A graph theoretical approach for a multistep mapping software for the FACETS project

Karsten Wendt
Technische Universität Dresden
ETIT/IEE - HPSN
D-01062 Dresden
GERMANY

Matthias Ehrlich
Technische Universität Dresden
ETIT/IEE - HPSN
D-01062 Dresden
GERMANY

René Schüffny
Technische Universität Dresden
ETIT/IEE - HPSN
D-01062 Dresden
GERMANY

Abstract: This paper discusses the mapping problem, i.e. the assignment of complex biological networks to a special hardware system. The problem is modeled as a map of two disjoint sets. Further a graph model is introduced to overcome the lack of adequate system representations for the mapping software framework, which controls the complex mapping process. The characteristics of the graph model are described and possible applications for both the biological networks and hardware systems are shown. Consequently the mapping is expanded to a graph relation. After a short consideration of implementation aspects the mapping process, i.e. the handling with the graph models during creation of the mapping itself is represented.

Key–Words: Graph Model, Mapping, FACETS, Software Framework

1 Introduction

The project *Fast Analog Computing with Emergent Transient States* - FACETS [1] aims at exploring various computational aspects of biological neural networks. The motivation is to emulate complex pulsing neural systems with up $5 \cdot 10^4$ neurons and $200 \cdot 10^6$ synapses with a speed-up between 10^3 and 10^4 compared to biological real-time.

This encompasses the development of a massive parallel neural hardware system consisting of configurable pulse coupled neural network IC as well as a software that allows for the configuration of said system in reasonable time.

The challenge of the work accomplished so far is to make the configuration and mapping process controllable by using a flexible framework of models and algorithms.

A short overview of the dimensions of the reconfigurable FACETS system and the resulting mapping, hence configuration problem shall be given.

It is followed by a short overview of the reconfigurable FACETS system which is currently in development in section 2. Section 3 considers the so called mapping problem and mapping framework of the FACETS project. Then a graph model with its biological and hardware application is introduced in section 4. The paper is completed by a short summary of implementation issues in section 5 and a presentation of the entire mapping process based on the graph models in section 6.

2 Current State of the FACETS System from the Mapping Viewpoint

The FACETS system [1] consists of analog modeled IF (integrate-and-fire) neurons and synapses, which implement a STDP (spike-time-dependency-plasticity) mechanism [2]. The neuron's and synapses' behavior is defined by a set of configurable parameters. A parameter set is shared by several neurons or synapses respectively.

Functional blocks, so called *HICANNs* (High Input Count Analog Neural Network) [2], contain up to 2^{17} (131072) synapses each, which are shared by a configurable set of hardware neurons equally, so that a single neuron can be connected to up to 2^{14} (16384) synapses. The input signals, i.e. neuron spikes from other HICANNs, and the output signals of each HICANNs are encoded in a digital time-continuous network, so called *layer-1* network. It consists of buses, which carry the neuron spikes of up to 64 neurons each.

The input of each HICANN is decoded configurably statically from up to $n_{maxInput}$ layer-1 buses. Therefore the number of different input signals is limited to $64 \cdot n_{maxInput}$ per HICANN and so also for a single neuron. The analog HICANN's output is transformed by a 64-to-1 priority encoder to digital pulse

[1]status quo of September 2007

[2]currently in development at the Kirchhoff Institut of Physik, Ruprecht-Karls-Universitaet, Heidelberg, Germany

signals of 1 up to 8 layer-1 buses, which are fed in the layer-1 network at predefined positions.

A grid of up to 400 HICANNs are arranged on an entire *wafer*, e.g. these blocks have to be placed and connected on a single cohesive plane. $n_{l1horizontal}$ horizontal and $n_{l1vertcial}$ vertical layer-1 buses between each HICANN provide one part of the intra-wafer communication system. At each cross point can be found a *crossbar*, which realizes the connectivity between horizontal and vertical buses. The crossbars are populated only sparsely with switches to reduce the amount of memory. At the borders of each HICANN *repeaters* are placed, connecting the layer-1 network of each adjacent HICANN to stop or relay single buses. The input selection of each HICANN is done by an also sparsely populated *select switch*.

