IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 3, 2020, accepted March 16, 2020, date of publication March 23, 2020, date of current version April 15, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2982675

Performance Evaluation of Application Mapping Approaches for Network-on-Chip Designs

WAQAR AMIN[1], FAWAD HUSSAIN[1], SHERAZ ANJUM[2], SARZAMIN KHAN[2], NAVEED KHAN BALOCH[1], ZULQAR NAIN[3], (Student Member, IEEE), AND SUNG WON KIM[3]

[1]Department of Computer Engineering, University of Engineering and Technology, Taxila 47050, Pakistan
[2]Department of Computer Science, COMSATS University Islamabad, Wah Campus, Wah Cantt 47040, Pakistan
[3]Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 712-749, South Korea

Corresponding author: Sung Won Kim (swon@yu.ac.kr)

This work was supported in part by the Brain Korea 21 Plus Program funded by the National Research Foundation of Korea (NRF) under Grant 22A20130012814, in part by the Ministry of Science and ICT (MSIT) South Korea, under the Information Technology Research Center (ITRC) support program, supervised by the Institute for Information and communications Technology Planning and Evaluation (ITRC), under Grant IITP-2019-2016-0-00313, and in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant 2018R1D1A1A09082266.

ABSTRACT Network-on-chip (NoC) is evolving as a better substitute for incorporating a large number of cores on a single system-on-chip (SoC). The dependency on multi-core systems to accomplish the high-performance constraints of composite embedded applications is on the rise. This leads to the realization of efficient mapping approaches for such complex applications. The significance of efficient application mapping approaches has increased ever since the embedded applications have become more complex and performance-oriented. This paper presents the detailed comparative analysis and categorization of application mapping approaches with current trends in NoC design implementation. These approaches target to improve the performance of the whole system by optimizing communication cost, energy, power consumption, and latency. Apart from the categorization of the discussed approaches, comparison of communication cost, power, energy, and latency of the NoC system carried out on real applications like VOPD and MPEG4. Moreover, the best technique identified in each category based on the evaluation of performance results.

INDEX TERMS Network-on-Chip, application mapping, NoC design, VOPD, System-on-Chip.

I. INTRODUCTION

System-on-Chip (SoC) is an archetype for the design and implementation of on-chip circuits that can support multiple systems on a single chip. The ever-rising number of processing cores on a single chip has made the efficiency of on-chip designs as one of the major aspects in evaluating the average performance of embedded SoC. To fulfill the performance needs and to provide flexibility in the designs the field of Network on Chip (NoC) has emerged that separates the communication from the computation. Many surveys [1]–[7] are published in general and textbooks are also available on the topic [8]–[10]. There is still a need to solve more advanced research problems in NoC. Application mapping on NoC architecture has a prominent place among all the research problems which can be arbitrated from the published papers and current trends. This survey assumes a textbook-level acquaintance of NoC terminology. The aim is to provide a comprehensive overview of all the aspects of application mapping (task graph generation, scheduling, optimization techniques, simulation setup, and performance evaluation metrics). The organization of this survey is as follows; first, we provide the reader the need and overview of application mapping in NoC presented in Section I. A detailed review and calcification of application mapping techniques in NoCs are discussed in Section II. The current and latest trends in application mapping are summarized in Section III. In Section IV performance comparison and the points that can create the difference between these techniques are highlighted. Finally, Section VI concludes this paper.

The associate editor coordinating the review of this manuscript and approving it for publication was Eyuphan Bulut.

A. MOTIVATION AND SCOPE

Mostly synthetic traffic patterns are used to imitate the functionalities of real cores. With the help of results of latency,

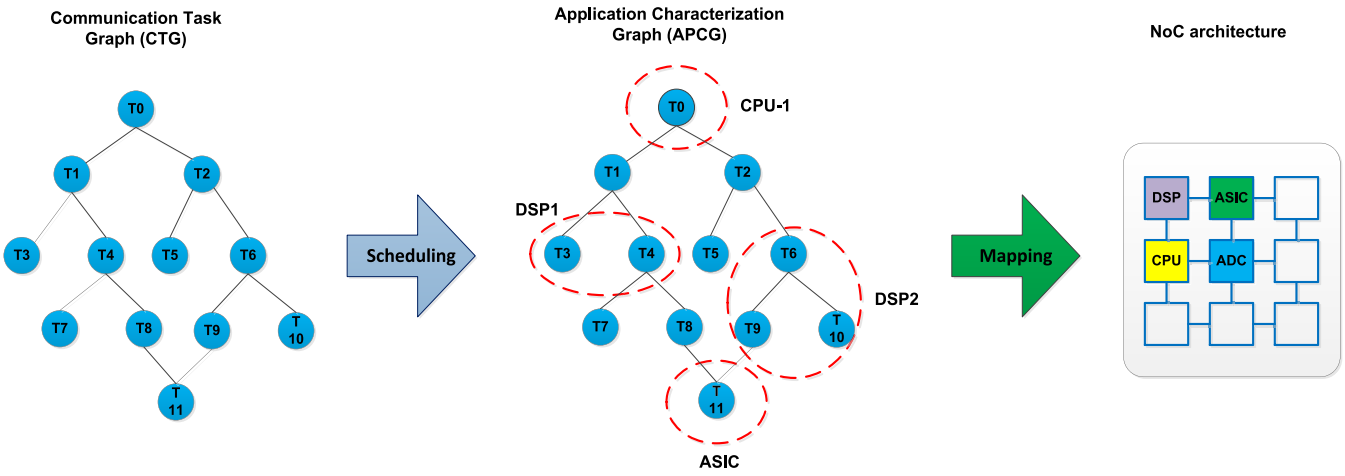


FIGURE 1. The scheduling, mapping and routing problems application.

throughput, and communication bandwidth derived from the simulated network, the designer can accurately estimate area and power consumption for NoC. Once the designs of communication infrastructure, communication methodology and evaluation framework for NoC have been addressed, a final and important phase is to associate and finds the best arrangement and placement of application tasks on NoC cores. This is very useful in improving the communication cost and overall performance of NoC and forms the fourth dimension namely application mapping in NoC [1], [8]–[10]. In this study, we have focused mainly on this research area to evaluate the application mapping approaches in NoC designs.

B. OVERVIEW OF APPLICATION MAPPING IN NOC

Application mapping has become a vital aspect in NoC architecture design. It performs in allocating application-related tasks to specific cores in the first instance an efficient mapping technique is then employed to calculate the optimized arrangement of these cores on the nodes of the NoC. The application mapping techniques must be designed keeping in mind the constraints of many-core on-chip systems, such as bandwidth, communication time, latency, throughput, power, and energy. Heuristic search approaches are frequently used to compute an optimal solution of mapping as it is acknowledged to be an NP-hard problem [1]–[7]. A general description for NoC architectures and applications and several outstanding research problems provided in [1], [2]. An interesting and comprehensive survey presented in [3] categorizes mapping techniques into two important domains; dynamic and static mapping.

This survey further classifies and elaborates these domains into sub-domains e.g. Exact mapping, Search-based mapping, and hybrid mapping techniques and discusses application algorithms presented under each sub-domain. In the end, the author computes the communication costs of algorithms and evaluate them for different benchmarks. Similar work is also presented in [5], [7]. In [4], authors have reported mapping and scheduling strategies for NoC and bus-based systems, their main objective was to discover the common issues between NoC and the bus-based systems. Fault-tolerant application mapping techniques are discussed and categorized based on the methodologies implemented to recover from failures in another informative survey [6]. The combination of routing and mapping algorithms, techniques based on task remapping and techniques based on redundancy are discussed in detail.

The first three research dimensions of NoC have been reviewed in detail and we can find several surveys on the first three research dimensions as discussed earlier. However, if we compare the fourth research dimension namely application mapping in NoC, the extent of surveys is quite less. While there are some good surveys in this research area, which provide a comprehensive, organized and detailed overview of mapping techniques in NoC. Still there exists ample space for a comprehensive study on the evaluation of application mapping techniques in NoC. This study mainly focuses on these recent works in the field of application mapping.

The primary objective is to organize the application mapping algorithms into different categories based on their salient features and functionalities, evaluate and compare them based on performance parameters and finally highlight some current research trends for NoC. The application mapping in NoC is generally classified into three main steps. In the first steps, the application is divided into a graph of concurrent tasks (threads). In the second step, assigning and scheduling of tasks to available cores is performed. While in the third step, cores are mapped to NoC tiles. **Fig. 1** explains the scheduling, application mapping problems.

C. TASK GRAPH GENERATION

An application running on NoC architecture is represented through a task graph. A task is defined in [11] as a set of instructions that are executed sequentially, on the same processor, without preemption. The Communication Task Graph

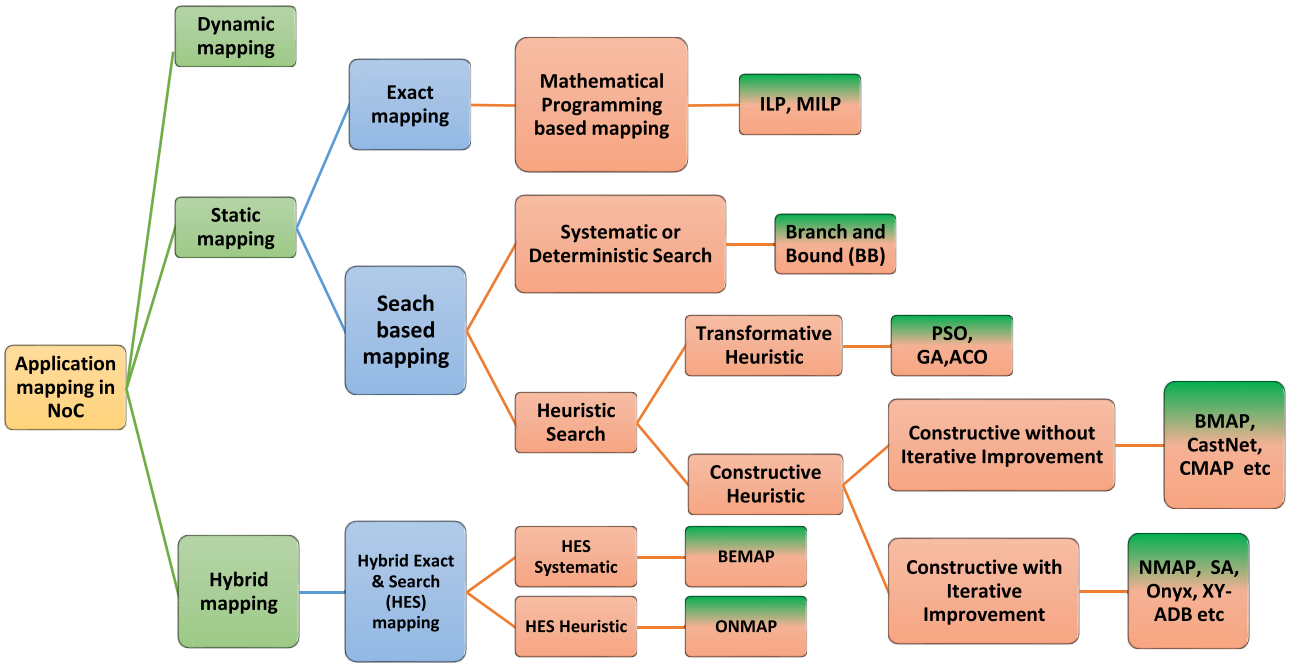


FIGURE 2. Classification of application mapping techniques in NoC.

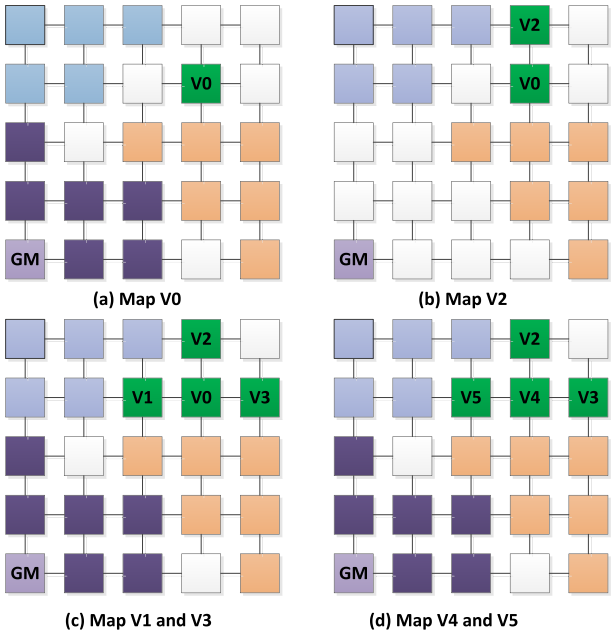


FIGURE 3. Hierarchical task mapping [22].

(CTG) presents the communication pattern of the application and divide the application into tasks (concurrent threads) [1]. It shows inter-task communication and the volume of the data exchange between tasks. The Directed Acyclic Graph (DAG) [11] is used to address the scheduling problems. The task is represented as a node in the DAG to which the weight is attached which denotes the computational cost. The acyclic property of a Communication Task Graph is denoted by Application Characterization Graph (APCG). Similar to a CTG, a directed arc of an APCG indicates data and control dependencies. But compared to a CTG, an APCG allows cycles. CTGs can be obtained in the following ways:

1. By using the TGFF tool [12]
2. Using the E3S benchmarks suite;
3. From real-world multithreaded applications, using the CETA [13] tool.
4. Benchmark scaling algorithm Gen(B(t)) [14] for generating large-scale task graphs for dynamic applications.

The main focus of this survey is only on the optimization techniques for the on-chip application mapping. Interested readers can review the literature on the task graphs generation techniques.

D. SCHEDULING

Time spent on computation and communication to run an application is an important factor in determining the performance of that application. While the computation time is mainly depending on the design of the IP cores, the communication time is determined; not only by the routing protocol but also by the method in which the tasks are scheduled . . . The execution order of tasks must be established to achieve optimized performance. This is called the scheduling problem. This can be done using a scheduling algorithm.

