Mapping Very Large Scale Spiking Neuron Network to Neuromorphic Hardware

Ouwen Jin, Qinghui Xin, Ying Li, Shuiguang Deng, Shuibing He, and Gang Pan.

Zhejiang University, Hangzhou, China

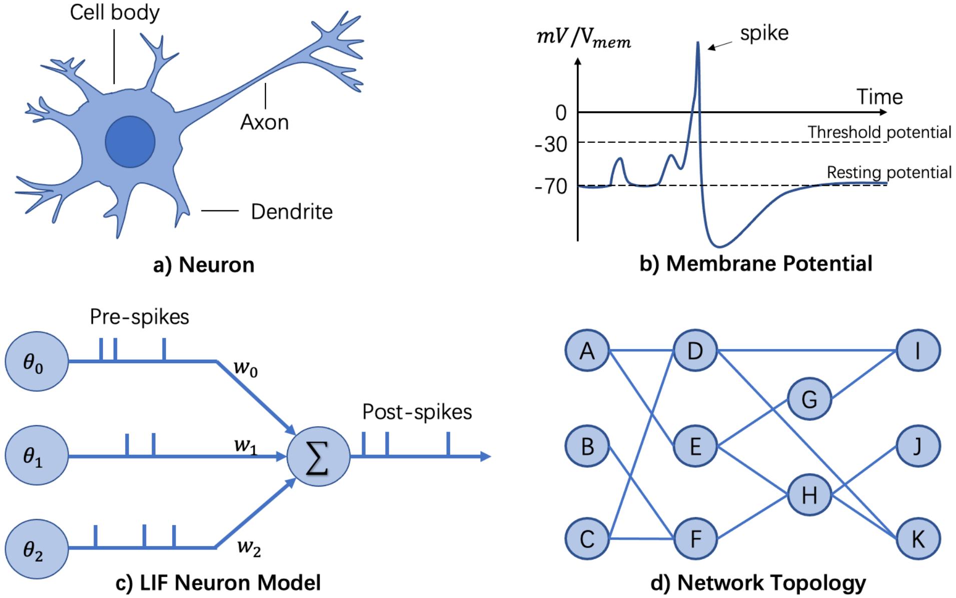
ASPLOS '23, Volume 3 (ASPLOS '23), March 25–29, 2023, Vancouver, BC, Canada. ACM, New York, NY, USA.https://doi.org/10.1145/3582016.3582038

Outline

- Background
- Method
- Experiments
- Conclusion

Background

Neuromorphic Computing

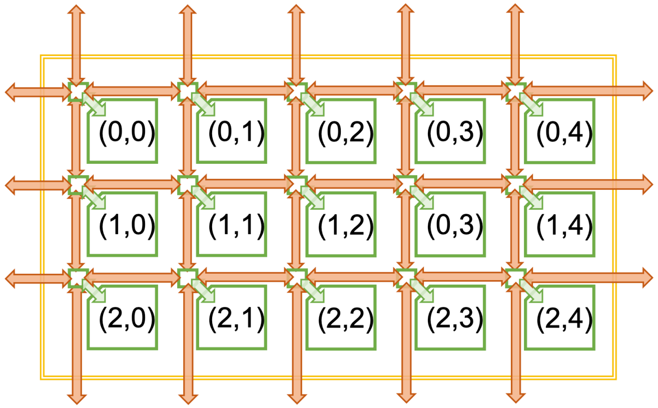


Imitating the brain in terms of neuron and synaptic connection models, **Spiking Neural Network (SNN)** features rich spatial-temporal information and high biological plausibility.

Background

Neuromorphic Hardware

A neuromorphic hardware platform is a computer system specifically designed to implement SNN applications.
They using a large number of specially designed neurosynaptic cores to simulate neurons dynamics in parallel.

