A Self-Organization Maps Approach to FPGA Placement

Motoki Amagasaki Yasuaki Tomonari Masahiro Iida Morihiro Kuga Toshinori Sueyoshi

Graduate School of Science and Technology, Kumamoto University
2-39-1 Kurokami, Kumamoto 860-8555, Japan

{amagasaki, iida, kuga, sueyoshi}@cs.kumamoto-u.ac.jp, tomonari@arch.cs.kumamoto-u.ac.jp

Abstract— Cell placement is an important phase of current Field Programmable Gate Array(FPGA) circuit design. However, this placement problem is NP-hard. Although nondeterministic algorithms such as Simulated Annealing(SA) are successful in solving this problem, they are known to be slow. In this paper, we introduce a new neural network approach to placement problem of FPGA. The used network is a Kohonen self-organization Map. A connection relation ship of cluster-level netlists is converted to a set of appropriate input vectors. These vectors which have higher dimensionality are fed to the self-organization Map at random to map themselves onto a 2 dimensional plane of the regular chip. The key feature is that SOM algorithm perform the cell placement to minimize total connection length in the circuit. In this paper, we evaluate our placement tool using some benchmark circuits.

I. Introduction

In the Field Programmable Gate Arrays (FPGAs), design automation domain such as placement and routing are the most processing time intensive steps. Especially, placement is quite an important physical design process. The locations for all circuit modules with signal wires should be determined, so that criteria for optimization can be met. These criteria are mainly the minimal chip area, the minimal wire length and the 100% routability etc. The placement problem has proved to be a NP-complete problem. The processing time problem has been somewhat alleviated by the advancements in processor speeds. However, the speedup of classic placement and routing algorithms obtained by using faster processors cannot keep pace with the rate at which the FPGA complexity increases. In fact, the complexity of SA based VPlace algorithm[1] is $O(n^{4/3})$ where n is the number of logic blocks in the circuit. Therefore, a new solution method is expected with the solution quality comparable to that of the SA and the execution time shorter to that of one.

In this paper, we present a placement algorithm based on Kohonen self-organization maps(SOM)[2]. The self-organization principle has been previously applied to the placement problem. The key feature is that the computational complexity of SOM based placement is $O(n)$. In our approach, a set of n-dimensional vectors is fed to the network and the map assign the modules to the slots(neurons). In the next section we discuss outline our main contribution. Then, we present details of the proposed placement method, which is based on Self-organization maps. Finally, we will present our experimental results, discussions, and conclusions.

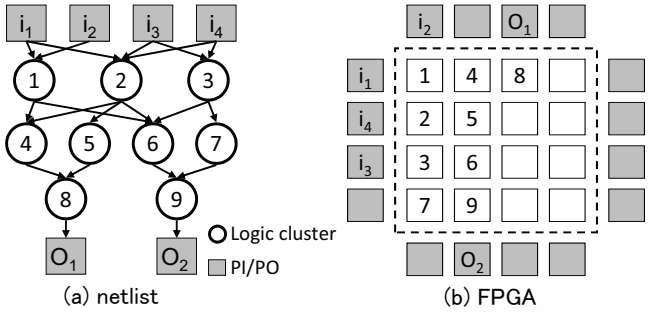

Fig. 1. FPGA placement(a)netlist(b)placement on FPGA

II. proposed placement algorithm

A. Definition of FPGA placement problem

FPGA has many homogeneous logic tile which consists of logic block, connection boxes, switch boxes. The logic tiles $C = \{(i,j)|1 \leq i,j \leq N\}$ on FPGA are located at a grid point (i,j). A netlist (M,E) is a set of circuit modules and connection between modules. $M = \{m_1, m_2, ..., m_M\}$ is the set of the circuit modules in the netlist. Given the set of logic tiles C and netlist (M,E). Find an injective mapping function Φ considering something cost function such as total path length.

$$\Phi : M \to C, m \to c \tag{1}$$

B. Self-organaization maps

A SOM consists of neurons organized on a regular two-dimensional grid. The number of neurons may vary from