For example, the EAS algorithm [15] can perform scheduling in real-time restrictions, while trying to optimize the energy consumption of the NoC architecture. A heuristic scheduling approach based on slack-budgeting proposed in [16]. The approach used is to assign more slack to applications that have high energy consumption and impact on

performance. The tasks are prioritized based on the weighted cost which is proportional to the execution time and energy consumption. To achieve better performance, the algorithm iteratively schedules and swaps more critical tasks with non-critical tasks mapping on the same core. In [17] a scheduling algorithm proposed for CDMA-based NoC where the combination of the Dual Round-Robin Matching (DRRM) algorithm with the Orthogonal Variable Spreading Factor (OVSF) codes are used to schedule tasks. The work in [18] presented the scheduling of multiple communication patterns and proposed multiple heuristics such as a backtracking approach, a greedy algorithm, and an oracle approach to solving the scheduling problem while reducing the resource utilization. In [19], the authors proposed a user-managed migration strategy based on user-level middleware support and code checkpoint for reducing the execution time of a task migration event. A greedy heuristic scheduling algorithm for real-world applications proposed in [20]. In this approach, the timing of packet release is determined to minimize contention at the destination. Regarding the congestion in the NoC architectures, a Congestion-

Controlled Best-Effort (CCBE) scheduling strategy which monitors the link utilization is proposed in [21]. A hybrid task scheduling algorithm based on task clustering (HTSTC) is proposed in [22]. The algorithm used a task clustering approach to integrate tasks and used task duplication techniques for processor selection. In [23], a task scale based scheduling and mapping approach are proposed. In this approach, tasks are first divided according to the task size. When the task scale is small, a given NoC substructure is selected to map the task, so that the tasks can be centralized; when the task scale is large, the tasks are arranged in clusters and divide certain tasks with dependencies to the same resource node.

II. CLASSIFICATION OF APPLICATION MAPPING TECHNIQUES

Mapping techniques in NoC can be grouped as dynamic (run-time) and static (offline). The dynamic mapping is also known as run-time or real-time and online mapping, where the assembling and assignment of application tasks to cores of NoC are executed in real-time. Dynamic mapping is an efficient technique because it performs mapping based on the runtime load of cores. It also distributes workload among the cores by analyzing the traffic load and identifying the performance bottleneck at any core. However, the computational complexity of the mapping algorithm in real-time increases energy consumption and introduces a run-time execution delay. Static mapping performs mapping of application tasks in off-line mode at design time. For example, the mapping is finalized before the execution of the application, static mapping algorithms employed only once, consequently result in better energy consumption and delay as compared to dynamic mapping [3]–[5]. **Fig. 2** shows the classification of application mapping techniques in NoC. Now we will explain all the three branches and algorithms mentioned in this figure.

A. DYNAMIC (RUN-TIME) MAPPING TECHNIQUES IN NOC

In dynamic mapping, the tasks are mapped and allocated to the cores by analyzing the communication bandwidth and a load of each core at run-time. The assignment of each task can also be altered in runtime. Time consumed to allocate tasks is critical since it impacts on the overall execution time of the application. To reduce the mapping overhead, it is significant to employ efficient mapping strategies. In [24], authors have presented and evaluated experimentally a compiler-based application mapping algorithm to reduce energy exhaustion and improved performance. Energy results are presented with or without packet routing and task mapping to cores. In [25], Path Load (PL) a dynamic congestion aware task mapping approach is reported, which gives a solution to shortcomings in previous techniques. Instead of using all links for mapping, Path Load (PL) only utilize links that are essential for task mapping. Path Load (PL) technique reduces the mapping time and gives the best solution._Similar work is presented in [26], the Best neighbor (BN) mapping algorithm which reduces the execution time. The authors also did the performance evaluation with benchmarks e.g. VOPD, MPEG4, MWD, and RBERG. In [27], a run-time solution for task mapping problem in homogeneous NoC architecture is proposed, where user behavior is incorporated during the task assignment procedure. This enables the system to respond more efficiently to changes and adjust accordingly during run time. A two-step dynamic approach to incrementally map various applications on a homogenous NoC architecture with multiple voltage levels has been addressed in [28]. In the first step, an appropriate region of NoC is selected for mapping and then a greedy run time heuristic employs for task mapping. Application execution time and communication energy were estimated using real application benchmarks. Occasionally, run-time task mapping becomes essential because the running tasks in an MPSoC can surpass available resources. In [29] a run-time approach has been presented to assign tasks in run-time, according to the requirement of communication bandwidth and load over links. The techniques minimize the congestion, packet latency, and channel load by implying different heuristics techniques such as Path Load (PL), Best Neighbor (BN), and Nearest Neighbor (NN). The technique proposed in [30] used an agent-based mapping approach for application mapping in a distributed manner. In comparison to centralized approaches, it reduces the computational effort and traffic monitoring for mapping procedures. In this agent-based mapping approach, agents are described as small tasks that manage resources and store information related to the state of resources. They can be implemented on any node in NoC. The agents communicate with each other to discover processing components appropriate for mapping. Global Agents (Gas) have the global knowledge of all clusters while Cluster Agents (CAs) received new mapping requests and share these requests with GAs.

In [31], authors have presented a spiral mapping technique Dynamic Spiral Mapping (DSM) for run-time task mapping. The techniques searched a suitable placement to map a task

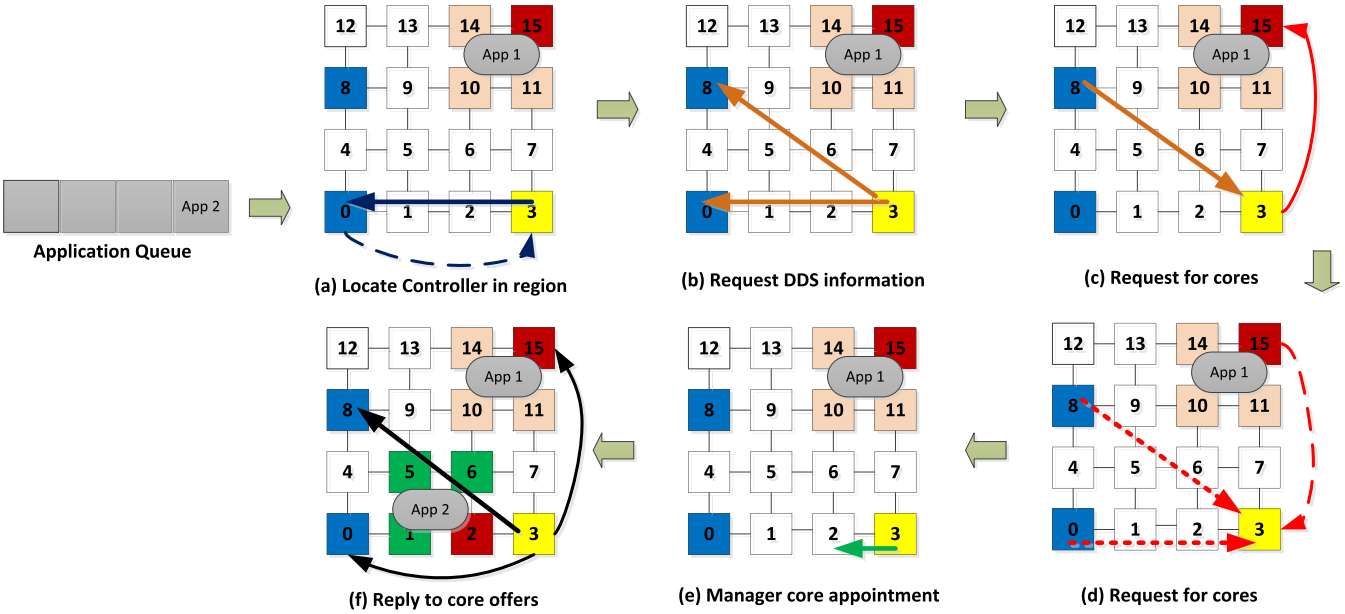


FIGURE 4. DistRM mapping technique [25].

in a spiral path beginning from the center to the edges of the network. To reduce communication time, runtime mapping/allocation time and task relocation time, it places the task closer to each other. A combination of software and hardware processing elements based on MPSoC architecture has been presented in [32]. In this architecture, one processing node among all nodes acts as a manager node. The manager node keeps states of resource control, reconfiguration control, task migration, and task binding, and assigns new mapping based on the information. A two-phase mapping algorithm also offered. In the first phase, preliminary mapping is performed based on the first vacant place that appeared in the network. In the second level, the new tasks requesting for placement are mapped by engaging a run-time algorithm. Similar and more detailed work has been proposed in [33] with new mapping approaches. Authors have proposed a Contiguous Neighborhood Allocation (CoNA) [34] mapping solution for NoC. The algorithm is a combination of three novel approaches. The first approach uses an effective procedure for finding the first node with a maximum no of free neighboring nodes available for mapping. The second approach picks the first task with the largest degree to be mapped onto the selected available node. Subsequently, creating an adjacent area of vacant nodes in the proximity of the first node to assign the remaining tasks of the application is the third role of the proposed algorithm. Extension of this work has been presented in [35], where the Smart Hill Climbing SHiC technique used for optimum first node selection for application task mapping. A Hierarchical And Dependency-aware (HAD) task mapping technique has been proposed in [36]. One core in the network, bottom left in this case, is dedicated as Global Manager (GM). GM executes the HAD algorithm where initially, the number of task transitions are compared with the available cores and in second step placement of cores are decided by calculating mean occupied position (MP). **[22, Fig. 3]**, shows the hierarchical task mapping approach. In [37], a runtime distributed task mapping approach DRTRM has been proposed. DRTRM is based on the scheme of local managers and controllers. This technique is classified into three categories named initial core, manager core and controller core. Initial core selects cores for mapping while the manager core manages resources and the controller core monitors activity of different regions of the platform. Another first node selection based technique namely proactive region selection strategy (MapPro) presented in [38]. The technique exploits the idle time between two consecutive application mapping requests to find appropriate candidate regions for mapping. The idea of the ripple effect defines adjacent neighboring nodes for mapping. MapPro technique offers a decrease in execution time and low congestion.

In [39], authors have proposed a run-time multi-agent-based distributed resource management approach DistRM for NoC. **[25, Fig. 4]**, explains the DistRM approach. Agents are distributed over the complete network instead of a centralized placement and manage tasks mapping to available cores. A run-time approach described the mapping of both run time applications and offline applications on the same network proposed in [40]. Critical run-time applications are mapped nearby, while noncritical applications are then spread over the available system nodes. The main goal of the technique is to minimize average latency. Non-contiguous application mapping improves the throughput of the system, however, mapping on disconnected nodes in non-contiguous mapping increase communication distance and may lead to network delay and increased power consumption. The work presented in [41] tried to improve non-contiguous mapping.

TABLE 1. Comparison of different dynamic mapping techniques.

Ref	Mapping Technique	Experimental Setup	Benchmarks	Comparison with	Optimization Goal
[24]	Compiler-based application mapping	Simics and Orion	SpecFP2000	-	Reduced energy consumption
[25]	Path Load (PL) Dynamic congestion aware task mapping approach	System C based simulation model	-	FF NN MAC MMC	Reduced execution time
[26]	Best neighbor (BN) mapping	VHDL based simulation model	VOPD MPEG-4 MWD RBERG	FF NN MAC MMC PL	Reduced execution time
[27]	User-aware dynamic task allocation	C++ based simulation model	E3S	-	Reduced communication cost
[28]	Energy- and performance-aware incremental mapping	C++ based simulation model	E3S TGFF v3.0	NN	Reduced Communication energy
[29]	Congestion-aware task mapping in heterogeneous MPSoCs	RTLVHDL System-C based simulation model	-	FF NN MAC MMC PL	Reduced Congestion & Latency
[30]	ADAM: Run-time agent-based distributed application mapping	XILINX Virtex2 FPGA based simulation model	MPEG VOPD MWD	nearest-neighbor (NN)	Reduced execution time and computational effort
[31]	Dynamic Spiral Mapping (DSM)	NA	-	FDSM PDSM	Reduced configuration time
[32]	Combination of software and hardware processing elements based mapping	System-C based simulation model	-	NN BN	Minimized Channel load and total execution time
[33]	Similar to [32]	System-C based simulation model	-	NN BN	Reduced Communication overhead
[34]	Contiguous Neighborhood Allocation (CoNA)	Xilinx ML605 Virtex-6 FPGA	-	FF NN INC	Reduced Congestion
[35]	Smart Hill Climbing SHiC technique	NOXIM	-	FF NN INC CoNA	Reduced internal Congestion
[36]	A Hierarchical And Dependency-aware (HAD) task mapping	NOXIM TGFF tool	-	FF, region-based	Reduced Energy consumption and latency
[37]	Runtime distributed task mapping approach DRTRM	Intel Single-Chip Cloud Computer 48 cores	-	DistRM, DRM	Reduced Communication overhead
[38]	Proactive region selection strategy (MapPro)	NOXIM	MPEG-4 VOPD	SHiC, INC NN	Reduced Average Latency
[39]	Multi-agent-based distributed resource management approach DistRM	C based simulation model	-	Centralized	Reduced Communication overhead
[40]	Mixed-criticality run-time task mapping	NOXIM	MPEG-4 UAV VOPD	-	Reduced Average Latency
[41]	WeNA: Deterministic Run-time Task Mapping	GARNET	-	CoNA FF NN	Improved AWMD and Average Latency
[42]	Dynamic task mapping with congestion speculation (DTMCS)	NOXIM	TGFF , E3S	FF NN PL	Reduced Average Latency
[43]	Dark silicon-power-thermal aware runtime mapping	Gem5 , GARNET	TGG E3S	-	Energy , Average Latency and throughput
[44]	Liso (L-shape isolated) dynamic mapping	Gem5	PARSEC SPLASH-2	MapPro, SHiC	Reduced communication interference
[45]	A load balancing inspired optimization framework	Dynamic Application Dependency Graph	Mandelbrot MM.1Stencil MD FFT Dijkstra Blackscholes	Mandelbrot MM.1Stencil MD FFT Dijkstra Black Scholes	Load balancing , speedup and scalability
[46]	User cooperation network coding approach	C++ cycle-accurate simulator.	Apache Db2v Ocean Oracle Sparse	XY−tree CRA CRA+AFD	To avoid network congestion branch-blocking and deadlock

In this approach, mapping of tasks is decided based on the communication volume and occupancy status. In [42], a dynamic task mapping with a congestion speculation (DTMCS) technique has been discussed. In DTMCS similar to [36] HAD approach, no of tasks, of incoming application, are compared with the available idle nodes and mapping process executed if no of tasks is more than the available free nodes. The source task is placed in the first place and the