Finally the FACETS system consists of several wafers connected by an additional digital not time-continuous package-based communication system, so called layer-2 network. The signal output points of each HICANN are connected with a *DNC* (Digital Network Core), which transforms necessary layer-1 signals into packages, including time and destination. Then the packages are routed, passing *FPGAs*, which provide access to the inter-wafer bus, to their entry points and are decoded back by a DNC to layer-1 pulses. Thus long-ranged intra-wafer and inter-wafer communication is realized. The configuration of the DNCs and FPGAs is done statically by routing tables. [3]

3 Mapping Problem

Obviously to see, that to set up this hardware system to simulate a specific neural network is a difficult task. Generally, given a biological system consisting of elements of B and a hardware system consisting of elements of H, the assignment of B to H can be presented by a map m:

$$m : B_p \rightarrow H_p \tag{1}$$

whereas

$$B_p \subseteq B \tag{2}$$

and

$$H_p \subseteq H \tag{3}$$

The different constraints of this map can be described informally as to

- minimize not realized neurons and synapses
- minimize violations of parameters
- minimize violations of timing
- maximize hardware efficiency
- realize adequate mapping time

Formally it can be described as following, differing in hard and soft constraints: Given 2 functions, so called cost functions:

$$c_{h,s} : (B, H, B_p \rightarrow H_p) \rightarrow \mathbb{R} \tag{4}$$

a mapping m has to be found so that:

$$c_h(B, H, m) = 0 \tag{5}$$

$$c_s(B, H, m) \rightarrow min. \tag{6}$$

Then the task can be characterized as a multi-objective optimization problem [6] [7], which makes a set of adequate algorithms, e.g. genetic searches, necessary. The algorithm's optimization is based on efficient cost functions, which evaluate the mapping with regard to the given constraints. Again the cost functions depend on flexible models of the biological and hardware systems. The complexity of this problem requires a software framework, which decouples the involved elements as shown in figure 1 as far as possible.

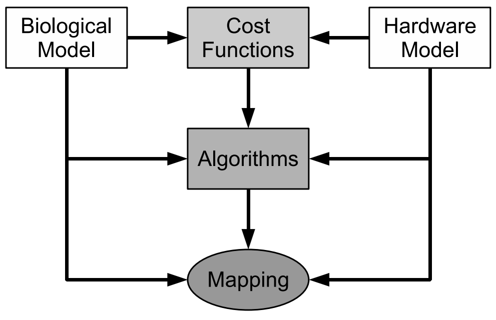

Figure 1: Mapping Framework Scheme

4 Modeling

4.1 Graph Model

As an universal model of data representation for the entire mapping framework and on which the cost functions and the algorithms will base, a special graph model was chosen. Generally, a graph is a pair of 2 finite sets V and E, which represent a set of nodes and a set of edges. V and E apply to the constraints: $V \neq \varnothing$ and $V \cap E \neq \varnothing$. Furthermore every graph is related to a map Ψ with $\Psi : E \rightarrow V \times V$. [4] [5]

The used graph model G consists of nodes $v_i \in V$, which can hold a name or a value as a data item respectively. It contains 3 types of directed edges $e_i \in E$:

- *hierarchical edges* modeling a hierarchy of nodes, i.e. to represent a container-component relationship
- *named edges* modeling a certain relationship between 2 nodes
- *hyper edges* connecting a named edge with a node, i.e. to model a detailed description of a node-node relationship.

