Neuro-NoC: Energy Optimization in Heterogeneous Many-Core NoC using Neural Networks in Dark Silicon Era

Conference Paper · February 2018

DOI: 10.1109/ISCAS.2018.8351580

CITATIONS
3

READS
87

5 authors, including:

Md Farhadur Reza
University of Central Missouri
9 PUBLICATIONS **33** CITATIONS
SEE PROFILE

Tung Le
University of Louisiana at Lafayette
8 PUBLICATIONS **61** CITATIONS
SEE PROFILE

Bappaditya Dey
imec
8 PUBLICATIONS **7** CITATIONS
SEE PROFILE

Magdy Bayoumi
University of Louisiana at Lafayette
582 PUBLICATIONS **3,721** CITATIONS
SEE PROFILE

Neuro-NoC: Energy Optimization in Heterogeneous Many-Core NoC using Neural Networks in Dark Silicon Era

Md Farhadur Reza*, Tung Thanh Le*, Bappaditya De*, Magdy Bayoumi*, Dan Zhao†
Email: {mxr7945, ttl8614, bxd9836, mab0778}@louisiana.edu, zhao@cs.odu.edu
*The Center for Advanced Computer Studies, University of Louisiana at Lafayette, Lafayette, LA, USA
†Department of Computer Science, Old Dominion University, Norfolk, VA, USA

***Abstract*—Due to the end of Dennard Scaling and the rise of dark silicon, it is essential to design energy-efficient heterogeneous NoC under critical power and thermal constraints. The challenge is to determine and configure NoC resources while meeting the application(s) requirements. Because of the large and complex many-core NoC design space (voltage/frequency scaling, link bandwidth, power-gating, etc.), design space becomes difficult to explore within a reasonable time for optimal decision at run-time. Furthermore, reactive resource management is not effective in preventing problems, such as creating thermal hotspots and exceeding power budget, from happening. Therefore, we propose a Neuro-NoC model, which utilizes neural networks learning algorithm to dynamically monitor, predict, and configure NoC resources based on online learning of the system status. Distributed cluster-wise neural network and a global neural network model for resource monitoring and configuration in many-core NoC has been proposed. Simulations demonstrate that Neuro-NoC can predict the global optimal NoC configuration with high accuracy (88%), sensitivity (97% true positive), and specificity (88% true negative).**

I. Introduction

Due to the advancement of transistor technology, hundreds to thousands of cores has now been integrated on a single chip. Transistor power consumption has not decreased in the same way as transistor size due to transistor leakage power limitation, which leads to Dennard Scaling failure [11]. For example, 21% and 50% of the chip resources may have to be dark at 22nm and 8nm process technology respectively [16], [39], [40]. Besides power limitation, a component of the chip can not exceed the manufactured maximum temperature at any instant of time as high temperature increases several failures (e.g., electro-migration) and can cause permanent damage to the chip (e.g., burn). As NoC consumes a significant percentage of chip energy [41], energy efficiency of NoC becomes a critical issue in many-core chip. Dark silicon (power-gating) and dim (voltage/frequency scaling) silicon become the major solutions for current energy limitation in many-core chip. Besides dark/dim silicon, heterogeneous cores (CPU, GPU, accelerator, etc.) are being integrated to increase energy-savings and throughput of chip [26]. As a significant power-thermal source, NoC resources (links and nodes) needs to be configured dynamically (e.g., scaling node-voltage and link-width, power-gating) depending on traffic demands to decrease pressure on meeting chip power and thermal budgets.

NoC design space involves tradeoff of many parameters, such as power and thermal budgets, latency and throughput requirements, resources power-gating, link bandwidth, node voltage, and router buffer-size. As these parameters interrelated to each other, NoC design space increases exponentially with the increase in problem size (many-core and many-tasks). Task-resource allocation in many-core networks can introduce many problems (energy hotspots, load-imbalance, etc.). Because of the limited power and thermal budgets, task-resource mapper needs to find out the required active resources while deactivating other resources. Reactive task-resource allocation may not produce a solution that meets task requirements or may not produce a solution within a reasonable period because of the size and complexity of NoC design space. Reactive task-resource allocation can give a temporary solution, which might lead to problems in network for subsequent mapping of tasks and applications at run-time. Many heterogeneous applications with unpredictable demands can enter the system for running, which makes run-time mapping and configuration a difficult job. The design challenges of many-core heterogeneous (nodes + links) NoCs includes run-time resource monitoring and configuration depending on the applications demands.