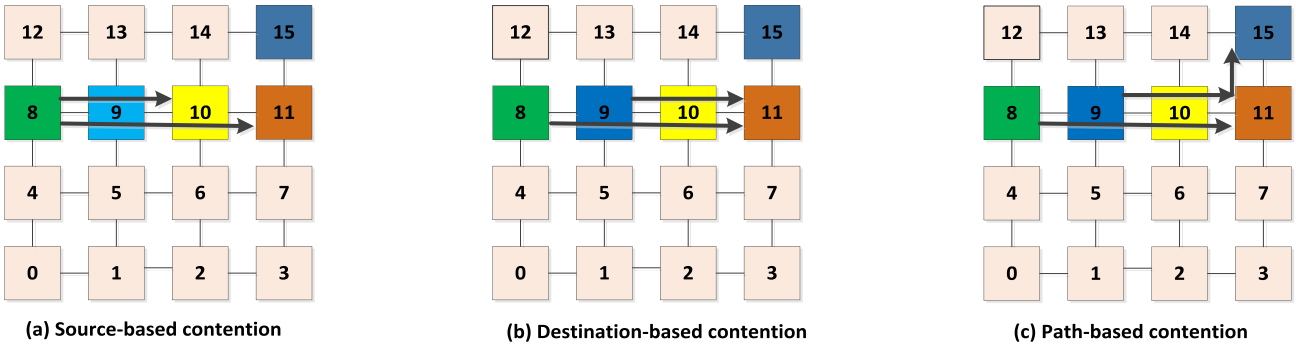


FIGURE 5. Different types of network contentions (a) Source-based contention (b) Destination-based contention and (c) Path-based contention [55].

distance of destination task is decided based on Manhattan Distance. Distance between two points is defined as Manhattan Distance. The proposed algorithm can decrease overall congestion. A thermal aware dynamic mapping technique with a Min Energy mapping algorithm has been presented in [43]. The applications are considered for mapping on the first-come-first-serve approach. If multiple applications request for mapping at once, then the applications are ordered based on communication and computation requirements. The application with the highest demand is mapped first to a node with the highest number of free neighbors, the rest of the applications are mapped around it. A dynamic application mapping approach Liso (L-shape isolated) has been proposed in [44]. Liso provides non-interference communication among multiple applications with common resources by dividing NoC into different isolated secure regions. Inspired by complex network theory of social communities, a novel approach to model the dynamic execution of an application and partition the application into an optimal number of clusters to provide load balancing, speedup and scalability and parallel execution has been presented in [45]. In this work analysis of low level virtual machine (LLVM) intermediate representation (IR) for a specific application adopted to create a dynamic application dependency graph and encoding its memory and computational operations. Subsequently, based on this graph an optimized to find optimal clusters has been proposed that maximized the intra-cluster edges, equalized the execution time of the clusters and kept cluster size equal to core count. In [46], a runtime approach based on user cooperation network coding has been proposed for NoC. In this approach, a corridor routing algorithm and adaptive flit dropping scheme presented to avoid congestion, branch-blocking and deadlock at run-time. **Table 1**, represents the comparison of experimental setup and benchmark used in above stated literature.

B. STATIC MAPPING TECHNIQUES IN NOC

In the static mapping, all application tasks are mapped on cores at design time. Resources on which an application task is going to be operated are defined offline and are not altered after that. Static mapping is also called offline mapping and numerous algorithms have been proposed to find an optimal application mapping. The static mapping application techniques can be classified into two main branches named **(1) Exact (mathematical based) mapping** and **(2) Search (heuristic) based mapping**, as shown in **Fig. 2.** Search based mapping can further be categorized into deterministic search and heuristic search.

1) EXACT (MATHEMATICAL) APPLICATION MAPPING

The Exact mapping or mathematical programming based mapping offers an optimal mapping solution. In [47], authors have reported a Mixed Integer Linear Programming (MILP) based application mapping technique for heterogeneous architectures. This is a combination of hardware and software processes and executes iteratively until the required results are achieved. In this multiprocessor technique, some processors are application-specific, while others are programmable. A MILP formulation for mapping has been proposed in [48]. The cores are mapped onto NoC by taking into account the core placements options, network interfaces for communication and switches for each core. It is stated that the energy consumption is improved when applied on random and real benchmarks

In [49], a two-phase mapping approach for heterogeneous NoC architecture has been presented. For the initial phase, cores are mapped by a greedy mapping approach and later on, in the second phase core placements are improved by applying Tabu search. A MILP based technique computed the size and position of the network components and cores. While MILP techniques produce optimal mapping solutions, the main bottleneck is the execution time. To minimize processing time, a MILP formulation [50], segregated the application task graph into different clusters and defined a custom topology. In this approach, the main optimization objective is to lessen power consumption. Network processors execute high-performance applications by incorporating features like block multithreading and symmetric multiprocessing (SMP). Mapping an application onto such a multifaceted multithreaded and multiprocessing processor is a challenging job. In [51], a two-stage Integer Linear Programming (ILP) based strategy has been proposed for mapping and

process distribution on SMP and block multithreading based processors. In [52], authors have evaluated and presented an ILP based scheme for reducing the energy consumption in an NoC based Chip Multiprocessors (CMP) network through voltage scaling and shutting down specific communication links. Energy-efficient approaches for mapping employed MILP approach have been described in [53] managing operating voltages, routing and application mapping. According to [54], first, the existing ILP formulation has been extended to take both communication and processing energy into account. Subsequently, a Simulated Annealing (SA) approach with Timing Adjustment (SA-TA) presented to enhance the optimization method. The work presented in [55] analyzes the sources that cause network contention. It proposes a contention aware ILP based application mapping solution to minimize the network contention. Different types of network contentions such as path-based, source-based, and destination-based as described in **[55, Fig. 5]**, are discussed and the result shows that packet latency is improved by decreasing the network contention, but with high communication energy.

According to [54], first, the existing ILP formulation has been extended to take both communication and processing energy into account. Subsequently, a Simulated Annealing (SA) approach with Timing Adjustment (SA-TA) presented to enhance the optimization method. The work presented in [55] analyzes the sources that cause network contention. It proposes a contention aware ILP based application mapping solution to minimize the network contention. To reduce energy consumption for different real-time benchmarks, ILP formulation based mapping solution has been discussed in [56]. ILP based techniques define optimum mappings. However, the execution time is very high. To cater to the problem of high execution time, a clustering-based approach has been presented in [57]. As in [50], the mesh network is divided into small zones or smaller meshes called clusters [57]. Clusters are mapped to corresponding smaller meshes and in the end, all smaller meshes are combined to get the final result. It has been reported that the execution time improved with a disadvantage of an increase in the communication cost.

In [58], a cluster-based mapping approach is implemented to cut communication costs and performance is evaluated on multiple benchmarks. ILP based application mapping technique compared with GA, SA, and heuristic-based application mapping techniques in [59]. A mathematical goal-oriented self-organized approach presented in [60] that offers large scale parallelism to cope with the challenges faced by the design of NoC as the backbone of Cyber-Physical Systems (CPS) and to optimize cost functions. In this work, the authors advocate for goal-oriented self-organized distributed algorithms for encouraging self-awareness and autonomy for design cost optimization and predict information about CPS future state. **Table 2** represents the exact different exact application mapping techniques.

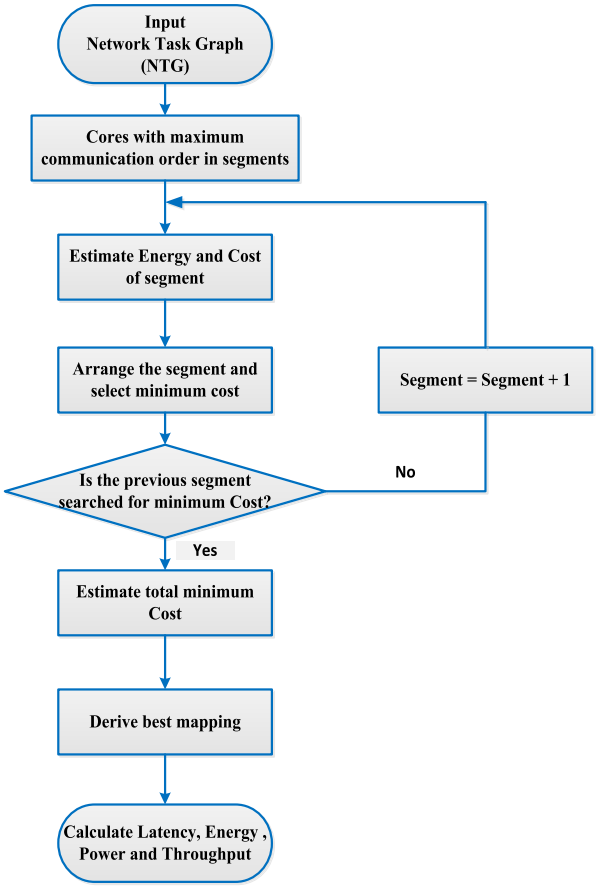


FIGURE 6. SBMAP algorithm [65].

2) SEARCH BASED MAPPING

Search based mapping can be set apart in two succeeding branches, (a) systematic or deterministic search and (b) heuristic search.

a: SYSTEMATIC AND DETERMINISTIC SEARCH

Application mapping algorithms employing Branch-and-Bound (BB) approach belong to the deterministic search. This approach can be used for smaller applications as for large applications it takes more execution time. The flow chart of the BB approach is presented in **Fig. 7**. In [61], authors have proposed an efficient branch-and-bound based scheme to resolve the problem of energy-aware application mapping. Here, the main objective was to reduce the total communication energy of regular network on chip architectures. The same issue was addressed in [62], where authors provided an improved solution to the problem of decreasing the total communication energy by proposing a deadlock-free deterministic routing. An IP core with high communication volume may cause heavy traffic load, which subsequently may become a hotspot on routers. Hotspot results may cause a rise in power density that can upset the reliability of the system. To address this issue, new Network Interfaces (NI) and traffic balanced IP mapping algorithm (TBMAP) has been

TABLE 2. Comparison of different exact mapping techniques.

Ref	Mapping Technique	Experimental Setup	Benchmarks	Result Comparison	Optimization Goal
[47]	MILP	-	H.261	-	Improved execution time
[48]	MILP	-	PIP HDTV MWA VOPD	PMAP NMAP PBB PCTSM NCTSM BCTSM	Reduced energy consumption
[49]	MILP	1 GHz SUN workstation	VOPD MPEG4 PIP MWA		Improved Area and Power
[50]	MILP	Xpress-MP Optimizer for solving MILP problems	H.263 MP3	Clustering PDH PRH	Minimized Power consumption
[51]	MILP	Intel IXA SDK simulator	-	IPSec ,AH , Diffserv, IPv4	Improved Runtime and throughput
[52]	MILP	Spec95 benchmark set	-	link shutdown, link shutdown with voltage scaling	Normalized Energy consumption
[53]	MILP	ILOG CPLEX 10.0 Concert technology for MILP	MPEG4 MWD OPD	Optimal mapping	Improved energy consumption
[54]	ILP	TGFF TG3	-	Pure SA heuristic	Reduced Computation time
[55]	ILP	lp-solve optimizer TGFF C++ based simulator	MPEG4 Parallel-1Parallel-2	ILP	Reduced Packet Latency
[56]	ILP	Xpress-MP tool for ILP	VOPD MPEG4 H.263 MP3	total communication time	Reduced Power consumption
[57]	ILP	NA	VOPD MPEG4 H.263 MP3	ILP	Reduced Execution time
[58]	ILP	Matlab	MPEG4 PIP	KL	Reduced Communication cost
[59]	ILP	TGFF Xpress-MP	VOPD MPEG-4 MWD H.263 MP3	GA SA CastNet ILP	Efficiency comparison of ILP, GA, SA, and heuristic methods
[60]	Cyber-Physical Systems (CPS)	Cyber physical task graph (CPTG)	Bio-sensor	-	Optimized design cost

presented in [63]. TBMAP used an improved and modified branch-and-bound technique. The traffic is balanced without compromising on the network performance. However, some paths can become longer to achieve a balance which may cause high communication costs. A two-step bandwidth constrained algorithm called Elixir has been presented in [64]. In the first step mappings having the smallest communication cost are defined by the search tree approach. Finally, the best mapping with minimum latency and power consumption is obtained by employing the Polaris toolchain technique.

In [65], authors have proposed a bandwidth-constrained multi-objective segmented brute-force mapping (SBMAP) algorithm for NoC. SBMAP reduces the computational complexity and communication energy of the NoC design. This algorithm segregates the application into multiple segments and applies a modular system search to find the optimized mapping solution. **[65, Fig. 6]** describes the flow chart of the SBMAP algorithm and **Table 3** represents the comparison of experimental setups and benchmarks used in deterministic search-based techniques.

b: HEURISTIC SEARCH

Various heuristic search mapping solutions have been proposed in NoC. These approaches can be ordered as transformative and constructive heuristics.

i) TRANSFORMATIVE HEURISTICS

Transformative heuristics use prevailing optimization solutions to achieve improved mapping solutions for application mapping in NoCs. Some prominent examples of such mapping solutions are the Genetic Algorithm (GA), Ant Colony Optimization (ACO), and Particle Swarm Optimization (PSO).

Genetic Algorithm (GA) by David E. Goldberg is a stochastic search algorithm derived from natural genetics operations. The chromosomes population grows over a specified number of generations based on the proposition of natural selection. A part of the population of the new generation usually evolves by directly copying top and fit chromosomes from the previous generation. The remaining population is formed by crossover and mutation. In crossover two chromosomes based on fitness, criteria are selected as parent

TABLE 3. Comparison of different deterministic search based mapping techniques.

Ref	Mapping Technique	Experimental Setup	Benchmarks	Result Comparison	Optimization Goal
[61]	Deterministic search	-	H.263 MP3	SA	Reduced Total Communication Energy
[62]	Deterministic search	-	H.263 MP3	SA	Reduced Total Communication Energy
[63]	Deterministic search	TSMC 0.13m CMOS technology	MPEG4	BMAP EPAM	Traffic load balancing
[64]	Deterministic search	Polaris toolchain	MPEG4 VOPD	PBB BMAP Onyx	Improved Power consumption and average latency
[65]	Deterministic search	NoCTweak Simulator	VOPD PIP MMS 80211arx	BB NMAP Random	Reduced power consumption and communication cost

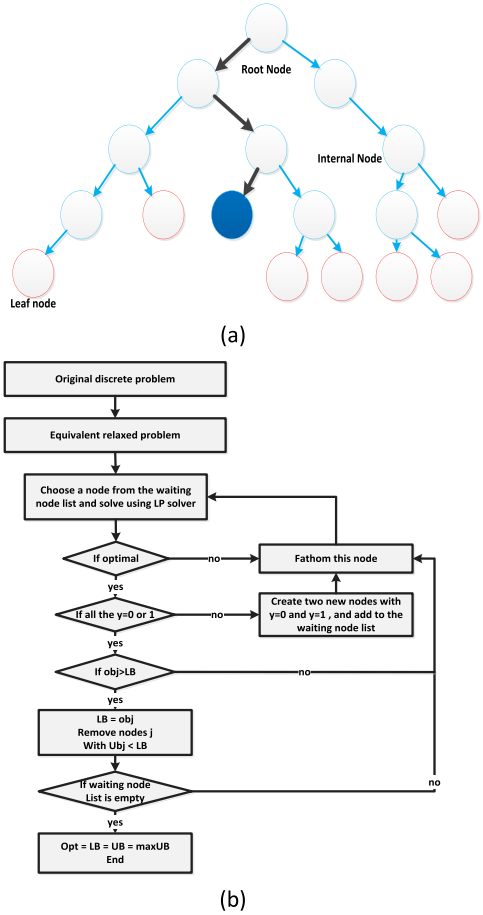


FIGURE 7. Branch and bound approach (a) Branch-and-bound search tree (b) Flow diagram of branch-and-bound technique.

chromosomes and their parts are used to form new offspring. While in mutation, portions of a parent chromosome are modified.

