Proceedings of EnCon2011
4th Engineering Conference
Kuching, Sarawak, Malaysia

Area Optimization for Networks-on-Chip Architectures using Deep Network Partitioning

A. Lit, H. Mohamed Basri, M. N. Marsono, S. N. S. Hussin and O. C. Yee

***Abstract*— This paper presents an area optimization for Network-on-Chip (NoC) architecture using deep Network Partitioning technique. Among the hardest problems in NoC design is customizing the topological structure and application mapping on on-chip network in order to cater for application demand at minimal cost. The area cost of NoC is cut down by utilizing multi-level network partitioning where it partitions large networks into smaller segments. The enhancement in area cost is obtained by reducing both router area and the number of global links. In terms of performance, the multi-level network partitioning offers a better solution by assigning computational cores with heavy inter-core communications into the same segment. Therefore, the average inter-node distances would be minimized. This directly results in better performance due to its shortest path. For verification, the proposed technique has been tested on various System-on-Chip (SoC) applications case studies. The proposed technique results in the reduction of more than 7% router area, 19% global links, and 12% average inter-node distance.**

***Keywords*—**Graph partitioning algorithm, Network-on-chip, Network partitioning, Traffic distribution graph

I. Introduction

The complexity of system-on-chip (SoC) designs drives both industrial researchers and academics for a better solution to resolve the on-chip computational cores communication problems. For SoCs with hundreds of computational cores, on-chip communication using conventional shared buses is no longer a practical solution. In order to alleviate this problem, Networks-on-Chips (NoCs) are proposed to achieve good on-chip communication among SoC computational cores [1].

NoC optimizes the design of the on-chip network in-line with the application prerequisites. Low area, low energy consumption and low latency are some of the main performance metrics of an NoC-based system [2]. The selection of a network topology and the mapping of applications cores greatly affect NoC area and performance. Graph partitioning techniques have been proposed to improve NoC metrics [3]. In [4], a topology generation methodology to minimize the area consumption cost of NoCs was proposed. Multilevel partitioning may further improve NoC performance to fit the design necessities, while minimizing the cost.

In this paper, a multi-level network partitioning technique is used to optimize NoC topology from both area and timing viewpoints. To this end, we set out three aims.

1) Investigating the impact of multi-level network partitioning on NoC area and average inter-node distance.
2) Formulating a partitioning technique for any SoC application. Application core placement is refined as a graph partitioning problem using the Fiduccia-Mattheyses (FM) algorithm [5], and
3) Validating this approach through three multimedia SoC applications as case studies. Results prove that employing a multi-level network partitioning reduce area cost and reduces the number of hops.

The rest of the paper is organized as follows. Related work is reviewed in Section II. The impact of employing multi-level network partitioning on NoC area is investigated in Section III. Section IV provides some background on traffic distribution graph (TDG). Mathematical formulation of the network partitioning problem as addressed in Section V. Section VI corresponds to some experimental results and rationalizes the efficiency of our method by three case studies. Finally, we draw the conclusion and present ideas for further work in Section VII.

II. Related Works

Application mapping is very challenging due to the increasing number of cores and the massive data exchange. Direct mapping of applications on large SoCs require the optimization of NoC cost and performance. In reality, SoCs are a combination of multiple networks and each core only connects with a small subset of cores. Network partitioning decomposes a large system to a smaller subsystems to optimize the metrics of interest.

Standard NoC topologies such as ring, octagon, mesh, and torus were compared and evaluated with respect to throughput, average latency, energy consumption and area in [6], [7]. Various mapping algorithms, like PBB [8], GMAP [8], BMAP [9], and NMAP [10] were proposed to map out any application onto those standard topologies. In [11], a tool called SUNMAP was introduced to automatically choose the best standard topology for any given application.

Graph partitioning algorithms divide an enormous graph into smaller networks [3]. The most well-known graph partitioning algorithms is the Fiduccia-Mattheyses (FM) which is the derivative from the Kerninghan-Lin (KL) algorithm [12]. Some other methods have also been used to execute the

Asrani Lit and Hazrul Mohamed Basri are with the Faculty of Engineering, Universiti Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia (e-mail: lasrani@feng.unimas.my). Other authors are with the Faculty of Electrical Engineering, Universiti Teknologi Malaysia , 81310 Skudai, Johor, Malaysia

partitioning duty such as scattered, spectral, linear and random techniques [13]. OIDIPUS is a tool that applies graph partitioning approach for NoC topology generation proposed in [14]. But its aims are restricted to reduce power consumption and response time. This tool required the size of each division and the number of divisions. Furthermore, every division was only fixed to the ring topology. Generated topologies from the tool were not assessed against any standard or custom topologies.