To clarify the role of the graph model components, they are shown exemplarily in figure 2. Thus the graph model can be characterized as a special form of a directed hypergraph. [4] [5]

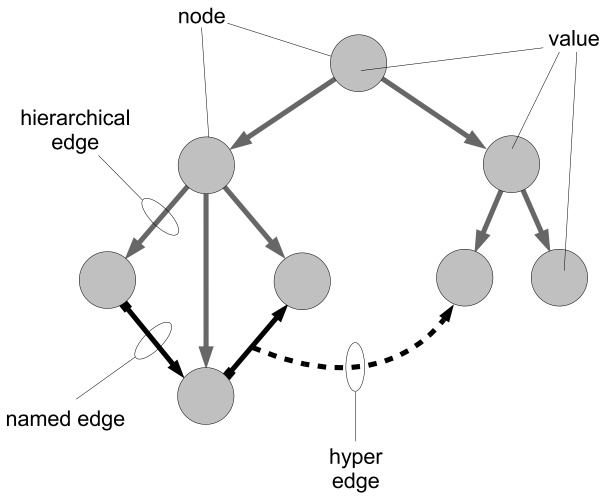

Figure 2: Graph Model

So it is possible to create flexible models in respect to the biological network's and hardware system's complexity. Because of its structure the model is fully navigable, which means every point, i.e. every node or edge is reachable from every position in the graph. By comparison a non graph model with 2 disjoint data sets, e.g. the synaptical connectivity as a (sparse) matrix or a list separated from the neural parameters as a vector, offers no direct access to the relations of these data sets.

Thus it provides an universal data interface for the optimization algorithms and cost functions, which transform needed data into adequate formats and store them back into the graph model making the data accessible for other components. Furthermore the decentralized structure makes the model suitable for massive parallelization.

On the other hand this kind of data representation consumes more memory and more build up time than more compact models by way of comparison.

4.2 Biological Model

The biological systems, which have to be mapped to the FACETS hardware, can be considered as networks of neurons and synapses. Each neuron is connected to a number of synapses and furthermore characterized to a set of parameters. Each synapse connects a source and a target neuron and is also assigned to parameters.

The biological graph model is referred to as $G_B = (V_B, E_B)$. To gain access to a simple structure all neurons and neural as wall as synaptical parameters are corresponded hierarchically to different nodes. A synaptical connection is modeled as a named edge, where the name classifies the edges role. The parameter assignments are also done by named connections and hyper connections for the synapses. A sample of this global view is shown in figure 3.

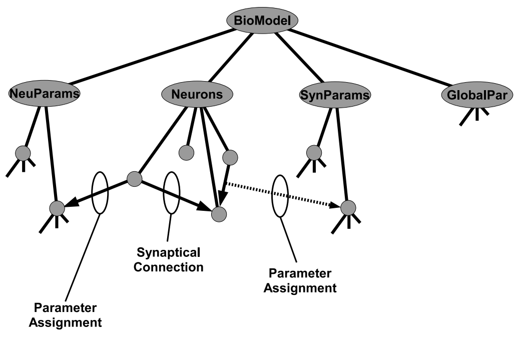

Figure 3: Hierarchical view of the biological model G_B

Given this structure a neural network can be automatically transformed into graph model representation G_B, as shown in figure 4. There, a system of 3 neurons $N^{0..2}$ with 2 parameters set $P_N^{0,1}$ and 5 synapses $S^{0..4}$ with also 2 parameter sets $P_S^{0,1}$ connecting the neurons are modeled as a graph described above. Each neuron and each parameter set is realized by a node. The synapses are formed by a named connection, where the connection names are not shown in this figure. The elements are assigned by named connections or hyper connections to their parameter nodes.

4.3 Hardware Model

The hardware system as described in section 2 can be formed as a hierarchical structure of hardware elements. The hardware graph model is referred to as $G_H = (V_H, E_H)$. A reduced model is shown in figure 5. On top level the FACETS system consists of a digital bus connecting several wafers. A single wafer contains a grid of HICANNs, connected by abstracted connection elements (CE). A HICANN in the

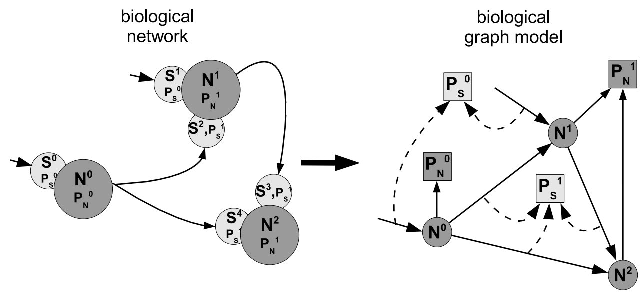

Figure 4: Simplified biological network transformed into a graph model G_B

last level holds a number of hardware neurons.