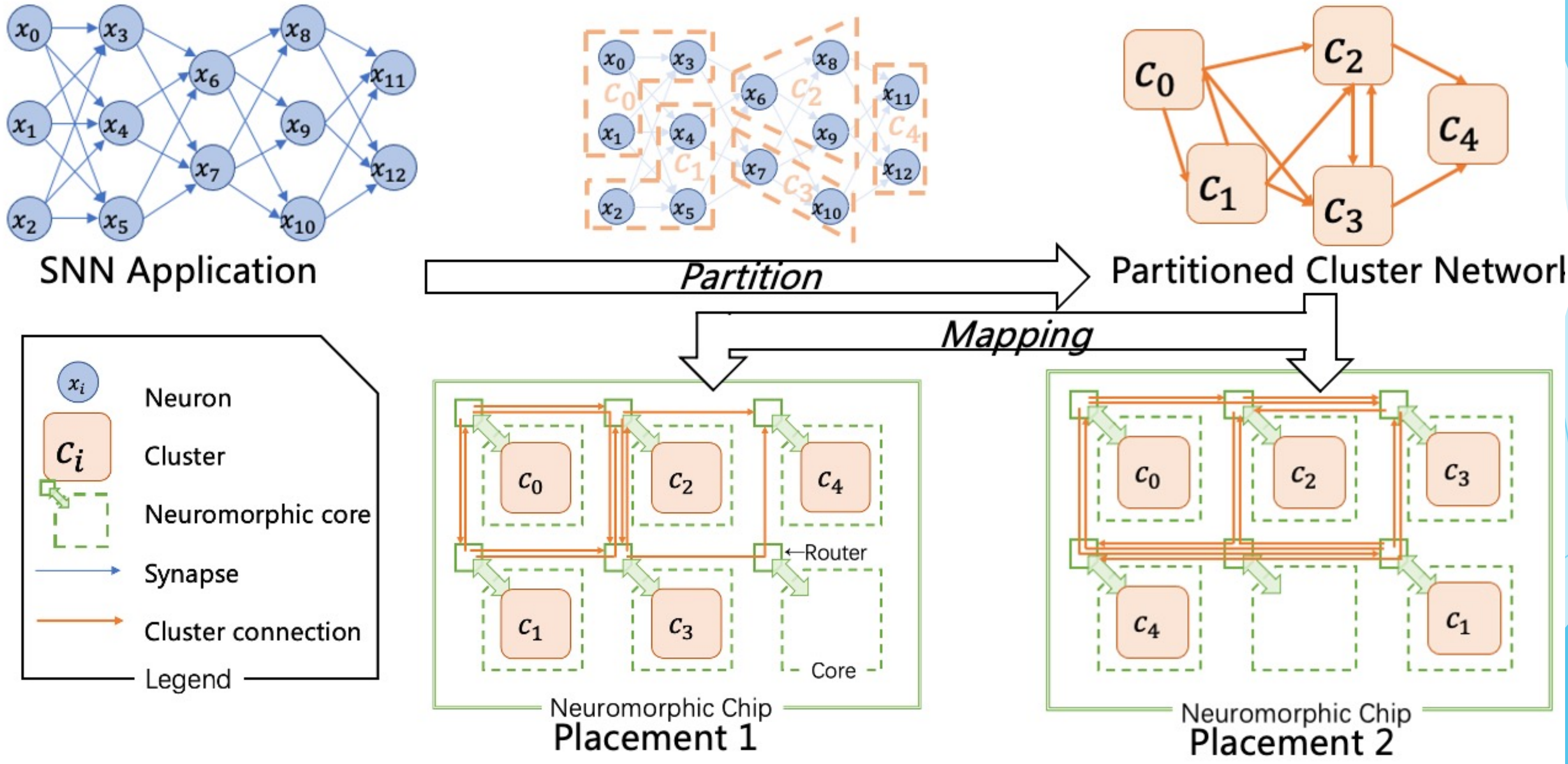


Common solution: 2D mesh

Neuromorphic Hardware: Darwin Mouse

Background

Spiking Neuron Network (SNN) mapping problem



Background

Neuromorphic Hardware

Multiple new neuromorphic computing platforms, including Loihi 2, SpiNNaker 2 and Darwin 3, aim to reach **tens of billions** of neuron capacities.