In [14] custom topology generation for NoC was proposed by manipulating a graph partitioning algorithm. Multilevel partitioning before core placement allows better partitioning as the critical path are in the same partition and easier for island placement in network topology especially for complex SoC. With critical paths in the same partition, communication latency and energy can be reduced [3].

Current network partitioning for application mapping only looks for top down partitioning. In most cases, multilevel partitioning continues partitioning until objectives are reached such as minimum number of cuts or the minimum average cut weights. Thus, in order to improve the NoC performance, deep network partitioning is needed. Partitioning NoCs may reduce the required buffering and global links for partition boundary nodes [3].

III. Impact of Multi-level Network Partitioning in NoC

The total NoC area is primarily determined by routers area [9]. Furthermore, the main region of any router area is established by the port with their buffers [15]. Thus, the less number of ports for the routers in the topology, the lower the NoC area. Table I shows the area of input-queue routers with various number of port [2]. Every port has identical buffer size and comprises of two channels for packets reception and transmission, respectively.

TABLE I
Number of port and Area [2]

Area	Number of Ports				
	1	2	3	4	5
μm^2	33 600	50 200	66 800	83 400	100 000

The multi-level of network partitioning divides the partitioned network into finer partitions. Subsequently, a custom topology is generated by linking the partitions together. The finer partitioning results in fewer number of ports compared to a single undivided network. Hence, multi-level network partitioning can be used to reduce the overall NoC router area.

This technique does not only decrease the router area, but the number of global links as well. Global links bridge up non-neighboring routers, whereas local links connect routers with computational cores. Normally, global links are long (relative to local links) and require repeaters and driving circuits. Hence, it is important to reduce the number of global links in the NoC topology. Deep network partitioning approach could be used to lower the two key factors in NoC area; the number of global links and router area. Therefore, this multi-level network partitioning approach is proposed to accommodate the NoC area reduction.

TABLE II
Number of router and total routers area before and after 2^{nd}-level of network partitioning

Case	5-port	4-port	3-port	Routers area (μm^2)
Unpartition	4	8	4	1322000
1^{st}-level	2	6	8	1234000
2^{nd}-level	2	2	12	1168400

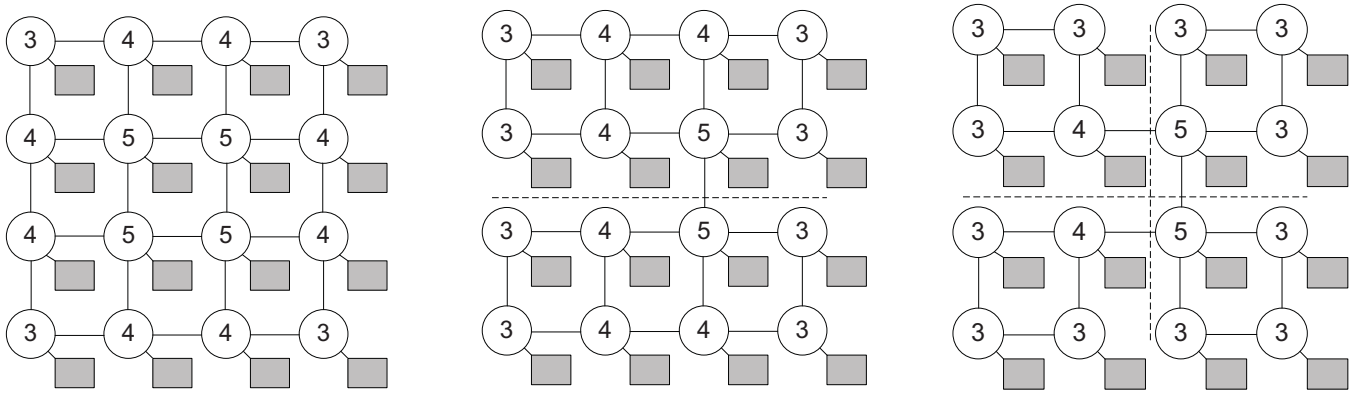
Fig. 1 shows a 4×4 mesh implementation of a 16 node application before and up to 2^{nd}-level of network partitioning. The partitioning scheme is discussed in depth in Section IV. The 1^{st}-level network partitioning removes three global links. The 2^{nd}-level network partitioning removes two additional global links. Table II summarizes the total number of routers required with different number of ports prior to partitioning, after the first partitioning, and after the second level of partitioning on the total NoC router area. In short, adopting 2^{nd}-level network partitioning decreases the NoC router area by 12.28% for this example.