Furthermore each level also contains data about the optimization algorithm and the cost function to use, which abstracts the constraints of the hardware to an algorithm compatible form. The information is used by the mapping framework to accomplish the mapping process and is shown only for the highest node in the figure.

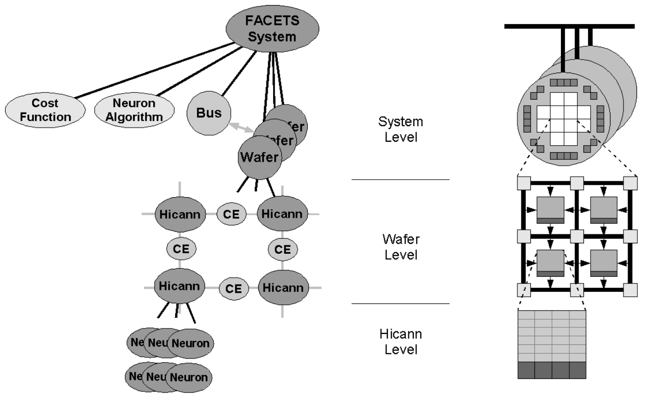

Figure 5: Hierarchical view of the FACETS hardware model G_H

As an example the hardware system can be modeled more detailed at the wafer level. In figure 6 a single HICANN, consisting of neurons $N^{0,1}$ and synapses $S^{0..3}$ characterized by the parameter sets P_N^0 and $P_S^{0,1}$, is embedded in layer-1 relevant hardware elements. It receives its spike signals via a select switch SW^1 from a layer-1 section $l1^1$. Its output is transformed by a *WTA-encoder* WTA^0 and can be routed through a crossbar CB^0. All these elements are characterizable by their parameter sets.

The transformation step to a graph model representation G_H converts all hardware elements to graph nodes and connects them to their parameters via named nodes. Only the simple signal transporting components like synapses and layer-1 buses are also modeled by named edges assigned to parameters via hyper edges.

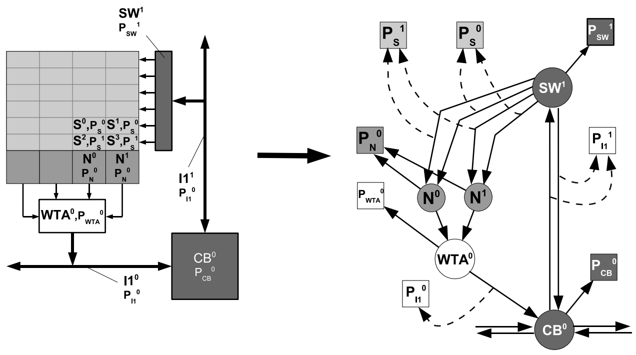

Figure 6: Simplified hardware system transformed into a graph model G_H

4.4 Graph Mapping

Consequently the mapping as a map from the set B to the set H, see section 3, must be expanded to a graph relation r_m:

$$r_m : V_B \times V_H \tag{7}$$

with

$$V_B \times V_H := \left\{ \begin{matrix} (b, h)| \\ (b \in V_B) \wedge (h \in V_H) \end{matrix} \right\} \tag{8}$$

whereas V_B is the set of nodes in G_B and V_H is the set of nodes in G_H. Thus the mapping relation r_m can be generalized to a set of tuples

$$r_m = \begin{cases} (b^0, h_0^{i_0}) \\ (b^0, h_0^{i_1}) \\ \dots \\ (b^0, h_0^{i_{max}}) \\ (b^1, h_1^{j_0}) \\ \dots \\ (b^n, h_n^{k_{max}}) \end{cases} \tag{9}$$

which assign every mapped node of the biological model to one or more nodes of the hardware system. In the graph model the mapping is done by named edges, so called mapping edges representing the assignment. An example is shown in figure 7, where 2 neurons are mapped to 2 different HICANNs and a synaptical parameter set is also mapped to both hardware nodes.