Because of heterogeneous application demands, run-time mapping needs to proactively determine the required resources in the next time interval. This motivates us to use neural network to monitor and predict resource utilization in future (in next time interval) and take actions accordingly to prevent any kind of problems, such as resource overloading, in the network. Existing mapping solutions do not consider neural network together with dark silicon during task-resource allocation. Our goal is to configure the heterogeneous NoC using neural network for optimized balancing of the total workloads while meeting the various chip constraints (power, thermal, etc.). We assume on-chip networks is initially partitioned into a uniform size of clusters. Every cluster has a neural network for resource monitoring and configuration. Neural networks monitor, collect, and configure the resources by using the control network in NoC. Data network is used for communicating traffic between tasks of an application. Cluster-wise neural network has been proposed because of its implementation feasibility in terms of hardware overhead for large-scale networks compared to node-wise neural network.

The contribution of this work is outlined below:

- *Neural Networks Mathematical Framework:* Neural networks mathematical model under heterogeneous NoC resources capacity, power, and thermal constraints has been proposed for task-resource co-allocation in many-core chip networks in Sec. III.

- *Neural Networks Learning Algorithm (Neuro-NoC):* Neural networks learning algorithm has been proposed for predictive configuration of NoC resources for energy-savings in many-core systems in Sec. IV.

Related work is discussed in Sec. II. Simulation results are presented in Sec. V

II. Related Work

Several works have proposed resource allocation policies while focusing on power and/or performance [22], [27], [32], [33], [43]. However, those papers have not addressed temperature induced problems. On the other hand, some works focus only on temperature-aware design [10], [25], [37], [44], where our proposed solution is both power and temperature aware. Several recent works have addressed both power and thermal challenges for NoC optimization [20], [21], [31], [36]. However, predictive techniques are needed for energy-efficient NoC at run-time. NoC architecture for neural networks has been started to be studied [18], [29], [42]. Machine learning techniques, such as regressions and neural network, are being explored for optimization in on-chip systems [4], [7], [9], [12], [13], [15], [19], [23], [24], [28], [35]. The authors of [13] propose an online learning algorithm to dynamically select voltage and frequency to minimize both energy consumption and performance delay. [9] proposes online learning algorithm to reduce the adverse effects of hot spots and temperature variations. [7] leverages backpropagation algorithm [34] to predict hybrid wireline and wireless on-chip architecture to meet the heterogeneous communication demands. The authors in [35] proposed an intelligent dynamic power management policy for NoCs to predict links on/off status. They feed the link utilization information of a sub-network to an artificial neural network (ANN), and ANN predicts links on/off status based on the pre-defined utilization threshold. In this work, we have proposed predictive configuration of both nodes and links of NoC using neural networks online learning for energy-savings while addressing both power and temperature constraints of many-core chip.

III. Neural Networks Mathematical Framework for NoC

A set of K applications can arrive at the many-core NoC system with n nodes for mapping. An application A_k can have R variable tasks, and r^{th} task of A_k is represented by T_k^r. Computation demand of T_k^r is $(cmp)_k^r$, and communication requirement between tasks r and s is $(com)_k^{rs}$. $(\alpha_k^r)^i$ be a binary variable to indicate if a task T_k^r is mapped to node i. Next, we outline the chip power, thermal, computation and communication constraints and then discuss the Neuro-NoC model and objective function.

A. Chip-Level Power Budget Constraint

The sum of power consumption of all the chip resources (nodes i + links ij) can not exceed the designed maximum power consumption budget PW_{budget} of the chip.

$$\sum_{1\leq i,j\leq n}(PW_i + PW_{ij}) \leq PW_{budget} \tag{1}$$

B. Tile-Level Thermal Budget Constraint

A tile contains core, router, and cache. Temperature of a tile is measured using ambient temperature (T_a), power consumption, surface area (SA), and thermal resistance (TR) of the nodes (core/router/cache) and corresponding links. Thermal consumption of any tile tl can not exceed the designed maximum thermal consumption threshold TH_{budget} of a chip.

$$\begin{aligned} TH_{tl} &= T_a + (PW_{tl} \cdot TR_{tl})/SA_{tl} \\ &\leq TH_{budget}, \forall 1 \leq tl \leq n. \end{aligned} \tag{2}$$