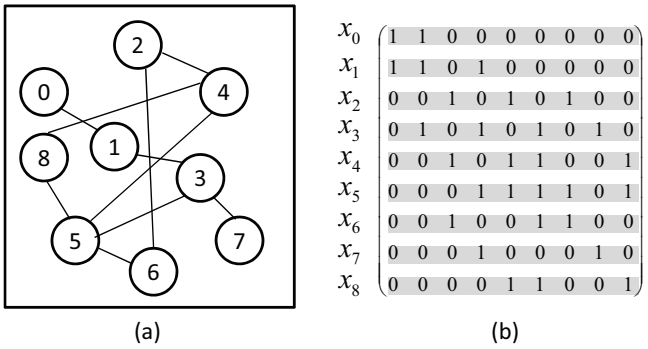

Fig. 2. Exsample of input vector sets(a)circuit(b) connection matrix

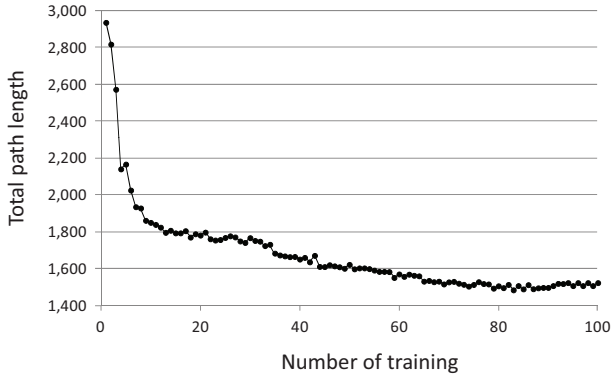

Fig. 3. Relation ship both number of learning and total path length in C432

a few dozen up to several thousand. Each neuron is represented by a n-dimensional weight vector $m_i(t) = [m_{i0}, m_{i1}, ..., m_{i(n-1)}]$, where n is equal to dimension of the input vectors. The neurons are connected to adjacent neurons by a neighborhood relation, which dictate the topology. The weight vectors are set to small random numbers initially and are updated at each iteration. Each time, an n-dimensional input vector $x_j(t) = [x_{j0}, x_{j1}, ..., x_{j(n-1)}]$ is applied to the network and all output nodes compare their weight vectors with the input vector. The neuron whose weight vector is closest to the input vector x_j is called the Best-Matching Unit(BMU), denoted here by c:

$$m_c(t) = \underset{i}{\operatorname{argmin}} \mid\mid x_j(t) - m_i(t) \mid\mid \tag{2}$$

The SOM update rule for the weight vector unit i is:

$$m_i(t+1) = \frac{\sum_{j=1}^{n} h_{c,i}(t) x_j(t)}{\sum_{j=1}^{n} h_{c,i}(t)} \tag{3}$$

where $h_{c,i}$ is neighborhood function[2]. This update method is called batch type training. In each training step, the data set is partitioned according to the voronoi regions of the map weight vectiors, ie. each data vector belongs to the data set of the map unit to which it is closest.

C. Modification for placement

We modify two points of SOM algorithm for FPGA placement. First, binary connection matrix is used as a input vector. A typical connection matrix is shown in Fig.2. Each column represents a vector of inputs associated with a cell. A 1 indicate interconnection between cells and we can obtain nine input vector sets. This connection matrix is used to place the cell with forced adjacency constraint. Next, in order to select BMU, we take weighted dot product. For example, in the case of input vector x_0, weighted dot product calculates following:

$$(x_0 \cdot m_i) = k \times x_{00} \times m_{i0} + \sum_{n=1}^{M-1} x_{0n} \times m_{in} \tag{4}$$

TABLE I
TOTAL PATH LENGTH, CHANNEL WIDTH AND DELAY REQUIRED TO PLACE AND ROUTE CIRCUITS.

Circuit	tool	total path length	channel width	delay(ns)
C432	SOM Place	1494.0	12.0	13.42
	VPlace(SA)	1207.0	12.0	10.05
s386	SOM Place	841.6	9.6	4.56
	VPlace(SA)	794.0	10.0	3.61
s344	SOM Place	485.0	6.8	3.59
	VPlace(SA)	428.6	6.0	3.00
s400	SOM Place	732.6	7.2	3.49
	VPlace(SA)	649.0	6.4	3.03
s382	SOM Place	696.2	6.0	3.74
	VPlace(SA)	608.0	6.0	3.76

III. Experimental Results

In this section we compare total path length, the minimum number of tracks per channel, delay required for a successful routing by two placement tools on a set of 5 MCNC benchmark circuits. The total path is defined by using manhattan distance. All the results in this section were obtained with a logic block consisting of a 4-input LUT plus a flip flop. The initial neighborhood is set so as to cover about 70% of the neurons, and then the coverage is decreased linearly with time to 0 after 100 iterations. Fig.3 shows the transition of learning of our method in circuit C432. Total path length can converge at about 80% of total learning. All the results in table 1 are obtained by VRoute[1] routing a placement produced by VPlace[1] and our method. Although total path length is worse than VPlace, delay and channel width are comparable.

References

[1] A. Marquardt, V. Betz and J. Rose, "Timing-driven placement for FPGAs," Proc. of the 2000 ACM/SIGDA eighth international symposium on Field programmable gate arrays(FPGA), pp.203–213, Feb. 2000

[2] T. Kohonen, "The self-organizing map ," Proc. of IEEE, Vol.78, pp.1464–1480, Sep. 1990