Particle Swarm Optimization (PSO) technique is animated by the social behavior of fish schooling and bird flocking. Here, multiple contender solutions, called a particle, cooperate with neighboring particles and evolve into problem space established on the experience of individual particle and of neighboring particles. The fitness value is the criteria to estimate the worth of any particle in search space [66]. **Ant Colony Optimization (ACO)** technique [67] is a probabilistic technique centered on population and inspired by the natural conduct of ants in navigating the trails from their colony to a food source. Once a path is defined by one ant all ants are expected to follow that trail. Other optimization approaches such as firefly optimization algorithm and artificial bee colony optimization algorithm based application mapping approaches have also been reported.

In the work [68], a two-step G A based application mapping that minimizes the execution time has been proposed. In the first step, application tasks are allocated to cores with speculation that all edge delays are persistent and equal to the overall edge delay. In the second step, to reduce the total delay of the system, the cores are assigned to tiles of NoC based on the actual delay of the network traffic model. Delay factors such as packet length, network contention have not considered in this work. A genetic algorithm (GA) based mapping [69] takes important delay factors mentioned above into account and perform mapping with reduced average delay. The population corresponds to the critical positions of the NoC. Initially, the population is selected randomly and after that to form a new generation, a multi-point crossover is employed with the randomly selected population. The fitness function, in this case, is the average waiting time. The number of cores in the constraint core graph is equal to the size of a chromosome. The chromosome with lower waiting time takes part in the crossover. The mutation is executed randomly to achieve local or global minima and this process performed repeatedly until the lowest average waiting time remained constant for a specific number of iterations.

In [70], a genetic algorithm called CGMAP presented, which finds application mapping by using the advantages of chaotic systems as an alternative of random processes in GA. It is reported that the offered algorithm performed well when compared to previous similar algorithms. Another two-phase application mapping algorithm GAMR based on GA approach has been reported in [71], in the first phase, it finds the suitable mapping of cores while in the second stage, GAMR creates deadlock-free routing for communication paths between nodes of NoC without disturbing the arrangement of cores created in the first phase. A3MAP [72]

TABLE 4. Comparison of different transformative heuristics based mapping techniques.

Ref	Mapping Techniq ue	Experimental Setup	Benchmarks	Result Comparison	Optimization Goal
[68]	GA	TGFF	-	Different GA based sceneries	Reduced average system delay
[69]	GA	Verilog-HDL	-	GA Random	Reduced average system delay
[70]	GA	NS-2	VOPD MPEG4	BMAP PBB NMAP	Improved Energy consumption, Latency and Communication cost
[71]	GA	-	VOPD MPEG4 MWD H.263 MP3	EPAM-OE NMAP	Reduced energy consumption and link bandwidth
[72]	GA	C++ based implementation	VOPD MPEG4 E3S	NMAP A3MAP-SR	Improved hop count and runtime
[73]	PSO	Java based implementation	-	GA	Reduced Execution time and Energy
[74]	PSO	System-C based simulator	VOPD MPEG4 PIP	NMAP LMAP BMAP GMAP PBB	Improved network latency, throughput and communication cost
[75]	PSO	-	VOPD MPEG4	-	Communication cost minimization
[76]	PSO	NOXIM	DVOPD MPEG4 PIP MWD H.263 MP3	Modified NMAP+MPSO NMAP	Reduced communication cost
[77]	DPSO	System-C based simulator , TGFF	DVOPD MPEG4 PIP MWD H.263 MP3	KL_BFT	Reduced communication cost
[78]	DPSO	TGFF CGG	VOPD DVOPD MPEG4 PIP	NMAP LMAP PSMAP GMAP DPSO	Reduced Communication cost
[79]	PSO+MI LP	C++ based implementation	PARSEC SPLASH-2	NMAP RMAP	Reliability MTTF
[80]	PSO +SCA	Matlab R2015b	VOPD MPEG4 PIP MMS MWD	BPSO GA ACO RAND	Improved in Energy consumption
[81]	PSO	TGFF tool	MPEG MWD	-	Reduced Communication cost
[82]	ACO	C++	VOPD MPEG4	Random	Reduced Communication bandwidth
[83]	ACO	TGFF CGG	-	Random	Energy consumption improvement
[84]	HPGA	Cycle-accurate NoC simulator in C++	Unger273d.6 Unger273d.7 Unger273d.8 Unger273d.9 Unger273d.10 Dill.2 Dill.3	Single-Slave 24-island	Improvement in speedup and reduction in hardware overhead

algorithm proposed for regular and irregular mesh-based NoC with homogenous and heterogeneous cores. The partition-based task mapping is found by two heuristics, an optimized mapping based on GA and a successive relaxation algorithm for fast mapping, which offers a better trade-off between runtime and performance. A multi-objective hybrid scheme presented in [73], where the PSO technique with Dijkstra's shortest path algorithm is used to decide the shortest path between cores and to improve overall performance. Work proposed in [74] calculated application mapping for NoC by using the PSO technique. Multiple real-time benchmark applications used to evaluate performance and results were compared with previous application mapping approaches. According to work presented in [75], multiple mappings solution generated and best mapping is decided based on three performance constraints. They are communication cost, contention factor and robustness index. A new application mapping technique reported in [76], where the Modified NMAP algorithm in combination with the PSO technique used for finding the best mapping solution. In [77], authors have discussed a new task mapping technique on Butterfly-Fat-Tree (BFT) based NoC, where they used Discrete Particle Swarm Optimization (DPSO) strategy for application mapping. In [78], authors have reported a hybrid technique for application mapping for NoC in combination with the Tabu search to expand the search space. The deflection approach forced swarm particles to discover more solution space before converging to the global best result.

A reliability aware model for mapping in NoC has been proposed in [79], where an amplified version of the standard PSO technique is used with a constructive heuristic to improve the reliability. The deterministic algorithms need significant computation time to explore the ideal mapping result. Meta-heuristic algorithms (MA) offered a solution to this problem; however, most MAs are aimed for uninterrupted problems and undergo early convergence. Mapping based

TABLE 5. Comparison of different Constructive heuristic with iterative improvement based mapping techniques.

Ref	Mapping Algorithm	Experimental Setup	Benchmarks	Result Comparison	Optimization Goal
[85]	NMAP	System C based simulator	MPEG4 DSD OPD MWA MWAG PIP	DPMAP DGMA	Improved Communication bandwidth and cost
[86]	MOCA		VOPD MPEG4 MWD	NMAP MILP	Reduced Energy consumption
[87]	CSA		VOP	NMAP SA	Reduced Communication cost
[88]	Onyx		VOPD MPEG4	PMAP GMAP PBB BMAP NMAP	Improved communication cost
[90]	Crinkle	SMAP tool	VOPD	NMAP GA Random Spiral	Improved communication cost, energy consumption and execution time.
[91]	LMAP	System C based simulator	VOPD MPEG4 PIP	NMAP BMAP	Improved Communication cost and average network latency
[92]	Map_graph	NOXIM	VOPD MPEG-4 PIP	ILP NMAP PSMAP	Improved Communication cost
[89]	RASMAP	NOXIM	VOPD MPEG4 MWD H.263 MP3 DVOPD	NMAP LMAP PSMAP CASTNET	Improved Communication cost
[93]	XY-ADB	NOXIM	VOPD MPEG4	RMAP ONYX CRINKLE	Improved Average Latency

on binary meta-heuristic has been proposed in [80], which offers a better balance between exploration and exploitation to solve the mapping issue. An adaptive scheme is employed to integrate the Sine Cosine Algorithm (SCA) and Particle Swarm Algorithm (PSO) to traverse search space more efficiently to converge on a globally optimal solution. A two-phase mapping strategy based on PSO used for reconfigurable design to curtail the communication cost has been discussed in [81]. A global mapping is achieved by merging multiple applications during the initial phase and multiplexers are used to reconfigure by relocating the cores to nearby routers. The proposed algorithm has not been compared with any existing approach. The bandwidth requirements have been minimized by the ACO based mapping approach [82]. In the work [83], a multi-objective ACO based approach is applied to discover the optimal solution for mapping that optimized the energy consumption and hotspot temperature. A hierarchical parallel genetic algorithm (HPGA) has been proposed for multi-core System-on-Chip (SoC) in [84], a new architecture consists of two multiplexing schemes specifically dynamic injection bandwidth multiplexing (DIBM) and time-division based island multiplexing (TDIM) with a task-aware adaptive routing algorithm presented to enhance speedup and minimize the hardware overhead. A quick summary of different transformative heuristic techniques is presented in **Table 4.**

ii) CONSTRUCTIVE HEURISTICS

In this type of heuristic, half-done mapping results are produced in sequence, and in the end, the decisive solution for application mapping is attained. The constructive heuristic is of two main types, heuristics without iterative improvement or with iterative improvement. Techniques based on constructive heuristic searches are generally much faster than the transformative heuristics.

- ***CONSTRUCTIVE HEURISTIC WITH ITERATIVE IMPROVEMENT***

In this approach, a preliminary solution on some predefined criteria is used to map cores from the core graph onto a topology graph. After initial placement, an iterative improvement procedure is performed on the initial mapping to achieve the improved final solution.

In [85], a three-phase mapping method to minimize average communication delay has been proposed which fulfills the bandwidth constraint. A polynomial time constructive heuristic approach named MOCA for mesh-based NoC architectures with low energy requirements has been presented in [86]. It is reported that the MOCA algorithm is less complex than that of NMAP [85]. A Cluster-based mapping approach combined with simulated annealing has been presented in [87]. Initially, cluster-based mapping is performed and then improved by applying a simulated annealing approach to derive a final mapping solution. In [88], a less complex bandwidth constrained constructive heuristic algorithm (Onyx) has been suggested. In this technique, at first, the core having the maximum communication bandwidth placed at the center of the network. Cores to be mapped next are ranked based on their communication demand with previously mapped cores. By using the lozenge-shaped path approach, the unallocated cores are placed at the distance of one or two hops from the related core until all empty tiles are mapped. A similar approach has been discussed in [89]. In [90], priority lists are created based on the communication bandwidth and degree of interconnection between nodes before mapping. Tasks are then mapped in a zigzag manner starting from one corner and end on another corner of the mesh-based NoC. LMAP algorithm has been reported in [91]. Here, the Kernighan-Lin bi-partitioning K-L scheme has been used to discover the

TABLE 6. Comparison of different constructive heuristic without iterative improvement-based mapping techniques.

Ref	Mapping Algorithm	Experimental Setup	Benchmarks	Result Comparison	Optimization Goal
[94]	PMAP	-	-	NN_embed , Optimal	Improved communication cost
[95]	UMARS	cycle-accurate systemC simulator	MPEG	Clustering Naïve optimized	Reduced communication energy
[96]	SMAP	SMAP simulation tool		GA , Random	Reduced communication energy
[97]	BMAP	-	VOPD MPEG4	NMAP, GMAP, PMAP	Improved Traffic load balance and hop count
[98]	RMAP	Cycle-accurate simulator	OPD	Random	Reduced Reliability and average packet delivery time
[99]	CastNet	-	VOPD H.263 MPEG-4 MP3 MWD G25 G36	ILP, GA ,SA , Random	Improved communication cost
[100]	BMA	-	VOPD DVOPD MPEG-4 H.263 MP3	ILP, NMAP, CastNet ,Onyx	Improved communication cost and power consumption
[101]	CHMAP	Simulation environment written in Java TGFF	VOPD H.263 MPEG-4 MP3 MWD PIP	NMAP CASTNET ILP	Improved communication cost
[102]	mapGtoM	Algorithm implemented in C	VOPD H.263 MPEG-4 MP3 MWD	NMAP MOCA CastNet CHMAP	Improved communication energy and CPU time

placement of cores by examining their bandwidth demand. The iterative improvement phase further used to refine the preliminary mapping. A sequential relaxation algorithm resolves the mapping and a genetic algorithm is realized as an initial mapping to yield improved final mapping **[55].**

In [92], a performance-aware mapping algorithm (Map_Graph) has been proposed. It displays rational enhancement in communication costs while taking into account the static operation of the system. The offered techniques show improvement in energy consumption and dynamic performance of the mapping solutions when compared to near-optimal ILP mapping. The work presented in [93], has hired the abstract graph mapping approach as a replacement for common application core graphs which has been employed in most mapping techniques. **Table 5** shows a summary of constructive heuristics with iterative improvement based mapping techniques.

- ***CONSTRUCTIVE HEURISTIC WITHOUT ITERATIVE IMPROVEMENT***

Mapping is performed based on some predefined criteria by assigning cores one by one onto the NoC topology graph. However, no optimization technique is applied to the primary mapping solution to derive a better mapping solution and the initial placement of cores remains unchanged. PMAP a two-stage mapping algorithm has been offered in [94], where clusters having high communication are assigned on parallel nodes. A unified mapping approach with time slot allocation and routing algorithm has been reported in [95]. SMAP [96] is a simulation-based application mapping algorithm that offers a reduction in communication energy and execution time. In this strategy, the task with top priority is placed at the center and rests of the tasks are placed spirally starting from the initially assigned/mapped task to the boundaries of the NoC. A binomial mapping scheme has been proposed in [97]. Cores are ranked based on the communication bandwidth between them. Subject to the IP core ranking, the IP core sets with the highest communication are merged two-by-two on each iteration. RMAP [98] mapping approach contemplates transient errors arising in switches and channels of the NoC caused by the data spreading over the channels. In RMAP, the application graph is divided into two subdomains and channels with fewer communication loads are used to route packets redundantly to achieve reliability.

CastNet [99] maintained a significance list of the application tasks based on overall and average communication bandwidth. Initial mapping is performed based on a priority list. The initial cores to be mapped are selected from different symmetry groups. This procedure repeated for the rest of the tasks for mapping. A set of mapping solutions are created

and a priority list is updated after each mapping. In the end, from a set of mapping solutions, the best mapping solution selected as a final application mapping solution. Cellular Learning Automata-based optimized mapping algorithm for NoC designs has been reported in [100]. In this strategy, set of five actions used features of Cellular Learning Automata to select the best states for every cell based on reduced communication cost.

