available for the purpose of our project. A few examples of such libraries includes scikit-learn and PyBrain – open-source machine-learning libraries available for Python.

Further possible extension to our project could include the use machine learning to obtain the Steiner tree from the minimum spanning tree, obtained in the previous stage, and compare our results with those obtained by standard Steiner tree computing algorithms.

**Goals:**

Our final aim is to determine whether by using machine-learning we are able to increase the efficiency in our computation of the minimum spanning tree in comparison to existing algorithms like PRIM and BPRIM, with regards to factors like time complexity and accuracy.

**Programming Language and Tools to be used:**

As discussed earlier, we will be using an open-source libraries like scikit-learn and PyBrain to implement the training using neural networks. The use of these libraries will require us to implement the major chunk of our training program in Python.

Finally, we plan to implement the algorithm to compute the Steiner tree from the minimum spanning tree by using C++. However, if time permits, we could implement another learning module for this step as well, which would require us to transfer this module to Python as well.

Moreover, we would like to request the permission to use MATLAB in our project implementation for the critical sections of our machine learning modules. We also plan to draw the final graphical output of the routes using a GUI tool like MATLAB (or some equivalent tool).

**Demonstration Plan:**

We plan to demonstrate our findings as a comparative analysis between the performance of our machine-learning implementation and that of standard algorithmic implementations. Moreover, we plan to plot our results graphically to represent the correctness of our results for various standard sets of nets.

**Tentative Schedule:**

As of yet, by mid-March, we plan to establish a database of the required input-output pairs and we plan to build an initial prototype of the final program by early-April.

**Alternate Choice:**

In case the above proposal is not accepted, we would like to be considered for any project that involves optimization of routing algorithms.

**References:**

[1] Tom Schaul, Justin Bayer, Daan Wierstra, Sun Yi, Martin Felder, Frank Sehnke, Thomas Rückstieß, Jürgen Schmidhuber. PyBrain. To appear in: *Journal of Machine Learning Research*, 2010.

[2] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.