Table 1: Capacity of several neuromorphic hardware platforms

	DYNAPs [24]	BrainScaleS [30]	Loihi [7]	SpiNNaker [11]	TrueNorth [8]
# Neurons/core	256	512	128	1000	256
# Synapses/core	16K	128K	500K	2K	262K
# cores/chip	1	1	1024	18	4096
# chips/system	4	8192	768	1M	64
High-performance system					
# Neurons	1K	4M	100M	1B	64M
# Synapses	65K	1B	100B	200B	1T

Background

related work

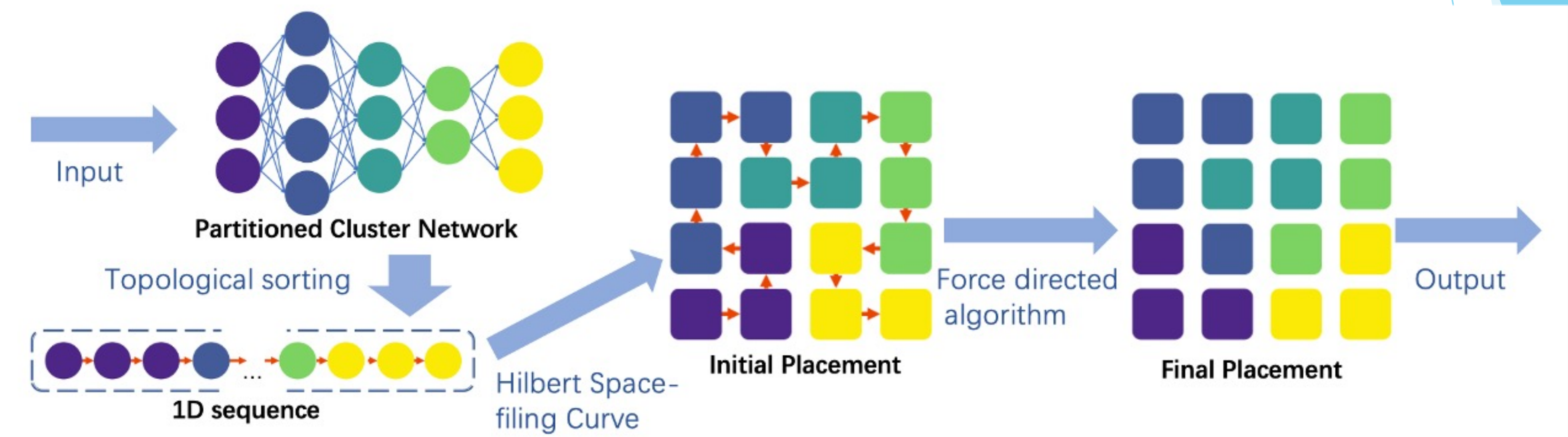
- Greedy Algorithm
- Heuristic search algorithms: Particle Swarm Optimization, Simulated Annealing, Genetic Algorithm
- Integer Linear Programming (ILP)

Motivation

Existing algorithms lack scalability and are unable to efficiently map large scale SNNs to neuromorphic Hardware

Method

Diagram of the proposed approach



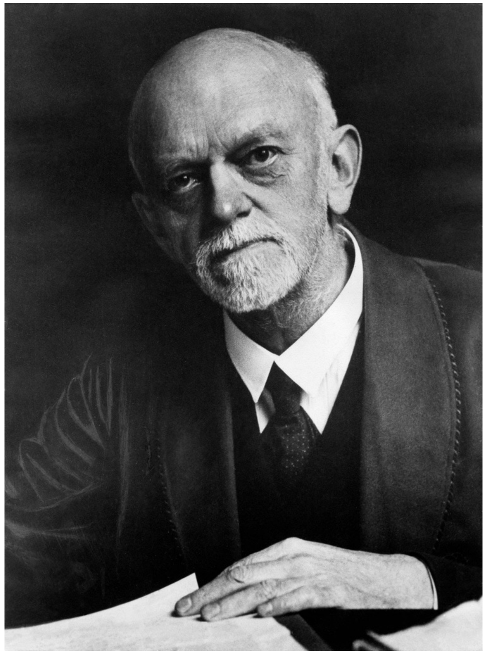
Contribution

- We are the first to apply Hilbert Space-filing Curve (HSC) to the SNN mapping problem.
- We propose the Force Directed (FD) algorithm
- We evaluate our approach with a large scale of 4 billion neurons and millions of cores on a general neuromorphic hardware model.