C. Computation and Communication Capacity Constraints

A task can not be mapped to a node (core) i if the computation demand $(cmp)_k^r$ of the task exceeds the maximum computation capacity P_i^{cap} of the selected node. However, more than one task can be mapped to a node if node capacity allows. Voltage of a node is configured based on the required computation at that node (core).

$$\rho_i = \sum_{\substack{1\leq k\leq K \\ 1\leq r\leq R}} (cmp)_k^r \cdot (\alpha_k^r)^i \leq P_i^{cap}, \forall 1 \leq i \leq n. \tag{3}$$

Two communicating tasks mapped to node p and q deliver the traffic through a path $Path_{pq}$, which consists of a set of NoC links. Communication bandwidth requirement of a link bw_{ij} can not exceed that maximum link bandwidth BW_{ij}.

$$bw_{ij} = (\sum_{\substack{1\leq p,q\leq n \\ 1\leq r,s\leq R \\ 1\leq k\leq K \\ \exists l_{ij}\in Path_{pq}}} (com)_k^{rs} \cdot (\alpha_k^r)^p \cdot (\alpha_k^s)^q) \leq BW_{ij}, \forall 1 \leq i,j \leq n. \tag{4}$$

D. Neural Networks Model and Objective

Distributed cluster wise neural network and a global neural network model has been proposed, as shown in Fig. 1, instead of a single centralized neural network as centralized controller can introduce hotspots in many-core chip networks [17]. Neural network is trained using NoC system-level information, such as links utilization, nodes utilization, power consumption, and thermal dissipation. Our proposed neural networks model, Neuro-NoC, is a generalized architecture for any kind of applications and network topologies (Mesh, Torus, etc.).

The objective of distributed local neural network is to minimize local energy consumption of a cluster. Let J_{local}^h denotes the energy consumption of local cluster h. Total load (energy consumption) at a node i is calculated by summing up its computation load and all the communication loads going through that node. Let x_i^h is the energy consumption at node i (including associated links) of cluster h, and θ_i^h is the weight factor of node i of cluster h. Weight factor θ_i^h of node i is updated with the changes in the utilization of NoC nodes and links depending on the tasks and applications demands.

$$min: J_{local}^h = \sum_{1\leq i\leq n^h} \theta_i^h \cdot x_i^h, 0 \leq h \leq m \tag{5}$$

where, m is the number of clusters in NoC and n^h is the number of nodes in cluster h.

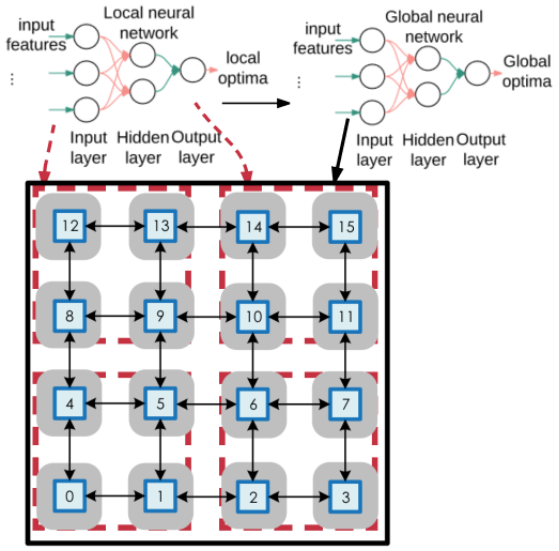


Fig. 1: Neuro-NoC Architecture for Local Cluster and Global NoC.

After training the neural networks model for the clusters, the output of each sub-neural network is the local optimal energy consumption. The trained outputs of the local clusters are fed to train the global neural network for near-optimal NoC configuration (link-width and node-voltage) for corresponding demands. Let J_{global} denotes the energy consumption of the entire NoC, and β^h denotes the weight factor of cluster h. β^h is updated to adapt to run-time changes in traffic demands. The objective of global neural network is to produce global optimal NoC configuration with minimal energy consumption.

$$min : J_{global} = \sum_{0 \leq h \leq m} \beta^h \cdot J^h_{local}, \tag{6}$$

subject to the constraints given in Eqs. (1), (2), (3), (4). The flows of online resource monitoring and configuration using neural networks is summarized in Fig. 2.

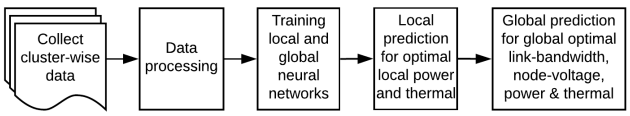


Fig. 2: Learning and Configuration Flow of Neuro-NoC Model.