5 Implementation

The graph model was implemented in C++. To reduce the memory consumption to be able to model large

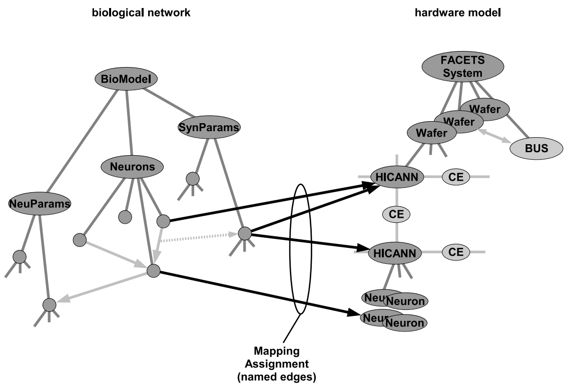

Figure 7: Mapping representation in the graph model

models, the classes were implemented light-weighted. The model consists of 2 elements, *nodes* and *connections*, which stand for the named edges. A node contains:

- a value, which holds the data item of the node
- a pointer to its superordinate node, which models the hierarchical edges backward
- a list of pointers to its subordinate nodes, which models the hierarchical edges forward
- a list of pointers to outgoing and incoming connections, which models the named connections forward and backward
- a list of pointers to connections, which models the hyper edges backward

A connection contains:

- a value or a name ID respectively, which holds the semantic meaning of the edge
- a pointer to the target and the source node, which models the named edges forward and backward
- a pointer to a hyper-connected node, which models the hyper edges forward

Thus the memory consumption of a connection is always 26 Bytes and a node consumes at least 96 Bytes depending on the size of the data item and the length of the lists.

For example a biological network of 10^4 neurons and $13.5 \cdot 10^6$ synapses causes a memory consumption of 866.4 MBytes in total, which means every neuron consumes 38.1 KBytes in average, the synaptical connections and their parameter assignments allocate 35.9 Bytes per synapse.

Furthermore the graph model provides an interface for the optimization algorithms and cost functions to interact with the model. On the one hand it gives access to functions to set up and control models, i.e. to node and connection creation and deletion function. On the other hand it offers functions to navigate inside the graph model, i.e. filter all subordinate nodes of a given node in regard to their values, find all outgoing or incoming connections with the same name ID or filter the source or targets of given connections regarding a given value. It is possible to combine these access functions to locate certain positions via a ”path” along the edges and nodes, which means to navigate inside the model.

6 Mapping Process

The global flow of the mapping process is build on a recursively called function *MappingStep*, which performs the mapping of a partial biological model with an adequate optimization algorithm and cost function to the current position, i.e. to subordinate nodes of current node in the hardware model. Figure 8 shows the global mapping process as a structure chart.

```
Global Model Creation
    Create biological model $G_B$
    Create hardware model $G_H$
    MappingStep($G_B$,$v_H^*$)
RoutingStep
Delete global models

MappingStep($G_B^v$,$v_H$)
    Instantiate optimization algorithm $OA_{vH}$($G_B^v$,$v_H$)
    Instantiate cost function $CF_{vH}$($G_B^v$,$v_H$)
    Execute $OA_{vH}$ with $CF_{vH}$
    Expand mapping of $G_B^v$
foreach $v_H$' as subordinate node of $v_H$
        Extract subordinate biological model $G_B^{v'}$
        Execute MappingStep($G_B^{v'}$,$v_H$')
        Update mapping from $G_b^{v'}$ in $G_B^v$
        Delete $G_B^{v'}$
```

Figure 8: Global mapping process

6.1 Global Model Creation

The first stage of the mapping process is the creation of the hardware and the biological model. While the hardware model creation is a trivial task to build up the representation of the biological model is not.

The neural networks textual representation follows a self defined neural syntax similar to a context free syntax of a formal language, enabling the possibility to use a parsing process for constructing an internal graph representation.