Method

Hilbert Space-filling Curve (HSC)

The Hilbert curve is a continuous fractal space-filling curve first described by the German mathematician David Hilbert. It provides a mapping relationship between 1D and 2D space.



David Hilbert

Method

Properties of HSC

- Infinity
- Provide dataflow layout
- Locality

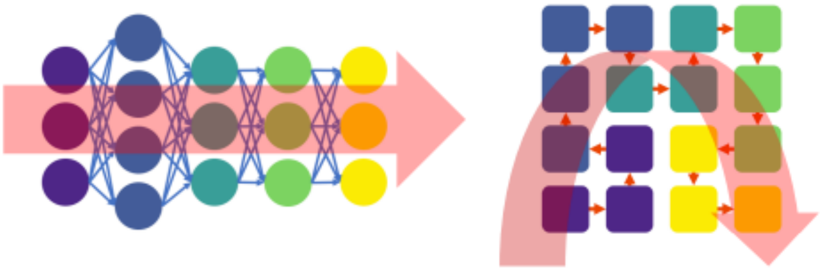


Figure 5: Data flow layout

Locality

- Locality of HSC

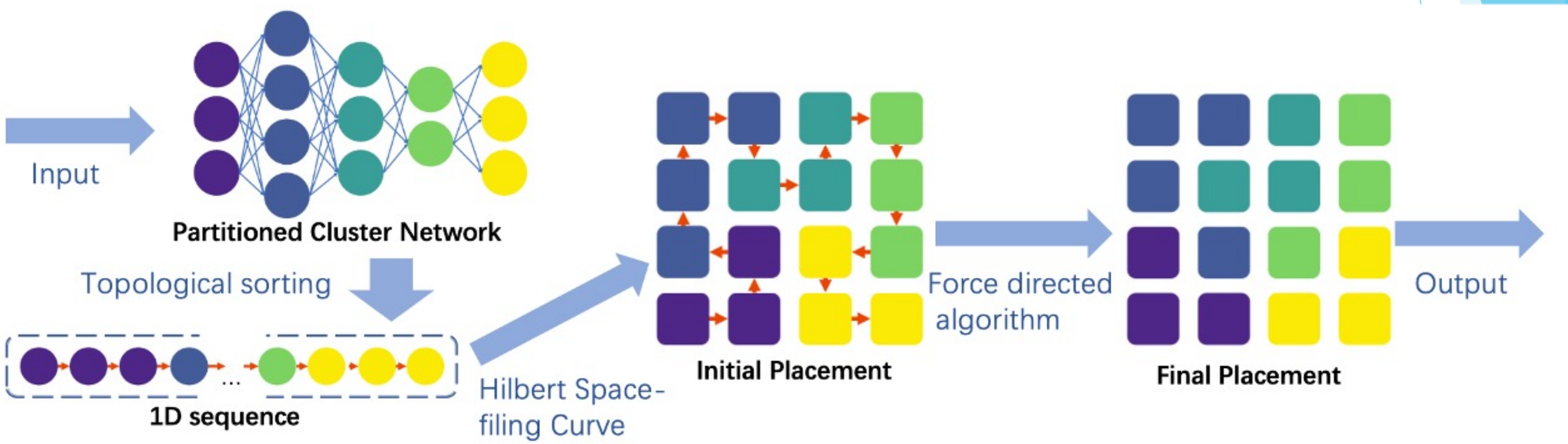
Two data points which are close to each other in 1D space are also close to each other after folding.

- Locality of SNN

Neurons are only connected to a few other neurons locally instead of being widely connected in the whole network.