In [101] maximum spanning tree is constructed by extracting the most prominent part of the application based on average communication cost. Mapping is decided based on the effects of communication cost, the longest path to descendants, and the core's degree. Finally, the most suitable tile to be mapped is selected which has more free neighbors and available connections. Two-step algorithm mapGtoM has been offered in [102]. In the first step, constructive application mapping is performed by selecting the tiles with tile symmetry as initial tiles for mapping and nearby cores are mapped close to each other.

Multiple mapping solutions are generated, and the best solution selected based on total communication energy. In the second step, all cores with no active neighbors swap their places providing they devour less energy than the previous state. **Table 6,** presents a summary of constructive heuristics without iterative improvement based mapping techniques. Task graphs of MPEG-4 and VOPD with final application mappings results of few selected mapping algorithms are represented in **Fig. 8**.

III. SUMMARY

In this study, we have tried to highlight a comprehensive review of the static and dynamic application mapping problem. As discussed earlier, the application mapping is an NP-hard problem and various heuristic algorithms have been proposed to optimize different performance metrics. A general observation regarding static mapping strategies is that most of the work focused on minimizing the communication energy consumption with the communication bandwidth on the network links used as a constraint. The algorithms presented in the static mapping domain try to optimize multiple traits using specific models, making certain assumptions, application mapping is performed at design-time, and it is not going to change during run-time or dynamically. In this approach, the platform is dedicated to operate on a specific application and dynamic features such as task insertion, task migration and removal are not considered during run-time. Search algorithms using Branch-and-Bound (BB) belong to this category. It is a systematic search algorithm that topologically finds the mapping by searching the solution in tree branches and bounding unallowable solutions. It can be applied to smaller problems, as search time grows exponentially with the size of the problem. Different from the static mapping approach, in dynamic mapping, application mapping is executed during run-time. In dynamic mapping, the time consumed by the mapping algorithms to run and produce an optimal solution is critical since it converts into an overhead to the overall execution time. A compromise has to be made between execution time and performance. The dynamic mapping algorithms are required to perform efficiently to provide quick output so greedy heuristic approaches are employed that locally search the solution space. However, these types of heuristic techniques do not guarantee to find globally optimal solutions or to achieve a required performance within a specific time. Therefore, the inherent non-deterministic hard real-time applications could not depend on dynamic mapping. The different mapping approaches reviewed in this study presented different mapping solutions that try to offer optimal mapping solutions making certain assumptions and using specific models. This review provides useful insight into many of these approaches that could help understand the problem of mapping real-time applications.

IV. CURRENT AND FUTURE CHALLENGES IN APPLICATION MAPPING

This section discusses some of the upcoming trends and challenges to be faced to take the mapping approaches into the next era. Some of the trends are addressed in the following section.

A. HYBRID APPLICATION MAPPING

Three types of application mapping techniques have been reported: Design-time mapping (static mapping), run-time (dynamic mapping) and hybrid mapping techniques. Mostly the design time techniques have been reported in the literature. Though, their incapability to manage dynamic/ real-time scenarios gave rise to the research on dynamic mapping techniques. Dynamic mapping strategies overcome the limitations of handling dynamic workload requirements in real-time but with the consequences of inefficient application mapping due to inadequate computational power in real-time. Consequently, the problems of the design stage and run-time techniques have been catered by hybrid application mapping techniques. Hybrid techniques try to integrate the benefits of both domains. The hybrid technique incorporates the features of design time approaches with real-time controlling and minimum computation time to find efficient mapping solutions for incoming applications.

In [103], authors have proposed a hybrid strategy that contemplates a design space examination at design time joined with appropriate online performance analysis. A tree-based hybrid application mapping strategy that explored the design space for heterogeneous MPSoC systems has been reported in [104]. Application clustering is integrated with a tree-based method during design time, while application classification and feature extraction are performed online based on prominent machine learning approaches. Work presented in [105] analyzed two techniques for hybrid application mapping, a backtracking problem-specific technique, and a general-purpose SAT solver. In [106], authors have presented a hybrid scheduling algorithm called HYSTERY for heterogeneous multiprocessor systems. Where design challenges are optimized at an offline phase by resolving

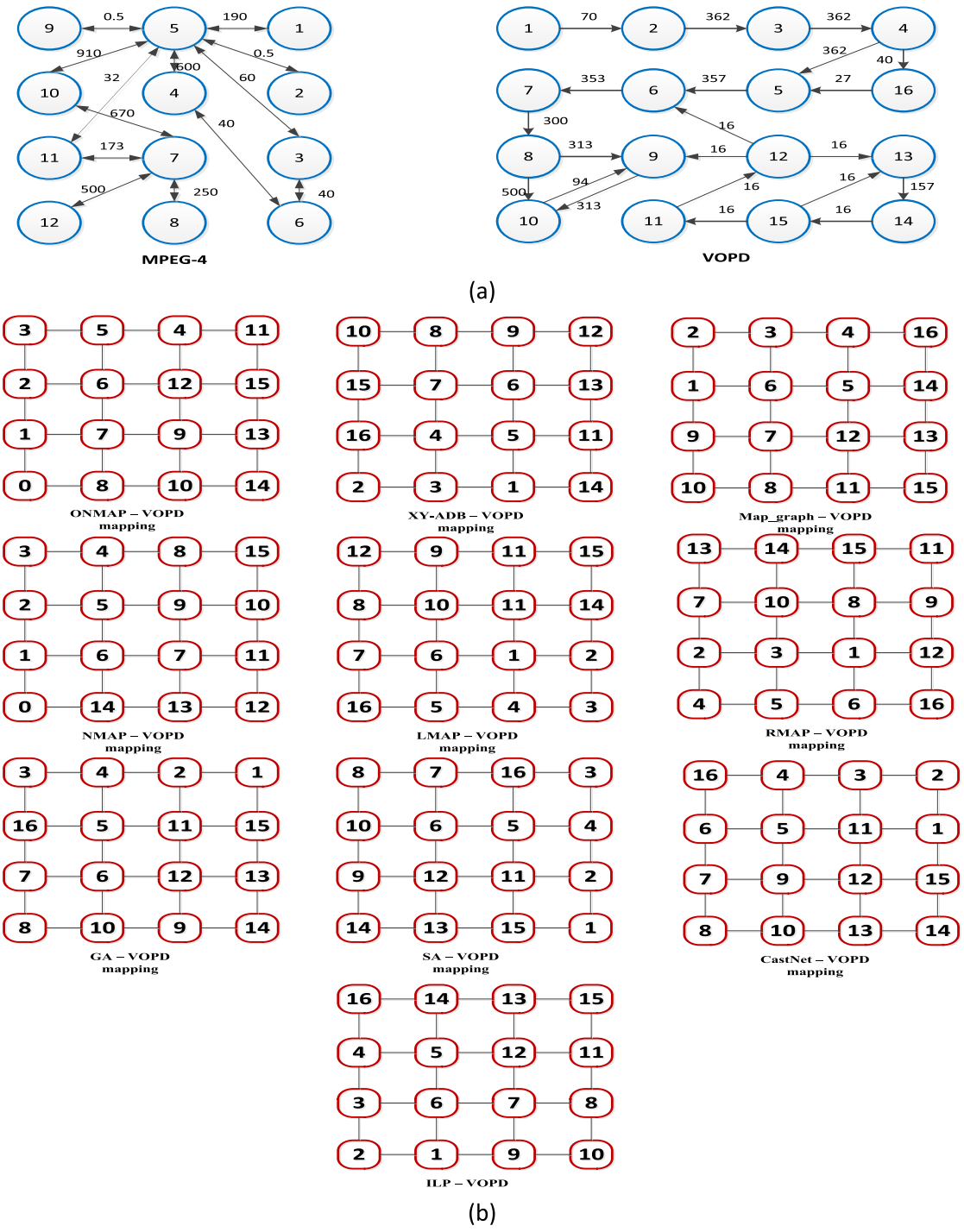


FIGURE 8. Mapping results of different application mapping algorithms (a) Task graph of MPEG-4 and VOPD (b) VOPD mappings.

an optimization problem and take into account the load balancing in task allocation. The mapping solution derived at design time is applied and the design parameters are analyzed periodically and controlled during run time. A hybrid approach Self-aware System-on-chip using a Static-dynamic (SSS) approach has been discussed in [107] to realize self-awareness, self-optimization, and minimization of hotspots in NoC design. Static mapping is used for self-optimization while for task migration a dynamic task manager has been proposed. In [108], authors have proposed an optimized, search based near-optimal mapping hybrid mapping algorithm called ONMAP which exploits the properties of NMAP and exact optimization approaches. The search space for mapping real-world application is optimized by segmentation

to achieve fast simulation results. BEMAP [109] algorithm employs the methodical search optimization and modular exact approach to achieve an improved multi-objective mapping solution for the application mapping problem of the NoC design.

B. MACHINE LEARNING AND NEURAL NETWORKS APPLICATION MAPPING

The acceptance and reputation of machine learning approaches have been on the rise in the previous decade. The learning feature of these procedures has made them a better choice for resource allocation and prediction. The machine learning based methods mostly need a large amount of data set to learn and then predict for a specific platform. If the system conditions remain the same during the earning or training phase and after it, high precision in prediction and correction of the design parameters can be achieved by using advanced machine learning and deep neural network methods. The machine learning methods can be categorized based on their capability to adjust to changes. The training phase of these methods is data-intensive, time-consuming and also requires thorough analysis and tuning at the design stage. These approaches trained with the help of predetermined platforms and workloads and improve them at runtime by learning new changes in the behavior of workload to enhance prediction precision and management efficacy. Artificial Neural Networks (ANNs) such as Convolutional neural networks (CNNs) and deep neural networks (DNNs) are machine learning models that take a set of inputs and automatically learn to estimate a target function based on it.

A machine learning based self-optimizing and self-programming computing system (SOSPCS) design framework for heterogeneous systems has been proposed in [110]. In the first phase, at compile-time, a task pool comprising of hybrid tasks with different processing elements (PE) grouped according to target applications. Neural networks identified the tasks from target applications to be run on GPUs. Next, at runtime, a distributed reinforcement learning-based technique is used to map the selected tasks onto the heterogeneous PEs. SVR-NoC, a learning based support vector regression (SVR) model for evaluating Network-on-Chip (NoC) latency performance has been proposed in [111]. Instead of using classical queuing theory SVR-NoC uses a kernel-based support vector regression method to predict the channel average waiting time and the traffic flow latency. A neural networks model, Neuro-NoC for online resource monitoring and configuration in many-core NoCs is proposed in [112] to achieve power-thermal efficiency in the dark silicon era. Neuro-NoC model exploits the neural networks learning algorithm and proposed a distributed cluster-wise neural network and a global neural network model to dynamically predict, configure and monitor NoC resources based on runtime learning of the system status.

In [113] several NoC topologies have been discussed and a concentrated mesh NoC architecture considering the communication patterns of deep neural networks has been proposed. Furthermore, a computation and communication aware mathematical model and mapping solution have been proposed which increases the parallelism and minimizes the hotspots in the NoC by distributing the loads evenly all over the NoC. The authors present a Cross-Layer based neural mapping method CLAMP [114] that maps synaptically connected neurons belonging to adjacent layers into the same on-chip network node. The strategy is suitable for different SNN topologies including all-to-all connection and connection partially layer by layer. To adapt to various input patterns, the approach also considers input spike rate and remap neurons for improving mapping efficiency. The method helps to reduce inter-core communication costs. An NoC-based DNN accelerator design paradigm is presented in [115]. The proposed accelerator is based on mesh topology, neuron clustering, random mapping, and XY-routing and offers a reduction in off-chip memory accesses with a flexible interconnect that optimized communication between processing elements on the chip. In [116], a novel NoC design Neu-NoC is proposed to optimize the traffic in neuromorphic acceleration systems. A set of techniques, i.e., sparsity-aware traffic reduction, NN-aware mapping, and multicast are proposed to reduce average hops account and redundant traffics. Neu-NoC can effectively reduce the average packet latency and energy consumption. To facilitate the evaluation of Neural Networks (NNs) such as Artificial Neural Network (ANN) and Convolutional Neural Network (CNN) on NoC, cycle-accurate NoC-based neural network simulators have been proposed in [117], [118].

C. APPLICATION MAPPING FOR 3D NoC

Integration of multiple layers of processing cores into a single system as 3D designs lead to reduce signal transmission delay, area, and power. These advantages make 3D multi-core architectures a prospective substitute to be used in future high-performance computing systems. Regardless of having several advantages, the 3D high integration density brings major apprehensions in the temperature increase that causes thermal hot spots and high-temperature gradients. This might lead to an unreliable system and degraded performance. The efficient thermal management of 3D architectures is challenging and requires an investigation of efficient methodologies. An energy and performance aware application mapping algorithm for inhomogeneous 3D NoCs is presented in [119]. In [120] an efficient runtime mapping algorithm to reduce both communication latency and overall application running time under thermal constraint for 3D NoCs is proposed. Work in [121] presents a knowledge-based multi-objective KBMA-based approach for successful mapping of IP blocks to 3D NoC tiles with standard NoC architectures like mesh, torus, ring, and BFT. Various prominent research activities in the domain of 3D NoC system design and test have been addressed in detail by authors in [122]. However, the development of efficient mapping approaches taking thermal issues into account for 3D architectures will continue in the coming future. Further, 3D heterogeneous architectures will require to be considered for better accomplishing

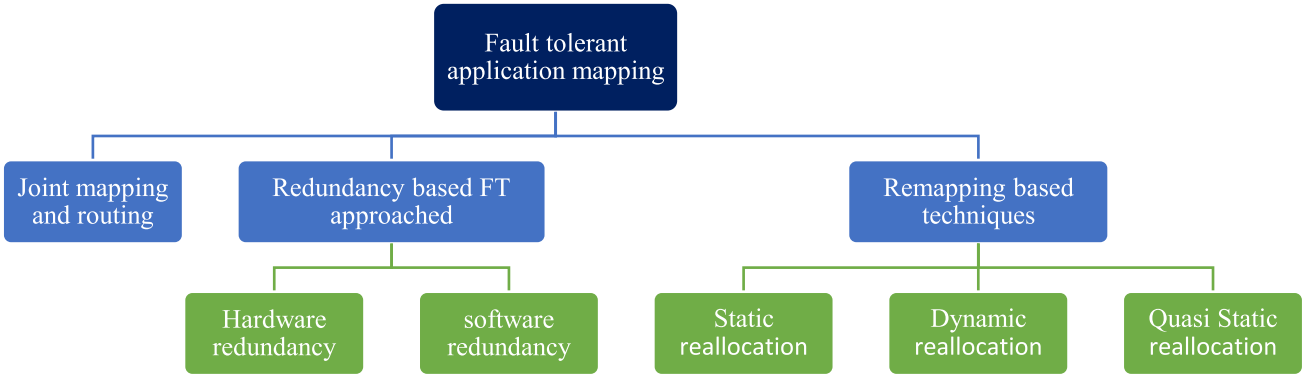


FIGURE 9. Fault-tolerant application mapping in NoC [6].

the increasing functional and non-functional demands. Heterogeneity inflicts further challenges for managing different types of cores. Some added challenges also need to be addressed to take the mapping approaches into the next era. For example, development of efficient programming models for large scale and 3D architectures, efficient synchronization, debugging of several concurrent executions and control of concurrently executing tasks on such architectures.