To generate the structure of the biological network a scanner reads the neural net lists input stream and identifies the tokens. The subsequent low level parsing process recognizes the neural semantic of the tokens read according to our self defined syntax and returns a collapsed statement. The information extracted is stored in a statement- or ”root”-tree from which the global syntactic tree representing the biomodel is finally build.

This split-parser approach allows separate optimization of the scanning/low level parsing process and the high level parser. The use of parsing methods enables an implicit error check during net list read-in and also handles the semantic recognition.

Finally 2 models as described in section 4.2 and 4.3 are created.

6.2 Mapping Step

Initially this function is called with global biological model G_B and the highest hardware node v_H^* as arguments.

First, as in the hardware model defined, an instance of the current optimization algorithm OA_{v_H} and an instance of the current cost function CF_{v_H} for this hardware node is created. These 2 components use the interface as described in section 5 to gain access to the models and transform the data into adequate computation formats.

Then the algorithm with the cost function is executed and creates a mapping between the graph models as shown in section 4.4. Subsequently the mapping is expanded to dependent nodes, e.g. parameter sets, to complete the mapping for this hardware element v_H.

Finally for each subordinate node v'_H a partial biological model $G_B^{v'}$ is extracted, which contains only biological elements which are relevant for the subsequent mapping steps, holding a connection to their origins. Recursively the function MappingStep is called with the model $G_B^{v'}$ and v'_H as arguments. It creates a more detailed mapping for the subordinate nodes of v_H. After that the specified mapping is updated to the current biological model G_B^v, i.e. the mapping edges are redirected to lower level hardware nodes, and the partial model is deleted.

The return of the highest MappingStep call follows a routing step which calculates the routing tables of the hardware, depending on the placement of the biological elements. This will be outlined in a later publication.

7 Conclusion

A hypergraph classified graph model consisting of 3 different edge types was introduced. It was shown that it is capable to model complex biological and hardware systems with the same underlying structure. The graph models were tested and developed with various biological networks, reaching from 10^2 to $2 \cdot 10^5$ neurons and different hardware systems, which mimic the described FACETS system.

Its implementation was optimized regarding access speed, memory usage and mapping process functions, see chapter 6.2. The graph model offers a basis to develop optimization algorithms independently, efficient cost functions and more complex models for the introduced mapping framework. Further it is highly capable of massive parallelization to achieve additional performance. The next task is to examine the memory behaviour in more detail, investigate the performance of single mapping steps and extend the graph model interface for further algorithms, cost functions and external routing functions.

Acknowledgements: The research project FACETS is financed by the European Union as Integrated Project (Nr. 15879) in the framework of the Information Society Technologies program.

References:

[1] K. Meier - Fast Analog Computing with Emergent Transient States in Neural Architectures, *Integrated project proposal, FP6-2004-IST-FET* Proactive, Part B. - Kirchhoff Institut of Physik, Ruprecht-Karls-Universitaet, Heidelberg 2004.

[2] J. Schemmel, A. Gruebl, K. Meier and E. Mueller - *Implementing Synaptic Plasticity in a VLSI Spiking Neural Network Model*, Kirchhoff Institut of Physik, Ruprecht-Karls-Universitaet, Heidelberg 2006.

[3] M. Ehrlich, C. Mayr , H. Eisenreich , S. Henker, A. Srowig, A. Gruebl, J. Schemmel, and R. Schueffny *Wafer-scale VLSI implementaions of pulse coupled neural network* - International Conference on Sensors, Circuits and Instrumentation Systems SSD07, Hammamet- Tunisia, 2007

[4] R. Diestel - *Graph Theory*, Springer 2005.

[5] M.I. Jordan - *Graphical Models*, Computer Science Division and Department of Statistics, University of California, Berkeley, California 94720-3860, USA 2004.

[6] D.F. Jones - *Multi-objective meta-heuristics: An overview of the current state-of-the-art*, School of Computer Science and Mathematics, University of Portmouth, UK 2001.

[7] E. Zitzler - *Comparison of Multiobjective Evolutionary Algorithms: Empirical Results*, Department of Electrical Engineering, Swiss Federal Institute of Technology, Switzerland 2000.