IV. Neural networks Controller and Learning Algorithm

We propose to minimize the energy loss of the many-core NoC. We achieve this objective by making self-reconfigurable heterogeneous NoC architecture, which dynamically configure the NoC resources with least capacity instead of assigning maximum homogeneous capacity. The features for our Neuro-NoC model are: task computation and communication demands, link communication capacity and utilization, node computation capacity and utilization, node and link power and thermal consumption, and chip power and thermal budgets. Local and global neural networks of Neuro-NoC model are trained using these features based on node-wise and cluster-wise information, respectively. During training, Neuro-NoC model classifies the sample local and global cluster configuration as correct or incorrect based on whether the power and thermal consumption of the configuration meets power and thermal budgets thresholds of local and global NoC, respectively. At run-time (testing), Neuro-NoC model monitors the utilization of NoC nodes and links and compute the power and thermal consumption. Based on the resources utilization, Neuro-NoC model predicts the optimal NoC configuration for energy efficiency. The pseudocode for Neuro-NoC learning algorithm for NoC configuration is presented in Algorithm 1.

input : Tasks Demands, Resource Capacity and Utilization, Power and Thermal Consumption and Budgets
output : Best possible NoC configurations corresponding to the input features.
1. Preprocess the input features for normalizing the data.
2. Training the local and global neural network classifiers.
3. Feed the preprocessed inputs to the trained local classifiers for predicting the local cluster classification.
4. Predicted results from local classifiers are fed to the global classifier. Output of global classifier is the task-to-node assignment, link-width, and node-voltage for optimal NoC configuration within power and thermal budgets.

Algorithm 1: Neuro-Noc Learning Algorithm

Our neural networks model enables the optimization for the local optimum in each cluster and the global optimum in the entire NoC. For a given NoC, we divide the NoC into several clusters. The number of clusters and cluster size depend on several factors including hardware support (e.g., neural processing unit, registers, buffers) for neural networks implementation [6], [38]. For each cluster, we deploy a neural network that includes an input layer, several hidden layers, and an output layer, as shown in Fig. 1. The input features are collected from the neural networks optimizer as designed in Sec. III. We select the number of hidden layers by considering time complexity of online learning as training time increases with the number of hidden layers (besides training samples and feature size).

Our learning algorithm initially scales the input features, and then train the local and global classifiers. In the testing phase, predicted results from local classifiers are fed to global classifier for global optimal NoC configuration. The output of the global neural network is the optimization of the local optima of all clusters. Hence, we can guarantee that the global output is the best solution possible for NoC configuration to run the input application(s).

V. Simulation & Comparison

We evaluate our proposed Neuro-NoC model using its prediction capability in terms of accuracy, sensitivity, and specificity to predict global optimal configuration for NoC. We use embedded system synthesis benchmark suite (E3S) [14] to optimize 64-node NoC configuration using Neuro-NoC model.

We generate the maximum and minimum values of the NoC features to classify the boundary of feasible and infeasible configuration labels for E3S benchmarks based on following parameters: CPU capacity 0.5 GFLOPS, GPU capacity 1 GFLOPS, link frequency 1 GHz, maximum link-width 256 bit, tile area 1.6384e-5 m^2, ambient temperature 45 °C , and thermal resistance 4.5 °C per watt [2]. For energy calculation, we use computation and communication energy coefficients of 45 *picojoule (pJ)* per floating point operations (we assume 8-bit per computational operation for simplicity) and 65 *pJ* per communicating bit in *22nm* technology, respectively [5]. Heterogeneous NoC is simulated by using different computation capacity nodes (CPU + GPU) and communication capacity links. Local dataset of 12000 samples is generated using node-wise (including associated links) data for 64-node. This node-wise dataset is used to train the local classifiers. Then node-wise dataset is clustered as cluster dataset by classifying power consumption and temperature thresholds. For the cluster dataset, we generate 44800 samples of cluster-wise data for various cluster sizes for 64-node NoC: 8, 16, and 32 nodes (e.g., 4 clusters of 16-node cluster). We use this cluster-wise dataset to train the global classifier. Dataset is split into training and test datasets, where training and test datasets contain 60% and 40% of overall data, respectively.