D. FAULT TOLERANCE APPLICATION MAPPING IN NOC

Ever since the density of transistors on integrated systems has increased and applications are becoming more complex. The reliability of the system has grown into a major challenge in NoC design [123]. Various research works have discussed the issue of reliability in NoC, addressed multiple problems and factors causing reliability issues and proposed fault-tolerant models and frameworks to detect and diagnose faults [124]–[126]. Work presented in [123] categorizes reliability into fault avoidance and fault tolerance. These approaches are used just after the detection and diagnosis steps in the reliability enhancement procedure. The organization of fault-tolerant approaches based on redundancy has been discussed in [127]. Authors in [128] have presented a low cost and efficient fault-tolerant router design for reliable NoC. Techniques presented in this particular design can prevent the router from permanent faults. Transient faults are also the major reason of failure in the NoC design and need prevention. A forward error correction based transient fault mitigation technique is presented in [129].

- **CATEGORIZATION OF FAULT-TOLERANT APPLICATION MAPPING**

Based on the central proposition implemented to address the issue of reliability in NoC, three classes of fault-tolerant application mapping can be pointed out [127].

1. Joint mapping and routing based approaches.
2. Redundancy based approaches.
3. Task remapping based approaches.

[6, Fig. 9], presents a summary of the fault-tolerant application mapping approaches. The detail of each category is given below.

1) JOINT MAPPING AND ROUTING BASED APPROACH

In this approach, mapping and routing approaches are combined to increase reliability. The application and network layers are evaluated together for fault tolerance in NoC.

2) REDUNDANCY BASED FAULT TOLERANCE APPROACH

Redundancy is a definitive approach typically employed to tolerate different kinds of faults and classified into the hardware and software redundancy-based fault tolerance approaches. Spare parts are used to avoid faults in hardware redundancy based approaches while communication or tasks are replicated for fault tolerance in software redundancy approaches [127].

3) FAULT-TOLERANT BASED ON REALLOCATION

Reallocation or task migration-based fault-tolerant techniques are mostly used to tolerate faults at a system level. Task migration involves relocating a task from one node to another for execution based constraints and performance requirements of NoC. The fault-tolerant FT approaches based on task migration can be classified into the following categories [130].

i. **Static reallocation:** In static reallocation and task migration technique, assignments and mappings are performed and protected offline at the design stage for various fault scenarios [131].
ii. **Dynamic reallocation:** Contrasting to static reallocation, the choice of tasks migration in dynamic reallocation techniques is made online during run-time concerning multiple measures, such as new enabling applications, fault setups, resource readiness, and NoC status. The migration approach, in this case, it needs to contemplate various constraints, including migration overhead [6].
iii. **Quasi-Static reallocation:** These approaches make use of the benefits of both static and dynamic reallocation approaches. The task migration decisions are performed online based on analysis done offline [6].

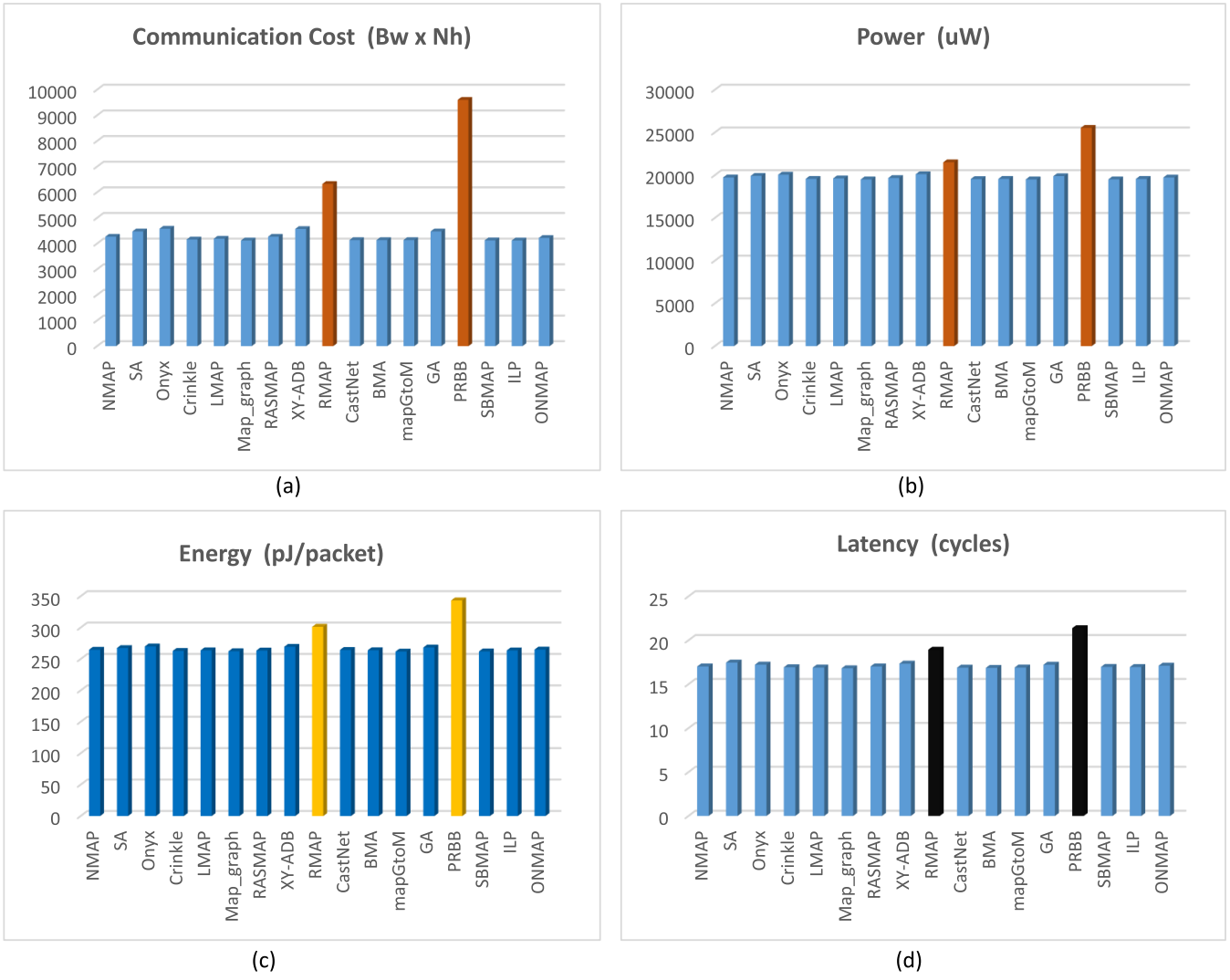


FIGURE 10. Performance factors comparison of different application mapping algorithms on VOPD application.

V. RESULTS AND PERFORMANCE COMPARISON

- ***SIMULATION SETUP***

The performance of application mapping algorithms is usually evaluated on a set of benchmarks. Based on the literature survey three broadly used benchmarks are VOPD, MPEG-4, and PIP. For performance comparison and evaluation, we have filtered out and selected the literature which has given final application mapping results of VOPD and MPEG4 benchmarks for mesh-based NoC architecture. NoCTweak [132] is an open-source SystemC based cycle-accurate NoC simulator. NoCTweak has the choice of multiple performance parameters like power, energy, latency, link bandwidth, and throughput. NoCTweak can be integrated with the ORION tool [133], [134] for finding the power at different CMOS nodes based on its computational power models. RTL designs in Verilog of all router components were synthesized with Synopsys Design Compiler and placed & routed with Cadence SoC Encounter using a 65 nm CMOS standard cell library. Post-layout power data of these components are fed to the simulator for power and energy estimation based on the activities of components while running a certain traffic pattern. However, it has limited NoC design capabilities like topology synthesis, mapping optimization, and accurate performance and energy models.

To overcome the shortcomings of the available NoC tools, the NoCTweak simulator has been modified to perform analysis and evaluation of different application mapping algorithms. The new modified NoC simulation framework, named as ENoCTweak is developed in this research work. This framework is mainly based on the NoCTweak simulator with added parameters and integration of other NoC components and simulation platforms. All algorithms having final mapping results are simulated on the ENoCTweak simulator on the Intel Core i5 platform with 8GB main memory and 2.5 GHz clock frequency. Finally, simulated results are

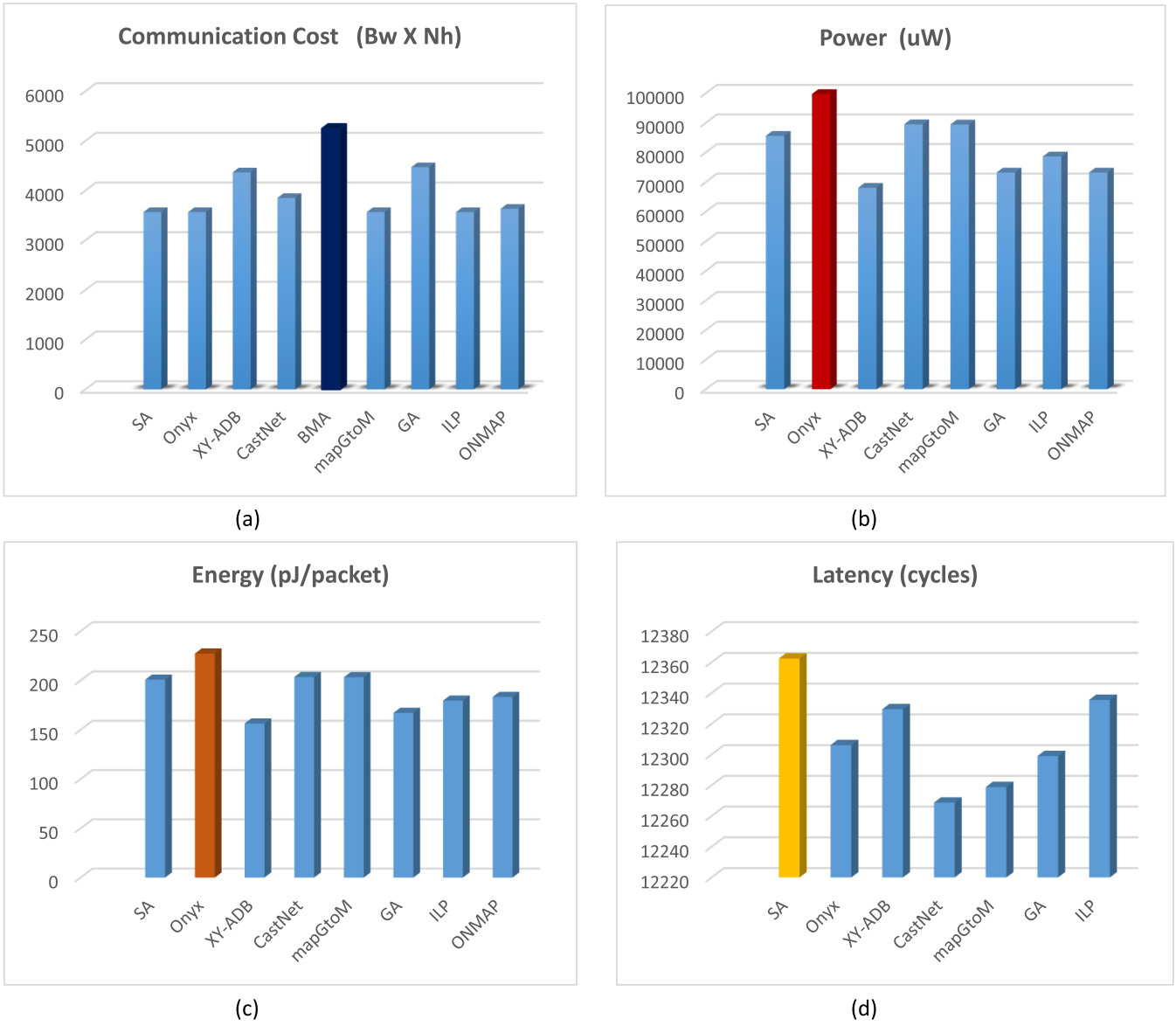


FIGURE 11. Performance factors comparison of different application mapping algorithms on MPEG4 application.

categorized in different mapping techniques based on their genre. The main objective here is to analyze and evaluate the best algorithm in every branch by executing VOPD and MPEG4 benchmarks. The results of different performance constraints such as communication cost, latency, throughput, energy, and power are calculated by following mathematical models.

The communication cost is calculated, using the following equation 1.

$$Cost = \sum_{i,j}^{n} [B_{ti,tj} \times N_h] \tag{1}$$

where,
$B_{ti,tj}$ = arc bandwidth from tile ti to tile tj
N_h = Manhattan distance of the NoC architecture

The Manhattan distance from the source (x_i, y_i) to the destination node (x_j, y_j) of the NoC architecture is given by:

$$N_h = |x_i - x_j| + |y_i - y_j|$$

The average latency Lt_{av} of the network is represented by equation 2:

$$Lt_{av} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{N_i} \sum_{j=1}^{N} Lt_{ij} \tag{2}$$

where,
Lt_{ij} : Latency of packet j,
N: Number of processors in the platform
N_i : Number of packets received by a single processor after i warm up time.

TABLE 7. Result comparison of different application mapping algorithms for VOPD.