Method

Force Directed (FD) algorithm

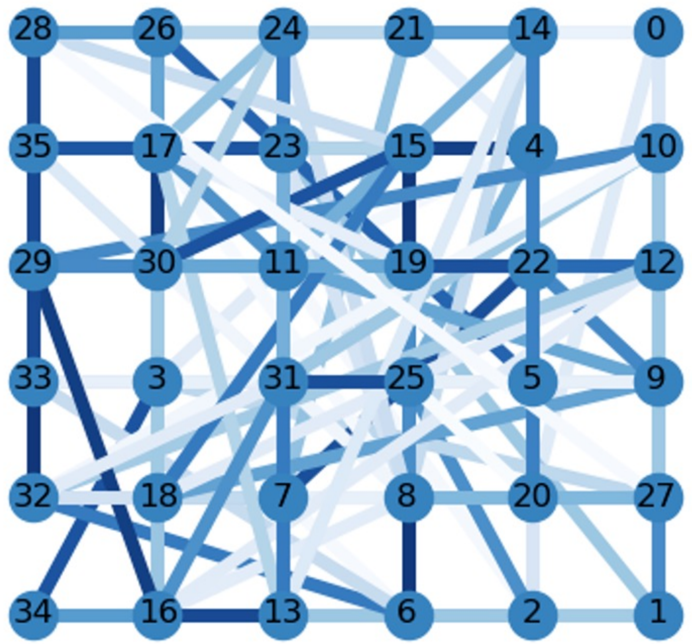


The HSC provides a placement that only maps clusters at a macro level, so there is a large room for local optimization. Therefore, we propose the FD algorithm to **finetune** the placement provided by HSC.

Method

Force Directed (FD) algorithm

The main idea of the FD algorithm is to regard clusters as particles on a 2D plane and the connection between clusters as the tension



Method

System modelling

Potential energy field：

$$U_{c_i}(c_j, P(c_i), P(c_j)) = u(P(c_j) - P(c_i)) * w_p(e_{i,j}). \qquad (18)$$

Cluster energy：

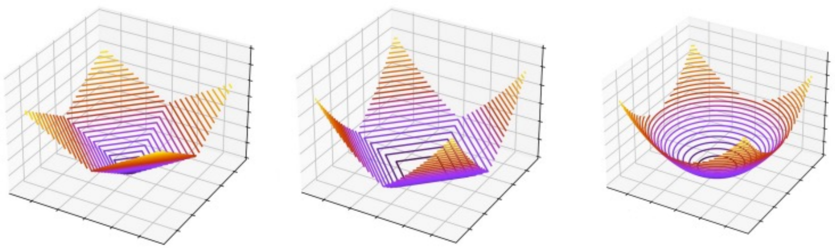
$$E_{c_i} = \sum_{c_j | e_{j,i} \in E_p} U_{c_j}(c_i, P(c_j), P(c_i)) \qquad (22)$$

System energy：

$$\begin{aligned} E_s &= \sum_{c_i \in V_P} E_{c_i} \\ &= \sum_{c_i \in V_P} \sum_{c_j | e_{j,i} \in E_p} U_{c_j}(c_i, P(c_j), P(c_i)) \qquad (23) \\ &= \sum_{e_{i,j} \in E_p} U_{c_i}(c_j, P(c_i), P(c_j)). \end{aligned}$$

Target：

$$\underset{P}{\operatorname{argmin}}\, E_s. \qquad (24)$$



$u_a(p) = |x_p| + |y_p|$ $\quad u_b(p) = (|x_p| + |y_p|)^2$ $\quad u_c(p) = x_p^2 + y_p^2$

Figure 7: Different potential energy fields

Method

Algorithm modelling

Force：

$$
\begin{aligned}
& Force_{c_i,d} \\
& = E_{c_i} - E'_{c_i} \\
& = E_{c_i} - \sum_{c_j \mid e_{j,i} \in E_p} U_{c_j}(c_i, P(c_j), P(c_i) + \nabla p_d),
\end{aligned} \tag{27}
$$