We use scikit-learn [30] to implement the Neuro-NoC model. We have implemented multi-layer perceptron (MLP) [3] neural network, which trains using backpropagation algorithm [34], for the Neuro-NoC model with five layers, including an input layer, three hidden layers, and an output layer. MLP is used because of its online learning and configuration capability using backpropagation. Each hidden layer consists of 10 neurons. Input layer consists of the features of computation, communication, temperature, and power consumption of a node or a cluster. Output layer is the configuration label, which indicates correct (1) or incorrect (0) configuration depending on whether the configuration meets the NoC power and thermal budget thresholds for local and global NoC. Local classifier is used to optimize the local temperature and power consumption of a cluster. On the other hand, global classifier is used to optimize the global temperature and power consumption of the entire NoC. After obtaining the predicted results for testing dataset from local classifiers, we use these results in global classifier to optimize the NoC configuration label. We compare Neuro-NoC model with support vector machine (SVM) solution as SVM algorithm can produce a good classifier solution, and they (MLP and SVM) are related [8].

Fig. 3 shows the comparison between SVM and Neuro-NoC classifiers in terms of prediction accuracy. Prediction accuracy is the capability of predicting the best configuration that returns the global optimum power consumption while meeting the power and thermal budgets and computation and communication capacity of the NoC resources. Local Neuro-NoC classifier has around 1% higher prediction accuracy than local SVM classifier. As for global predictors, Neuro-NoC has around 0.5% higher classification accuracy than SVM. Thus,

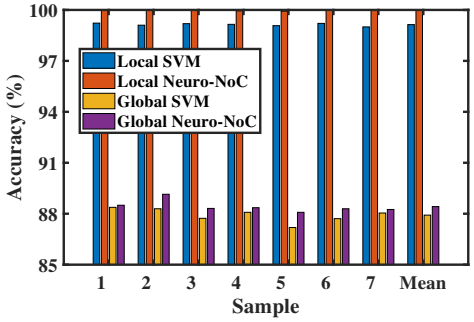
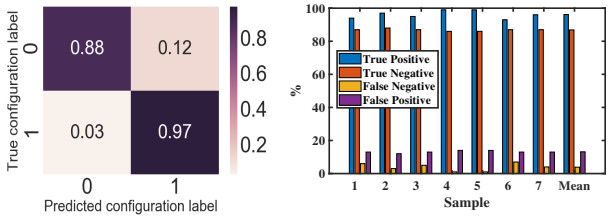


Fig. 3: Prediction Accuracy Comparison between Neuro-NoC and SVM.



(a) (b)

Fig. 4: Normalized Confusion Matrix of Global Neuro-NoC Prediction.

Neuro-NoC classifiers perform better than SVM classifiers for both local and global models.

Fig. 4 illustrates the normalized confusion matrix of the global Neuro-NoC. In the matrix in Fig. 4a, the dark-colored cells represent the number of points for which the predicted configuration label is equal to the true configuration label, and light-colored cells indicate the incorrect labeling by the global Neuro-NoC classifier. The higher diagonal values of confusion matrix indicate higher prediction accuracy. In this particular case, the true positives at position (1,1) is 97%, which means only 3% of miss prediction (specificity). It also indicates 88% correct rejection on configuration labels and only 12% false alarm [1]. To improve the quality of the predicted results of Neuro-NoC classifier, we simulated several test data samples to obtain the mean prediction value, as shown in Fig. 4b. We observe that the mean predicted configuration label of the Neuro-NoC classifier is around 96% in terms of true positives. This indicates that Neuro-NoC classifier predicts accurately the correct NoC configuration labels corresponding to the input features. Therefore, Neuro-NoC model can solve an non-convex optimization problem (as in Sec. III) by avoiding the local optima(s) to get the global near-optimal solution.

VI. Conclusion

We have proposed a neural networks model, Neuro-NoC, for online resource monitoring and configuration in many-core NoC for power-thermal efficiency in dark silicon era. This is the first work (to the best of our knowledge) for applying neural networks learning to minimize both power consumption and thermal hotspots in NoC. Neuro-NoC proactively configures the global optimal NoC with high accuracy, sensitivity, and specificity. Our Neuro-NoC enables optimized and configurable heterogeneous energy-efficient NoC at run-time for large networks and applications.

REFERENCES

[1] Confusion Matrix, http://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html.

[2] Intel Atom Processor Thermal Design Guide, https://www.intel.com/content/www/us/en/embedded/products/bay-trail/atom-e3800-m-d-i-soc-thermal-design-guide.html.