Application mapping algoithms	Communication Cost (Bw X Nh)	Power (uW)	Energy (pJ/packet)	Latency (cycles)	Throughput (packets/cycle)
Constructive heuristic with iterative improvement					
NMAP	4265	19733.443	264.701	17.011	0.005
SA	4471	19920.963	267.395	17.456	0.005
Onyx	4577	20059.964	269.986	17.220	0.005
Crinkle	4157	19569.701	262.850	16.924	0.005
LMAP	4189	19622.254	263.740	16.892	0.005
Map_graph	**4119**	**19503.875**	**262.325**	**16.808**	**0.005**
RASMAP	4265	19668.058	263.470	17.012	0.005
XY-ADB	4568	20111.653	269.413	17.333	0.005
Constructive heuristic without iterative improvement					
RMAP	6323	21507.99	301.23	18.921	0.004
CastNet	4135	19547.333	264.332	16.878	0.005
BMA	4135	19564.340	263.848	16.853	0.005
mapGtoM	**4135**	**19506.422**	**261.831**	**16.892**	**0.005**
Transformative heuristic techniques					
GA	4471	19882.835	268.324	17.212	0.005
Deterministic search techniques					
PRBB	9601	25533.360	343.19	21.372	0.005
SBMAP	**4125**	**19509.064**	**262.042**	**16.964**	**0.005**
ILP based exact mapping techniques					
ILP	4119	19559.691	263.608	16.946	0.005
Hybrid mapping techniques					
ONMAP	4217	19726.849	264.968	17.100	0.005

The average throughput Th_{av} of the network is described in equation 3:

$$Th_{av} = \frac{1}{N\,(T_{sim} - T_{wrm})} \sum_{i=1}^{N} \frac{1}{N_i} \tag{3}$$

where,

T_{sim} : Simulation time

T_{wrm} : Warm-up time

The average Power Pw_{av} of the network is presented in equation 4:

$$Pw_{av} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{1}{N_i} \left[\alpha_{i,j} Pw_{act,j} + \left(1 - \alpha_{ij}\right) Pw_{inact,j}\right] \tag{4}$$

where,

$Pw_{act,j}$: Active power

$Pw_{inact,j}$: Inactive power of component j.

α_{ij} : Active measurement of the component j in the router i

Finally the average energy En_p spent by each packet in the network is given in equation 5:

$$En_p = \frac{(T_{sim} - T_{wrm})}{(NN_p)} \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{1}{N_i} \left[\alpha_{i,j}\, Pw_{act,j} + \left(1 - \alpha_{ij}\right) Pw_{inact,j}\right] \tag{5}$$

where, NP is the total number of packets, travelled across the NoC network.

Simulation results of different algorithms when executed with VOPD and MPEG4 benchmarks on the NoCTweek simulator are shown in **Table 7** and **Table 8** respectively. In **Fig. 10** and **Fig. 11**, a graphical comparison of various mapping algorithms performance on VOPD and MPEG4 benchmarks is presented. In Constructive heuristic with iterative improvement branch, the **Map_graph**

TABLE 8. Result comparison of different application mapping algorithms for MPEG4.

Application mapping algoithms	Communication Cost (Bw X Nh)	Power (uW)	Energy (pJ/packet)	Latency (cycles)	Throughput (packets/cycle)
Constructive heuristic with iterative improvement					
NMAP	-	-	-	-	-
SA	3567	85594.979	200.833	12362.432	0.027
Onyx	3567	99733.956	227.340	12306.113	0.027
Crinkle	-	-	-	-	-
LMAP	-	-	-	-	-
Map_graph	-	-	-	-	-
RASMAP	-	-	-	-	-
XY-ADB	4368	68042.740	156.348	12329.447	0.027
Constructive heuristic without iterative improvement					
RMAP	-	-	-	-	-
CastNet	3852	89503.106	203.462	12268.680	0.027
BMA	5246	-	-	-	-
mapGtoM	3567	89423.707	203.282	12278.818	0.027
Transformative heuristic techniques					
GA	4471	73193.323	167.127	12298.962	0.027
Deterministic search techniques					
PRBB	-	-	-	-	-
SBMAP	-	-	-	-	-
ILP based exact mapping techniques					
ILP	3567	78621.849	179.543	12335.515	0.027
Hybrid mapping techniques					
ONMAP	3633	13715.7	330.5	-	0.005

algorithm gave the best cumulative results on VOPD as compare to other algorithms related to the same branch of application mapping. The **mapGtoM** algorithm has better performance than other Constructive heuristics without iterative improvement algorithms. It is evident from the results that **Map_graph** and **Integer Linear Programming (ILP)** methods produce optimal results at the cost of greater execution time because application mapping is Non-Polynomial hard (NP-hard) problem. ILP methods fail to converge in polynomial time for larger applications and should be used only for small size embedded applications. The heuristics and Metaheuristics methods are faster and are good enough but do not guarantee optimal results for real-time embedded applications. Hybrid methods are faster as well as produce a near optimal solution and can be utilized for most of the embedded applications. NoC application mapping is still an open research problem for the research community to develop accurate and dynamic mapping tools and algorithms.

VI. CONCLUSION

This paper studied and evaluated most of the reported application mapping techniques for NoC. The classification of available mapping techniques into different categories is presented based on their proposed mapping algorithms. Dynamic mapping techniques are presented whereas static mapping approaches are discussed in detail and have further been unveiled as exact application mapping and search based mapping schemes. The comparative analysis is carried out for the performance comparison of different static (offline) application mapping techniques by mapping real-time embedded application benchmarks i.e; VOPD and MPEG4. Based on simulated results, it is concluded that the **Map_graph** algorithm gave the best cumulative results on the VOPD application as compare to other algorithms related to Constructive heuristic with iterative improvement algorithms. **mapGtoM** algorithm has better performance than other Constructive heuristics without iterative improvement algorithms. ILP and Branch and Bound BB also achieved the optimum

results, however, for application with large numbers of nodes it is taking too much computation time to generate a solution. Similarly, application mapping techniques based on genetic algorithm and simulated annealing obtain better solutions most of the time but at the cost of higher run-time latencies. Constructive heuristic algorithms are fastest and can be more useful when execution time is crucial for the application mapping of NoC designs.

REFERENCES

[1] U. Y. Ogras, J. Hu, and R. Marculescu, "Key research problems in NoC design: A holistic perspective," in *Proc. 3rd IEEE/ACM/IFIP Int. Conf. Hardw./Softw. Codesign Syst. Synth. (CODES+ISSS)*, 2005, pp. 69–74.
[2] R. Marculescu, U. Y. Ogras, L.-S. Peh, N. E. Jerger, and Y. Hoskote, "Outstanding research problems in NoC design: System, microarchitecture, and circuit perspectives," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 1, pp. 3–21, Jan. 2009.
[3] P. K. Sahu and S. Chattopadhyay, "A survey on application mapping strategies for Network-on-Chip design," *J. Syst. Archit.*, vol. 59, no. 1, pp. 60–76, Jan. 2013.
[4] R. Pop and S. Kumar, "A survey of techniques for mapping and scheduling applications to network on chip systems," *Comput. Eng.*, vol. 4, p. 4, 2005.
[5] A. K. Singh, M. Shafique, A. Kumar, and J. Henkel, "Mapping on multi/many-core systems," in *Proc. 50th Annu. Des. Autom. Conf. (DAC)*, vol. 13, 2013, p. 1.
[6] N. Kadri and M. Koudil, "A survey on fault-tolerant application mapping techniques for network-on-chip," *J. Syst. Archit.*, vol. 92, pp. 39–52, Jan. 2019.
[7] T. Bjerregaard and S. Mahadevan, "A survey of research and practices of network-on-chip," *ACM Comput. Surv.*, vol. 38, no. 1, pp. 71–121, 2006.
[8] R. Marculescu and P. Bogdan, "The chip is the network: Toward a science of network-on-chip design," *Found. Trends Electron. Design Autom.*, vol. 2, no. 4, pp. 371–461, 2007.
[9] S. Kundu and S. Chattopadhyay, *Network-on-Chip: The Next Generation of System-on-Chip Integration*. Boca Raton, FL, USA: CRC Press, 2018.
[10] W. J. Dally and B. Towles, "Proceedings 2001—Design automation conference," in *Proc. Des. Autom. Conf.*, 2001, pp. 684–689.
[11] Y.-K. Kwok and I. Ahmad, "Static scheduling algorithms for allocating directed task graphs to multiprocessors," *ACM Comput. Surveys*, vol. 31, no. 4, pp. 406–471, Dec. 1999.
[12] R. P. Dick, D. L. Rhodes, and W. Wolf, "TGFF: Task graphs for free," in *Proc. Int. Work. Hardw./Softw. Codesign*, 1998, pp. 97–101.
[13] A. H. Liu and R. P. Dick, "Automatic run-time extraction of communication graphs from multithreaded applications," in *Proc. 4th Int. Conf. Hardw. Softw. Codesign Syst. Synth. (CODES+ISSS)*, 2006, pp. 46–51.
[14] Y. Xue and P. Bogdan, "Scalable and realistic benchmark synthesis for efficient NoC performance evaluation: A complex network analysis approach," in *Proc. Int. Conf. Hardw./Softw. Codesign Syst. Synth. (CODES+ISSS)*, 2016, pp. 1–10.
[15] B. Cao, T. Srikanthan, and C. H. Chang, " Efficient reverse converters for four-moduli sets $\{2^n\text{-}1, 2^n, 2^n\text{+}1, 2^{n+1}\text{-}1\}$ and $\{2^n\text{-}1, 2^n, 2^n\text{+}1, 2^{n-1}\text{-}1\}$," *IEE Proc.-Comput. Digit. Techn.*, vol. 152, no. 5, pp. 687–696, Sep. 2005.
[16] J. Hu and R. Marculescu, "Communication and task scheduling of application-specific networks-on-chip," *IEE Proc. Comput. Digit. Tech.*, vol. 152, no. 5, pp. 643–651, 2005.
[17] M. Kim, D. Kim, and G. E. Sobelman, "Adaptive scheduling for CDMA-based networks-on-chip," in *Proc. 3rd Int. IEEE Northeast Work. Circuits Syst. Conf. (NEWCAS)*, Jun. 2005, pp. 357–360.
[18] S. Stuijk, T. Basten, M. Geilen, A. H. Ghamarian, and B. Theelen, "Resource-efficient routing and scheduling of time-constrained streaming communication on networks-on-chip," *J. Syst. Archit.*, vol. 54, nos. 3–4, pp. 411–426, Mar. 2008.
[19] S. Bertozzi, A. Acquaviva, D. Bertozzi, and A. Poggiali, "Supporting task migration in multi-processor systems-on-chip: A feasibility study," in *Proc. Design, Autom. Test Eur. DATE*, vol. 1, Mar. 2006, pp. 1–6.
[20] S. Manolache, P. Eles, and Z. Peng, "Buffer space optimisation with communication synthesis and traffic shaping for NoCs," in *Proc. Design, Autom. Test Eur. DATE*, vol. 1, 2006, pp. 1–6.
[21] J. W. van den Brand, C. Ciordas, K. Goossens, and T. Basten, "Congestion-controlled best-effort communication for networks-on-chip," in *Proc. Design, Autom. Test Eur. Conf. Exhib.*, Apr. 2007, pp. 948–953.
[22] Q. Tian, J. Li, D. Xue, W. Wu, J. Wang, L. Chen, and J. Wang, "A hybrid task scheduling algorithm based on task clustering," *Mobile Netw. Appl.*, pp. 1–10, Jan. 2020.
[23] J. Fang, T. Yu, and Z. Wei, "Improved ant colony algorithm based on task scale in network on chip (NoC) mapping," *Electronics*, vol. 9, no. 1, p. 6, 2020.
[24] G. Chen, F. Li, and M. Kandemir, "Compiler-directed application mapping for NoC based chip multiprocessors," *ACM SIGPLAN Notices*, vol. 42, no. 7, pp. 155–157, Jul. 2007.
[25] E. Carvalho, N. Calazans, and F. Moraes, "Heuristics for dynamic task mapping in NoC-based heterogeneous MPSoCs," in *Proc. 18th IEEE/IFIP Int. Workshop Rapid Syst. Prototyping (RSP)*, May 2007, pp. 34–40.
[26] E. L. D. S. Carvalho, N. L. Calazans, and F. G. Moraes, "Dynamic task mapping for MPSoCs," *IEEE Des. Test. Comput.*, vol. 27, no. 5, pp. 26–35, Sep. 2010.
[27] C.-L. Chou and R. Marculescu, "User-aware dynamic task allocation in networks-on-chip," in *Proc. Design, Autom. Test Eur.*, Mar. 2008, pp. 1232–1237.
[28] C.-L. Chou, U. Y. Ogras, and R. Marculescu, "Energy- and performance-aware incremental mapping for networks on chip with multiple voltage levels," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 10, pp. 1866–1879, Oct. 2008.
[29] E. Carvalho and F. Moraes, "Congestion-aware task mapping in heterogeneous MPSoCs," in *Proc. Int. Symp. Syst.-Chip*, Nov. 2008, pp. 1–4.
[30] M. A. Al Faruque, R. Krist, and J. Henkel, "ADAM: Run-time agent-based distributed application mapping for on-chip communication," in *Proc. 45th Annu. Conf. Design Autom. (DAC)*, 2008, pp. 760–765.
[31] A. Mehran, A. Khademzadeh, and S. Saeidi, "DSM: A heuristic dynamic spiral mapping algorithm for network on chip," *IEICE Electron. Exp.*, vol. 5, no. 13, pp. 464–471, 2008.
[32] A. K. Singh, W. Jigang, A. Prakash, and T. Srikanthan, "Mapping algorithms for NoC-based heterogeneous MPSoC platforms," in *Proc. 12th Euromicro Conf. Digit. Syst. Design, Archit., Methods Tools*, Aug. 2009, pp. 133–140.
[33] A. K. Singh, T. Srikanthan, A. Kumar, and W. Jigang, "Communication-aware heuristics for run-time task mapping on NoC-based MPSoC platforms," *J. Syst. Archit.*, vol. 56, no. 7, pp. 242–255, Jul. 2010.
[34] M. Fattah, M. Ramirez, M. Daneshtalab, P. Liljeberg, and J. Plosila, "CoNA: Dynamic application mapping for congestion reduction in many-core systems," in *Proc. IEEE 30th Int. Conf. Comput. Design (ICCD)*, Sep. 2012, pp. 364–370.
[35] M. Fattah, M. Daneshtalab, P. Liljeberg, and J. Plosila, "Smart hill climbing for agile dynamic mapping in many-core systems," in *Proc. ACM/EDAC/IEEE*, vol. 1, 2013, pp. 1–6.
[36] C.-H. Huang, C.-Y. Chen, and H.-Y. Huang, "Hierarchical and dependency-aware task mapping for NoC-based systems," in *Proc. 11th Int. Workshop Netw. Chip Archit. (NoCArc)*, Oct. 2018, pp. 1–6.
[37] V. Tsoutsouras, I. Anagnostopoulos, D. Masouros, and D. Soudris, "A hierarchical distributed runtime resource management scheme for NoC-based many-cores," *ACM Trans. Embedded Comput. Syst.*, vol. 17, no. 3, pp. 1–26, Jun. 2018.
[38] M.-H. Haghbayan, A. Kanduri, A.-M. Rahmani, P. Liljeberg, A. Jantsch, and H. Tenhunen, "MapPro: Proactive runtime mapping for dynamic workloads by quantifying ripple effect of applications on networks-on-chip," in *Proc. 9th IEEE/ACM Int. Symp. Netw.-Chip (NOCS)*, 2015, pp. 1–8.
[39] S. Kobbe, L. Bauer, D. Lohmann, W. Schröder-Preikschat, and J. Henkel, "DistRM," in *Proc. 7th IEEE/ACM/IFIP Int. Conf. Hardw./Softw. Codesign Syst. Synth. (CODES+ISSS)*, vol. 11, 2011, p. 119.
[40] M. Fattah, A.-M. Rahmani, T. C. Xu, A. Kanduri, P. Liljeberg, J. Plosila, and H. Tenhunen, "Mixed-criticality run-time task mapping for NoC-based many-core systems," in *Proc. 22nd Euromicro Int. Conf. Parallel, Distrib., Network-Based Process.*, Feb. 2014, pp. 458–465.
[41] L.-T. Huang, H. Dong, J.-S. Wang, M. Daneshtalab, and G.-J. Li, "WeNA: Deterministic run-time task mapping for performance improvement in many-core embedded systems," *IEEE Embedded Syst. Lett.*, vol. 7, no. 4, pp. 93–96, Dec. 2015.
[42] H.-L. Chao, S.-Y. Tung, and P.-A. Hsiung, "Dynamic task mapping with congestion speculation for reconfigurable Network-on-Chip," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 10, no. 1, pp. 1–25, Dec. 2016.