where

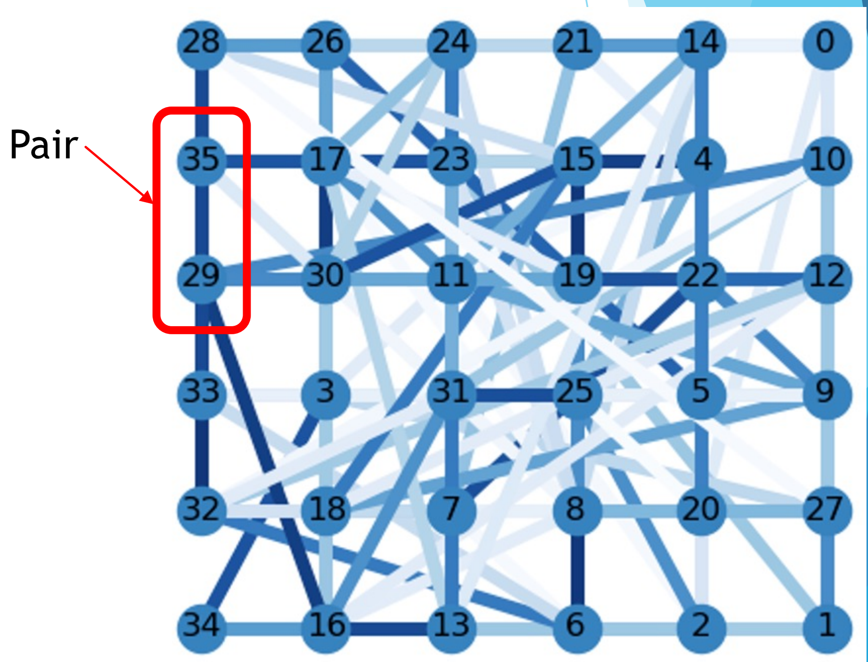
$$d \in \{UP, DOWN, LEFT, RIGHT\} \tag{28}$$

$$\nabla p_d = \begin{cases} (-1,0) & d = UP \\ (1,0) & d = DOWN \\ (0,-1) & d = LEFT \\ (0,1) & d = RIGHT \end{cases}. \tag{29}$$

Tension：

$$Tension_{c_i,c_j} = Force_{c_i,d_{ij}} + Force_{c_j,d_{ji}} \tag{30}$$

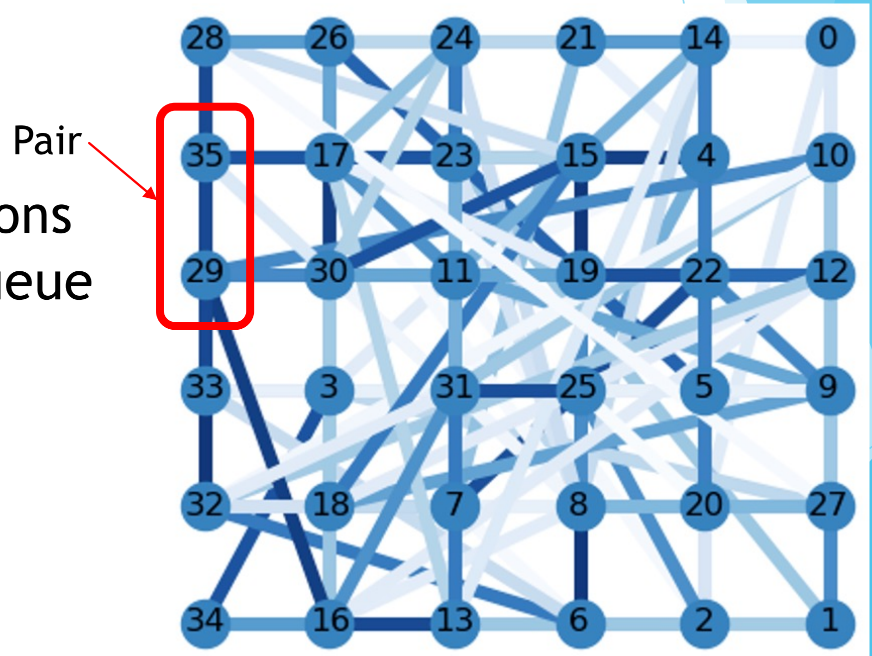
where $\quad ||P(c_i) - P(c_j)|| = 1,$



Method

Work flow

1. Compute tensions of all pairs
2. Sort pairs according to the tensions
3. Swap pairs in the front of the queue
4. Repeat until convergence



Method

Design Choices of FD Algorithm

- Check before the swapping process
  The convergence of the algorithm is guaranteed

- Hyperparameter λ
  Balance the efficiency of the algorithm with the quality of the solution

- Introducing of $L_{affected}$
  Significantly reduce the algorithm overhead when approaching convergence

Experiments

Setting

Comparison Approaches
- The baseline: Randomly mapping
- Truenorth
- DFSynthesizer
- PSO
- Proposed approach

Evaluation Metrics
- Energy consumption
- Average latency
- Maximum latency
- Average congestion
- Maximum congestion
- **Algorithm execution time**