As the number of nodes increases, the number of links removed by network partitioning increases as well. Besides, removing a link results in lower number of port per router. In Table I, the number of ports eliminated is a linearly proportional with the router area reduction. The more computational cores in an SoC, the more global links can be removed by partitioning, and more routers area is saved. Eventually, partitioning become more practical in reducing the area cost with the increasing number of computational cores.

The FM algorithm [5] is used for the graph partitioning technique. The selection of FM is two-fold. First, it reduces the impact of partitioning on the average delay by minimizing the inter-partition traffic (cut-edge). Second, it prevents inducing communication bottleneck on the NoC based system. Hence, buffering requirement on inter-partition routers (located at the boundary of partitions) can be reduced.

IV. Graph Theoretic Approach

Graph theoretic concept can be used to represent the topological structure of any interconnection network. The traffic criteria and application description are assumed to be statistically analyzed and identified. Therefore, any SoC application could be described by a traffic distribution graph (TDG) [10], [11]. A TDG describes the computational cores in the application as well as the communication traffic among them. A TDG incorporates of directed graph $G(C, W)$, where each vertex $c_i \in C$ represents a computational core in the SoC application, and the directed edge $w_{ij} \in W$ is the communication between computational cores c_i and c_j (in terms of the number of packets per time step) as shown in Fig. 2.



(a) Unpartition Network (b) 1^{st}-level of Network Partitioning (c) 2^{nd}-level of Network Partitioning

Fig. 1. 4×4 mesh implementation before and after 2^{nd} level of partitioning

In NoC terminology, traffic distribution matrix (λ) is another form of TDG [2]. For n computational cores in a particular SoC application, the dimension of the matrix is $n \times n$. The weight of the edge w_{ij} is represented by λ_{ij}, which is the number of packets transmitted by computational core c_i to computational core c_j per time step. For instance, Fig. 2(a) depicts the representation of MPEG-4 application in the traffic distribution matrix form [2].

$$\lambda = \begin{bmatrix} 0 & 0 & 0 & 0 & 190 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 60 & 40 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 600 & 40 & 0 & 0 & 0 & 0 & 0 & 0 \\ 190 & 0.5 & 60 & 600 & 0 & 0 & 0 & 0 & 0.5 & 910 & 32 & 0 \\ 0 & 0 & 40 & 40 & 190 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 250 & 0 & 670 & 173 & 500 \\ 0 & 0 & 0 & 0 & 0 & 0 & 250 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 910 & 0 & 670 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 32 & 0 & 173 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 500 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

V. Problem Formulation

This section introduces the mathematical formulation of the partitioning strategy practiced in this paper. Network partitioning is NP-hard [17]. The FM algorithm is chosen because of its outstanding performance over other network partitioning methods [17].

To formulate the partitioning problem mathematically, n application cores are divided into m partitions $P_1, \ldots, P_m$. The number of cores within any partition P_k is n_k. Consequently, the total traffic (λ_{total}), in packets/time step. The communication among all application cores is defined as [3]:

$$\lambda_{total} = \sum_{i=1}^{n} \lambda_{ij} \tag{1}$$

For any partition P_k, the overall traffic within the partition (λ_k), in packets/time step, is given by [3]:

$$\lambda_k = \sum_{i=1}^{n} \lambda_{ij} \ \forall \ i,j \quad \text{where } c_i, c_j \in P_k \tag{2}$$

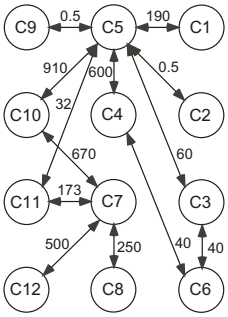
Likewise, for any partition P_k, the overall traffic sent from its nodes to all nodes in another partitions $(\overline{\lambda_k})$, in packets/time step, is shown as [3]:

$$\overline{\lambda_k} = \sum_{i=1}^{n} \lambda_{ij} \ \forall \ i,j \quad \text{where } c_i \in P_k, c_j \notin P_k \tag{3}$$