[3] Multilayer Perceptron, http://scikit-learn.org/stable/modules/neural_networks_supervised.html.

[4] Y. Bai, V. W. Lee, and E. Ipek. Voltage regulator efficiency aware power management. In *Proceedings of ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pages 825–838, April 2017.

[5] S. Borkar. Exascale computing - a fact or a fiction? In *Keynote at IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2013.

[6] Y. H. Chen, T. Krishna, J. S. Emer, and V. Sze. Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. *IEEE Journal of Solid-State Circuits*, 52(1):127–138, Jan. 2017.

[7] W. Choi et al. Hybrid network-on-chip architectures for accelerating deep learning kernels on heterogeneous manycore platforms. In *Proceedings of IEEE/ACM International Conference on Compilers, Architectures and Synthesis for Embedded Systems*, pages 13:1–13:10, Oct. 2016.

[8] R. Collobert and S. Bengio. Links between Perceptrons, MLPs and SVMs. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 23–30, July 2004.

[9] A. K. Coskun, T. S. Rosing, and K. C. Gross. Temperature management in multiprocessor SoCs using online learning. In *Proceedings of ACM/IEEE Design Automation Conference (DAC)*, pages 890–893, June 2008.

[10] A. K. Coskun, T. S. Rosing, and K. Whisnant. Temperature aware task scheduling in mpsocs. In *Proceedings of IEEE Design, Automation and Test in Europe (DATE)*, pages 1–6, April 2007.

[11] R. H. Dennard, F. H. Gaensslen, V. L. Rideout, E. Bassous, and A. R. LeBlanc. Design of ion-implanted MOSFET's with very small physical dimensions. *IEEE Journal of Solid-State Circuits*, 9(5):256–268, 1974.

[12] G. Dhiman and T. S. Rosing. Dynamic power management using machine learning. In *Proceedings of IEEE/ACM International Conference on Computer-aided Design (ICCAD)*, pages 747–754, Nov. 2006.

[13] G. Dhiman and T. S. Rosing. Dynamic voltage frequency scaling for multi-tasking systems using online learning. In *Proceedings of ACM/IEEE International Symposium on Low Power Electronics and Design (ISPLED)*, pages 207–212, Aug. 2007.

[14] R. Dick. Embedded system synthesis benchmarks suites (e3s), http://robertdick.org/tools.html.

[15] D. DiTomaso, A. Sikder, A. Kodi, and A. Louri. Machine learning enabled power-aware network-on-chip design. In *Proceedings of IEEE Design, Automation and Test in Europe (DATE)*, pages 1354–1359, March 2017.

[16] H. Esmaeilzadeh, E. Blem, R. St. Amant, K. Sankaralingam, and D. Burger. Dark silicon and the end of multicore scaling. *IEEE Micro*, 32(3):122–134, May 2012.

[17] M. A. A. Faruque, R. Krist, and J. Henkel. ADAM: Run-time agent-based distributed application mapping for on-chip communication. In *Proceedings of ACM/IEEE Design Automation Conference (DAC)*, pages 760–765, June 2008.

[18] A. Firuzan, M. Modarressi, and M. Daneshtalab. Reconfigurable communication fabric for efficient implementation of neural networks. In *International Symposium on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC)*, pages 1–8, 2015.

[19] J. Gao. Machine learning applications for data center optimization. Google Research, 2014.

[20] M. Ghasemazar, H. Goudarzi, and M. Pedram. Robust optimization of a chip multiprocessor's performance under power and thermal constraints. In *Proceedings of IEEE International Conference on Computer Design (ICCD)*, pages 108–114, Sept. 2012.

[21] J. Henkel, S. Pagani, H. Khdr, F. Kriebel, S. Rehman, and M. Shafique. Towards performance and reliability-efficient computing in the dark silicon era. In *Proceedings of IEEE Design, Automation and Test in Europe (DATE)*, pages 1–6, March 2016.

[22] J. Hu and R. Marculescu. Energy and performance aware mapping for regular NoC architectures. *IEEE Trans. TCAD*, 24(4):551–562, April 2005.

[23] D.-C. Juan, S. Garg, J. Park, and D. Marculescu. Learning the optimal operating point for many-core systems with extended range voltage/frequency scaling. In *Proceedings of IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, pages 8:1–8:10, Sept. 2013.