Table 3: Benchmarks

Applications	G_{SNN} Neurons	G_{SNN} Synapses	G_{PCN} Clusters	G_{PCN} Connections	Target Hardware
DNN_65K	65536	805M	16	48	4×4
DNN_16M	16.7M	4T	4096	258048	64×64
DNN_268M	268M	70T	65536	4M	256×256
DNN_4B	4B	1125T	1M	67M	1024×1024
CNN_65K	65536	2M	16	48	4×4
CNN_16M	16.7M	528M	4096	16384	64×64
CNN_268M	268M	8B	65536	262K	256×256
LeNet-MNIST	9118	0.4M	9	19	3×3
LeNet-ImageNet	1.0M	188M	251	2151	16×16
AlexNet	0.9M	1.0B	229	4289	16×16
MobileNet	6.9M	0.5B	1688	37418	42×42
InceptionV3	14.6M	5.4B	3570	117597	60×60
ResNet	28.5M	11.6B	6956	478602	84×84

Table 2: Parameters of target neuromorphic hardware

Parameter	Value
CON_{npc}	4096
CON_{spc}	64K
EN_r	1
EN_w	0.1
L_r	1
L_w	0.01

Experiments

Algorithm execution time

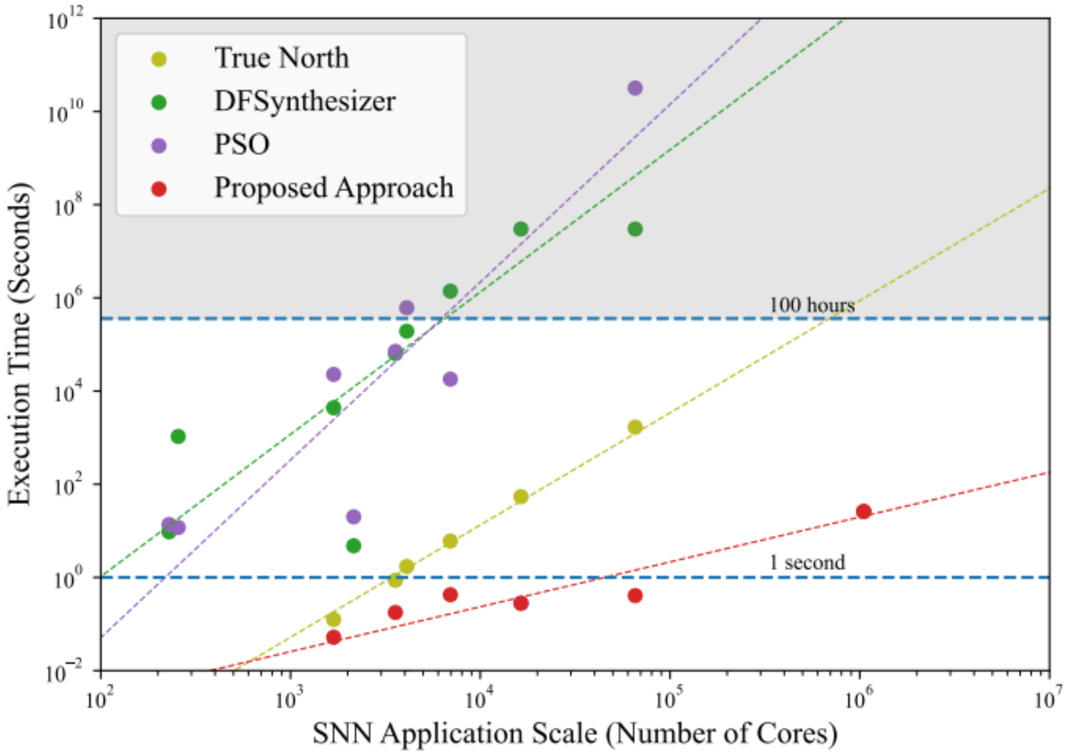


Figure 9: Results on execution time

Experiments

Results on energy consumption

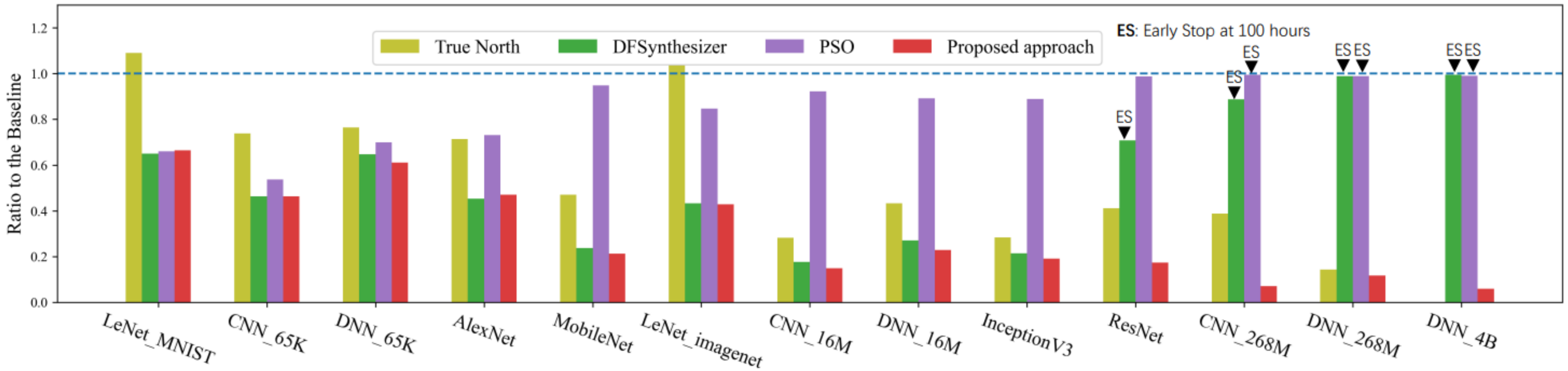
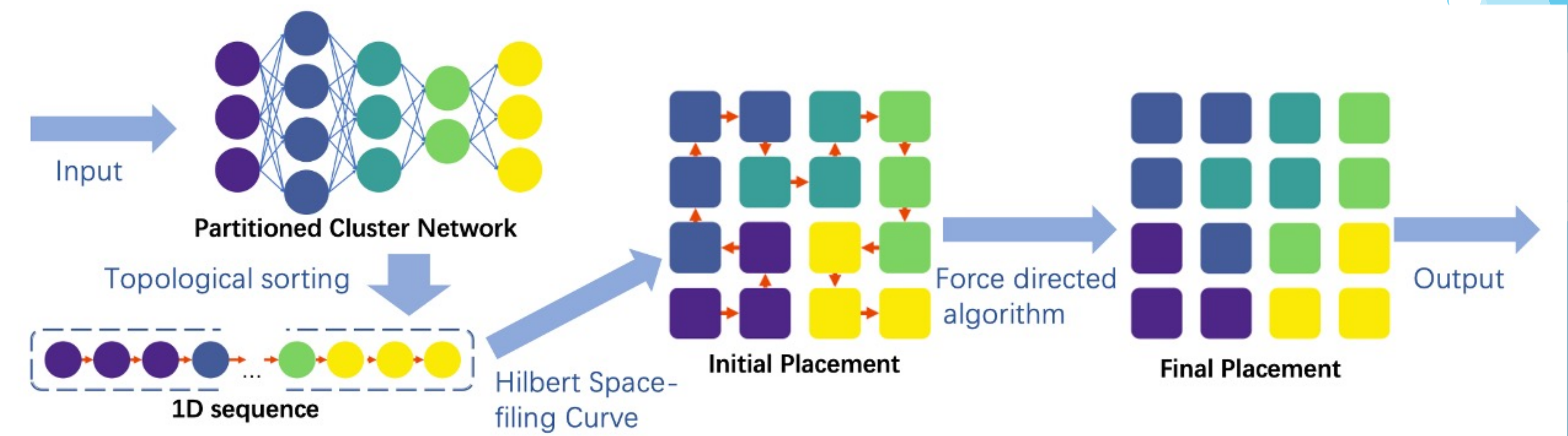


Figure 10: Results on energy consumption

Conclusion

We proposed an approach based on HSC and FD to map very large scale SNN applications to neuromorphic hardware, achieved the SOTA performance in both solving speed and quality of the solution.



Thank you & any question ?