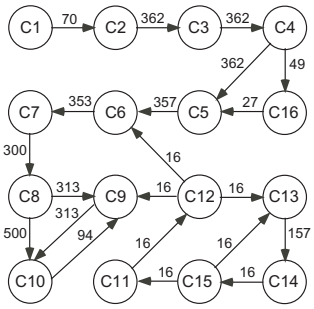
The overall traffic (λ_{total}) is based on the application itself. The intra-partition traffic (λ_k) and the inter-partition traffic $(\overline{\lambda_k})$ are used by the FM partitioning scheme. As previously mentioned, the network partitioning is aimed at minimizing the total traffic among all partitions. Hence, the overall inter-partition traffic (λ_{inter}), in packets/time step, is given by [3]:

$$\lambda_{inter} = \sum_{k=1}^{m} \overline{\lambda_k} \tag{4}$$

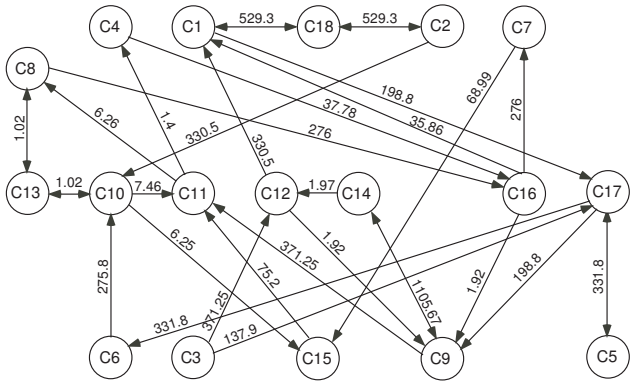
Every partition generates a set of cut edges, E_c specified as the subset of E. The weight of each subset, $|S_i|$ is formed to be the number of vertices mapped to that subset by P. With a graph as the input, the graph partitioning problem attempts to discover a p-way partitioning in which each subset consists approximately the equivalent number of vertices $\left(|S_i| \leq \frac{|V|}{p}\right)$ and the number of cut edges, $|E_c|$, is minimized [3]. The cut edges constitute the interprocessor communication needed by the distribution. Thus, the graph partitioning problem aims at a distribution that balances the computation handled by each processor while minimizing the entire interprocessor communication [17].



(a) MPEG [4]



(b) VOPD [9], [10]



(c) VA [16]

Fig. 2. Traffic Distribution Graph

VI. Experimental Results

The multi-level network partitioning is applied to three SoC multimedia applications with various number of computational cores as the proof-of-concept: Video Audio (VA) benchmark with 18 computational cores as shown in Fig. 2(c) [16], Video Object Plane Decoder (VOPD) with 16 computational cores, as shown in Fig. 2(b) [9], [11] and MPEG-4 decoder with 12 computational cores, as shown in Fig. 2(a) [9], [11]. The level of network partitioning is chosen up to 2^{nd}-level since the number of computational cores is relatively small. Matlab is utilized to accomplish the partitioning. Meanwhile, for the purpose of computational core placement, heuristic mapping is used. Mesh topology is applied for all partitions and the rationality behind mesh selection is two-fold. First, the mesh topology has an upper hand compared to other NoC topologies in regards of power consumption and area [7]. Second, almost all of practical NoC are defined in grid-based topologies, such as torus and mesh [18]. Furthermore, grid based topology could be naturally implemented on chips.

The total routers area and the number of global links obtained from up to the 2^{nd}-level of network partitioning using the FM algorithm are compared with standard mesh topology