[24] E. Kakoulli, V. Soteriou, and T. Theocharides. Intelligent hotspot prediction for network-on-chip-based multicore systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 31(3):418–431, March 2012.

[25] H. Khdr, S. Pagani, M. Shafique, and J. Henkel. Thermal constrained resource management for mixed ILP-TLP workloads in dark silicon chips. In *Proceedings of ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, June 2015.

[26] R. Kumar, D. M. Tullsen, N. P. Jouppi, and P. Ranganathan. Heterogeneous chip multiprocessors. *IEEE Computer*, 38(11):32–38, Nov. 2005.

[27] T. T. Le, R. Ning, D. Zhao, H. Wu, and M. Bayoumi. Optimizing the heterogeneous network on-chip design in manycore architectures. In *IEEE International System-on-Chip Conference (SOCC)*, pages 184–189, Sept 2017.

[28] T. T. Le, D. Zhao, and M. Bayoumi. Efficient reconfigurable global network-on-chip designs towards heterogeneous CPU-GPU systems: An application-aware approach. In *Proceedings of IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 439–444, July 2017.

[29] E. Painkras et al. SpiNNaker: A 1-w 18-core system-on-chip for massively-parallel neural network simulation. *IEEE Journal of Solid-State Circuits*, 48(8):1943–1953, Aug. 2013.

[30] F. Pedregosa et al. scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[31] M. F. Reza, D. Zhao, and M. Bayoumi. Dark silicon-power-thermal aware runtime mapping and configuration in heterogeneous many-core NoC. In *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–4, May 2017.

[32] M. F. Reza, D. Zhao, and H. Wu. Task-resource co-allocation for hotspot minimization in heterogeneous many-core NoCs. In *Proceedings of ACM International Great Lakes Symposium on VLSI (GLSVLSI)*, pages 137–140, May 2016.

[33] M. Ruggiero, A. Guerri, D. Bertozzi, F. Poletti, and M. Milano. Communication-aware allocation and scheduling framework for stream-oriented multi-processor systems-on-chip. In *Proceedings of IEEE Design, Automation and Test in Europe (DATE)*, pages 3–8, March 2006.

[34] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Neurocomputing: Foundations of research. chapter Learning Representations by Back-propagating Errors, pages 696–699. MIT Press, 1988.

[35] A. G. Savva, T. Theocharides, and V. Soteriou. Intelligent on/off dynamic link management for on-chip networks. *JECE*, 2012:6:6–6:6, Jan. 2012.

[36] M. Shafique, S. Garg, J. Henkel, and D. Marculescu. The EDA challenges in the dark silicon era: Temperature, reliability, and variability perspectives. In *Proceedings of ACM/IEEE Design Automation Conference (DAC)*, pages 185:1–185:6, June 2014.

[37] L. Shang, L.-S. Peh, A. Kumar, and N. K. Jha. Temperature-aware on-chip networks. *IEEE Micro*, 26(1):130–139, Jan. 2006.

[38] V. Sze, Y. Chen, J. S. Emer, A. Suleiman, and Z. Zhang. Hardware for machine learning: Challenges and opportunities. *CoRR*, abs/1612.07625, 2016.

[39] M. B. Taylor. Is dark silicon useful? harnessing the four horsemen of the coming dark silicon apocalypse. In *Proceedings of ACM/IEEE Design Automation Conference (DAC)*, pages 1131–1136, June 2012.

[40] M. B. Taylor. A landscape of the new dark silicon design regime. *IEEE Micro*, 33(5):8–19, Sept. 2013.

[41] M. B. Taylor et al. The raw microprocessor: A computational fabric for software circuits and general-purpose programs. *IEEE Micro*, 22(2):25–35, March 2002.

[42] D. Vainbrand and R. Ginosar. Network-on-chip architectures for neural networks. In *Proceedings of IEEE/ACM International Symposium on Networks-on-Chip (NOCS)*, pages 135–144, May 2010.

[43] Y. Zhang, X. Hu, and D. Z. Chen. Task scheduling and voltage selection for energy minimization. In *Proceedings of ACM/IEEE Design Automation Conference (DAC)*, pages 183–188, June 2002.

[44] D. Zhu, L. Chen, T. M. Pinkston, and M. Pedram. TAPP: Temperature-aware application mapping for NoC-based many-core processors. In *Proceedings of IEEE Design, Automation and Test in Europe (DATE)*, pages 1241–1244, March 2015.