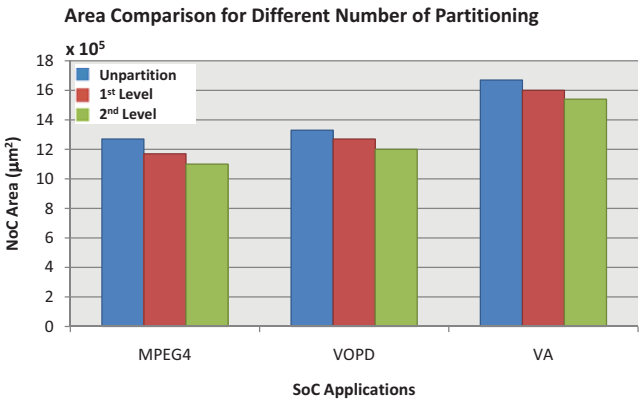


Fig. 3. Routers area for different level of partitioning

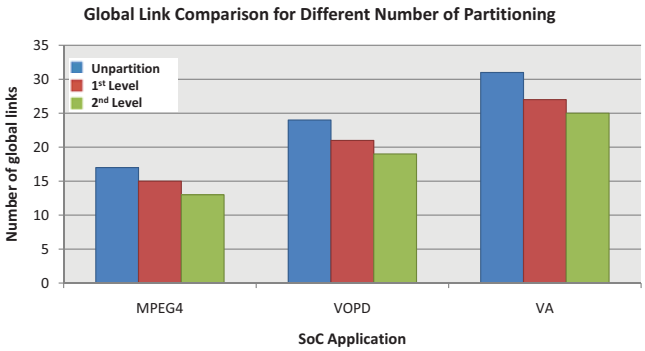


Fig. 4. Number of global links for different level of partitioning

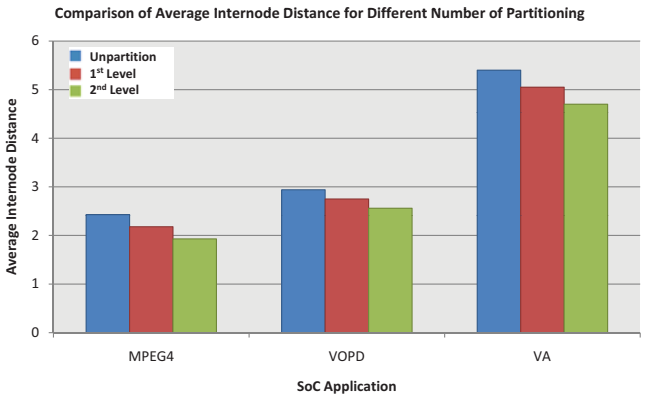


Fig. 5. Average internode distance for different level of partitioning

without partitioning and with 1^{st}-level of network partitioning. The results are shown in Fig. 3 and 4, respectively. By employing the FM algorithm, NoC router area is reduced by 13.39%, 9.77% and 7.78% for MPEG-4, VOPD, and VA applications, respectively. Likewise, the number of global links are removed by 25.53%, 20.83% and 19.35%. From these results, it is visible that multi-level network partitioning is a persuasive technique to cut down NoC area specially for applications with vast quantity of computational cores. To find

out the impact of expanding FM algorithm on the latency, the average inter-node distances for three applications are counted and shown in Fig. 5. The average inter-node distance are improved by 20.58%, 12.93% and 12.96% for MPEG4, VOPD and VA applications, respectively. These results clearly show that deep multi-level network partitioning is capable of reducing router areas, the number of global links, and the inter-node distances.

VII. Conclusion and Future Work

NoC is a competent paradigm to resolve the NoC communication problem in state-of-the-art SoC applications. In this paper, we examined how the NoC can be reduced by employing multi-level network partitioning techniques. The finer partitioning allows for a lower number of buffers and ports. The multi-level network partitioning is an efficient approach to decrease NoC area particularly for application with a large number of computational cores and this has been verified by experimental results. We plan to broaden this work in these respective directions. First of all, we will try to look into the optimal level of network partitions, the optimal amount of computational cores per partition, and the optimal network topology for each partition. Secondly, we are going to combine partitioning with 3D NoCs to improve the NoC performance. Third, we aim at analyzing the timing performance with numerous metrics such as throughput, average queue size, waiting time, and packet loss.

VIII. Acknowledgement

The authors would like to express their sincere gratitude to Research and Innovation Management Centre (RIMC), Universiti Malaysia Sarawak (UNIMAS) for the financial support.

References

[1] W. Benini, "Application specific NoC design" in Proceeding of the IEEE Design, Automation, and Test in Europe Conference (DATE'06), vol. 1, Munich, Germany, Mar. 6-10, 2006, pp. 1–5.

[2] H. Elmiligi, A. A. Morgan, W. W. El-Kharashi and F. Gebali, "A topology-based design methodology for Network-on-Chip applications". In Proceedings of the second International Design and Test Workshop (IDT'07), Cairo, Egypt, Dec 16-18, 2007, pp. 61–65.

[3] A. Morgan, H. Elmiligi, W. El-Kharashi and F. Gebali, "Application Specific Network-on-Chip Topology Customization Using Network Partitioning", in Proceedings of the First International Forum on Next-Generation Multicore/Manycore Technologies (IFMT'08), November 2008, Cairo, Egypt, pp.13:1–13:6.

[4] H. Elmiligi, A. A. Morgan, M. W. El-Kharashi and F.Gebali, "Power-aware topology optimization for network-on-chips." In Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'08), Seattle, WA, USA, May 18-21, 2008, pp. 360–363.

[5] C. Fiduccia and R. Mattheyses, "A linear-time heuristic for improving network partitions." In Proceedings of the 19th Annual IEEE/ACM Design Automation Conference (DAC'82), Las Vegas, NV, USA, June 14-18, 1982, pp. 175–181.

[6] L. Bononi and N. Concer, "Simulation and analysis of Network-on-Chip architectures : Ring, spidergon and $2D$ mesh" in Proceeding of the IEEE Design, Automation and Test in Europe Conference (DATE'06), vol. 2, Munich, Germany, Mar. 6-10, 2006, pp. 154–159.

[7] P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh. "Performance evaluation and design trade-offs for Network-on-Chip interconnect architecture", IEEE Transactions on Computers, vol. 54, no. 8, pp. 1025–1040, August, 2005.

[8] J. Hu and R. Marculescu, "Energy-aware mapping for tile-based NoC architecture under performance constraints", in Proceeding of the IEEE Asia and South Pacific Design Automation Conference (ASP-DAC'03), Kitakyushu, Japan, Jan. 21-24, 2003, pp. 233–239.

[9] W. Shen, C. Chao, Y. Lien and A. Wu, "A new binomial mapping and optimization algorithm for reduced-complexity mesh-based on-chip network." In Proceeding of the IEEE International Symposium on Network-on-Chip (NOCS'07), Princeton, NJ, USA, May 7-9, 2007, pp. 317–322.

[10] S. Murali and G.D. Micheli, "Bandwidth-constrained mapping of cores onto NoC architectures", in Proceeding of the IEEE Design, Automation and Test in Europe Conference and Exhibition (DATE'04), vol. 2, Paris, France, Feb. 16-20, 2004, pp. 896–901.

[11] S. Murali and G.D. Micheli, "SUNMAP: A tool for automatic topology selection and generation for NoCs", in Proceedings of 41st IEEE/ACM Design Automation Conference (DAC'04), San Diego, CA, USA, June 7-11, 2004, pp. 914–919.

[12] B. Kerninghan and S. Lin, "An efficient heuristic procedure for partitioning graphs", Bell System Technical Journal, vol. 49, no. 2, pp. 291-307, February 1970.

[13] P. Chan, M. Schlag and J. Zien, "Spectral k-way ratio-cut partitioning and clustering", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 13, no. 9, pp. 1088–1095, September 1994.

[14] T. Ahonen, D. Siguenza-Tortosa, H. Bin and J. Nurmi, "Topology optimizatio for application-specific Networks-on-Chip", in Proceedings of the 2005 ACM international workshop on System level interconnect prediction (SLIP'04), Paris, France, Feb. 14-15, 2004, pp. 53–60.

[15] M. Coenen, S. Murali, A. Radulescu, K. Goosens and G. D. Micheli, "A buffer-sizing algorithm for networks on chip using TDMA and credit-based end-to-end flow control", in Proceeding of the Third IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS'06), Seoul, Korea, Oct. 22-25, 2006, pp. 130–135.

[16] V. Dumitriu and G. Khan, Throughput-oriented NoC topology generation and analysis for high performance SoCs, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 17, no. 10, pp. 1433–1446, October 2009.

[17] B. Chamberlain, "Graph partitioning algorithms for distributing workloads of parallel computations", Technical Report UW-CSE-98-10-03, University of Washington, WA, USA, October 1998.

[18] G. Karypis, K. Schloegel and V. Kumar "PARMETIS: Parallel graph partitioning and sparse matrix ordering library, Version 3.1", University of Minnesota, Minneapolis, USA, Tech. Rep. 2003.