[43] M. F. Reza, D. Zhao, and M. Bayoumi, "Dark silicon-power-thermal aware runtime mapping and configuration in heterogeneous many-core NoC," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4.

[44] M. S. Sadeghi, S. B. Sarmadi, and S. Hessabi, "Toward on-chip network security using runtime isolation mapping," *ACM Trans. Archit. Code Optim.*, vol. 16, no. 3, pp. 1–25, Aug. 2019.

[45] Y. Xiao, Y. Xue, S. Nazarian, and P. Bogdan, "A load balancing inspired optimization framework for exascale multicore systems: A complex networks approach," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2017, pp. 217–224.

[46] Y. Xue and P. Bogdan, "User cooperation network coding approach for NoC performance improvement," in *Proc. 9th IEEE/ACM Int. Symp. Netw.-Chip (NOCS)*, 2015, pp. 1–8.

[47] A. Bender, "Task mapping for heterogeneous multiprocessor systems," in *Proc. EURO-DAC Eur. Design Autom. Conf. EURO-VHDL*, 1996, pp. 190–197.

[48] C. E. Rhee, H. Y. Jeong, and S. Ha, "Many-to-many core-switch mapping in 2-D mesh NoC architectures," in *Proc. IEEE Int. Conf. Comput. Design, VLSI Comput. Processors (ICCD)*, Oct. 2004, pp. 438–443.

[49] S. Murali, L. Benini, and G. De Micheli, "Mapping and physical planning of networks-on-chip architectures with Quality-of-Service guarantees," in *Proc. Asia South Pacific Des. Autom. Conf. (ASP-DAC)*, vol. 1, 2005, pp. 27–32.

[50] K. Srinivasan, K. S. Chatha, and G. Konjevod, "Linear programming based techniques for synthesis of network-on-chip architectures," in *Proc. IEEE Int. Conf. Comput. Design, VLSI Comput. Processors (ICCD)*, 2004, p. 25.

[51] C. Ostler and K. S. Chatha, "An ILP formulation for system-level application mapping on network processor architectures," in *Proc. Design, Autom. Test Eur. Conf. Exhib.*, Apr. 2007, pp. 99–104.

[52] O. Ozturk, M. Kandemir, and S. W. Son, "An ilp based approach to reducing energy consumption in nocbased CMPS," in *Proc. Int. Symp. Low power Electron. Design (ISLPED)*, 2007, pp. 411–414.

[53] P. Ghosh, A. Sen, and A. Hall, "Energy efficient application mapping to NoC processing elements operating at multiple voltage levels," in *Proc. 3rd ACM/IEEE Int. Symp. Netw.-Chip*, 2009, pp. 80–85.

[54] J. Huang, C. Buckl, A. Raabe, and A. Knoll, "Energy-aware task allocation for Network-on-Chip based heterogeneous multiprocessor systems," in *Proc. 19th Int. Euromicro Conf. Parallel, Distrib. Netw.-Based Process.*, Feb. 2011, pp. 447–454.

[55] C.-L. Chou and R. Marculescu, "Contention-aware application mapping for Network-on-Chip communication architectures," in *Proc. IEEE Int. Conf. Comput. Design*, Oct. 2008, pp. 164–169.

[56] S. Tosun, O. Ozturk, and M. Ozen, "An ILP formulation for application mapping onto Network-on-Chips," in *Proc. Int. Conf. Appl. Inf. Commun. Technol.*, Oct. 2009, pp. 1–5.

[57] S. Tosun, "Cluster-based application mapping method for Network-on-Chip," *Adv. Eng. Softw.*, vol. 42, no. 10, pp. 868–874, Oct. 2011.

[58] A. Aravindhan, S. Salini, and G. Lakshminarayanan, "Cluster based application mapping strategy for 2D NoC," *Procedia Technol.*, vol. 25, pp. 505–512, 2016.

[59] S. Tosun, O. Ozturk, E. Ozkan, and M. Ozen, "Application mapping algorithms for mesh-based network-on-chip architectures," *J. Supercomput.*, vol. 71, no. 3, pp. 995–1017, Mar. 2015.

[60] P. Bogdan, "A cyber-physical systems approach to personalized medicine: Challenges and opportunities for noc-based multicore platforms," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2015, pp. 253–258.

[61] J. Hu and R. Marculescu, "Energy-aware mapping for tile-based NoC architectures under performance constraints," in *Proc. ASP-DAC Asia South Pacific Design Autom. Conf.*, Jan. 2003, pp. 233–239.

[62] J. Hu and R. Marculescu, "Exploiting the routing flexibility for energy/performance-aware mapping of regular NoC architectures," *Des. Autom. Test Eur. Most Influ. Papers 10 Years Date*, 2008, pp. 141–155.

[63] T.-J. Lin, S.-Y. Lin, and A.-Y. Wu, "Traffic-balanced IP mapping algorithm for 2D-mesh on-chip-networks," in *Proc. IEEE Workshop Signal Process. Syst.*, vol. 1, Oct. 2008, pp. 200–203.

[64] M. Reshadi, A. Khademzadeh, and A. Reza, "Elixir: A new bandwidth-constrained mapping for networks-on-chip," *IEICE Electron. Exp.*, vol. 7, no. 2, pp. 73–79, 2010.

[65] S. Khan, S. Anjum, U. A. Gulzari, T. Umer, and B.-S. Kim, "Bandwidth-constrained multi-objective segmented brute-force algorithm for efficient mapping of embedded applications on NoC architecture," *IEEE Access*, vol. 6, pp. 11242–11254, 2018.

[66] B. Chopard and M. Tomassini, "Particle swarm optimization," in *An Introduction to Metaheuristics for Optimization*. Cham, Switzerland: Springer, 2018, pp. 97–102.

[67] A. Colorni, M. Dorigo, and V. Maniezzo, "Distributed Optimization by ant colonies," *Proc. 1st Eur. Conf. Artif. Life*, Jun. 1991, pp. 134–142.

[68] T. Lei and S. Kumar, "A two-step genetic algorithm for mapping task graphs to a network on chip architecture," in *Proc. Euromicro Symp. Digit. Syst. Design*, 2003, pp. 180–187.

[69] W. Zhou, Y. Zhang, and Z. Mao, "An application specific NoC mapping for optimized delay," in *Proc. Int. Conf. Design Test Integr. Syst. Nanosc. Technol. (DTIS)*, 2006, pp. 184–188.

[70] F. Moein-darbari, A. Khademzade, and G. Gharooni-fard, "CGMAP: A new approach to Network-on-Chip mapping problem," *IEICE Electron. Exp.*, vol. 6, no. 1, pp. 27–34, 2009.

[71] F. Ge and N. Wu, "Genetic algorithm based mapping and routing approach for Network on chip architectures," *Chin. J. Electron.*, vol. 19, no. 1, pp. 91–96, 2010.

[72] W. Jang and D. Z. Pan, "A3MAP," *ACM Trans. Design Autom. Electron. Syst.*, vol. 17, no. 3, pp. 1–22, Jun. 2012.

[73] A. Benyamina, P. Boulet, A. Aroui, S. Eltar, and K. Dellal, "Mapping real time applications on NoC architecture with hybrid multi-objective algorithm," in *Proc. Int. Conf. Metaheuristics Nat. Inspired, Comput.*, 2010, pp. 1–10.

[74] P. K. Sahu, P. Venkatesh, S. Gollapalli, and S. Chattopadhyay, "Application mapping onto mesh structured network-on-chip using particle swarm optimization," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI*, Jul. 2011, pp. 335–336.

[75] A. R. Fekr, A. Khademzadeh, M. Janidarmian, and V. S. Bokharaei, "Bandwidth/fault tolerance/contention aware application-specific NoC using PSO as a mapping generator," in *Proc. World Congr. Eng. (WCE)*, vol. 1, 2010, pp. 247–252.

[76] A. Roy, K. Manna, and S. Chattapadhay, "Effect of core ordering on application mapping onto mesh based network-on-chip design," in *Proc. Int. Conf. Comput. Sustain. Glob. Dev. (INDIACom)*, 2015, pp. 363–369.

[77] P. K. Sahu, A. Sharma, and S. Chattopadhyay, "Application mapping onto mesh-of-tree based network-on-chip using discrete particle swarm optimization," in *Proc. Int. Symp. Electron. Syst. Design (ISED)*, Dec. 2012, pp. 172–176. vol. 115, no. 19, pp. 172-176, 2012

[78] M. Obaidullah and G. N. Khan, "Application mapping to mesh NoCs using a tabu-search based swarm optimization," *Microprocessors Microsyst.*, vol. 55, pp. 13–25, Nov. 2017.

[79] N. Chatterjee, P. Mukherjee, and S. Chattopadhyay, "Reliability-aware application mapping onto mesh based Network-on-Chip," *Integration*, vol. 62, Dec. 2017, pp. 92–113, 2018.

[80] X. Wang, Y. Sun, and H. Gu, "BMM: A binary metaheuristic mapping algorithm for mesh-based network-on-chip," *IEICE Trans. Inf. Syst.*, vol. E102.D, no. 3, pp. 628–631, Mar. 2019.

[81] M. Upadhyay, M. Shah, P. V. Bhanu, J. Soumya, and L. R. Cenkeramaddi, "Multi-application based network-on-chip design for mesh-of-tree topology using global mapping and reconfigurable architecture," in *Proc. 32nd Int. Conf. VLSI Design 18th Int. Conf. Embedded Syst. (VLSID)*, Jan. 2019, pp. 527–528.

[82] J. Wang, Y. Li, S. Chai, and Q. Peng, "Bandwidth-aware application mapping for NoC-based MPSoCs," *J. Comput. Inf. Syst.*, vol. 7, no. 1, pp. 152–159, 2011.

[83] Y. Liu, Y. Ruan, Z. Lai, and W. Jing, "Energy and thermal aware mapping for mesh-based NoC architectures using multi-objective ant colony algorithm," in *Proc. 3rd Int. Conf. Comput. Res. Develop.*, Mar. 2011, pp. 407–411.

[84] Y. Xue, Z. Qian, G. Wei, P. Bogdan, C.-Y. Tsui, and R. Marculescu, "An efficient network-on-chip (NoC) based multicore platform for hierarchical parallel genetic algorithms," in *Proc. 8th IEEE/ACM Int. Symp. Netw.-Chip (NoCS)*, Sep. 2014, pp. 17–24.

[85] S. Murali and G. De Micheli, "Bandwidth-constrained mapping of cores onto NoC architectures," in *Proc. Des. Autom. Test Eur. Conf. Exhib.*, vol. 2, 2004, pp. 896–901.

[86] K. Srinivasan and K. S. Chatha, "A technique for low energy mapping and routing in network-on-chip architectures," in *Proc. Int. Symp. Low Power Electron. Design (ISLPED)*, 2005, pp. 387–392.

[87] Z. Lu, L. Xia, and A. Jantsch, "Cluster-based simulated annealing for mapping cores onto 2D mesh networks on chip," in *Proc. 11th IEEE Workshop Design Diag. Electron. Circuits Syst.*, Apr. 2008, pp. 92–97.

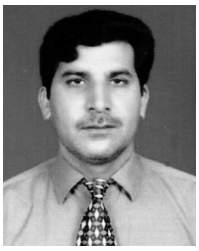
[88] M. Janidarmian, A. Khademzadeh, and M. Tavanpour, "Onyx: A new heuristic bandwidth-constrained mapping of cores onto tile-based network on chip," *IEICE Electron. Exp.*, vol. 6, no. 1, pp. 1–7, 2009.
[89] R. Seidipiri, A. Patooghy, S. Afsharpour, and M. Fazeli, "RASMAP: An efficient heuristic application mapping algorithm for network-on-chips," in *Proc. 8th Int. Conf. Inf. Knowl. Technol. (IKT)*, Sep. 2016, pp. 149–155.
[90] S. Saeidi, A. Khademzadeh, and F. Vardi, "Crinkle: A heuristic mapping algorithm for network on chip," *IEICE Electron. Exp.*, vol. 6, no. 24, pp. 1737–1744, 2009.
[91] P. K. Sahu, N. Shah, K. Manna, and S. Chattopadhyay, "A new application mapping algorithm for mesh based Network-on-Chip design," in *Proc. Annu. IEEE India Conf. (INDICON)*, Dec. 2010, pp. 1–4.
[92] P. K. Sahu, K. Manna, T. Shah, and S. Chattopadhyay, "A constructive heuristic for application mapping onto mesh based network-on-chip," *J. Circuits, Syst. Comput.*, vol. 24, no. 8, Sep. 2015, Art. no. 1550126.
[93] E. Alikhah-Asl and M. Reshadi, "XY-axis and distance based NoC mapping (XY-ADB)," in *Proc. 8th Int. Symp. Telecommun. (IST)*, Sep. 2016, pp. 678–683.
[94] N. Koziris, M. Romesis, P. Tsanakas, and G. Papakonstantinou, "An efficient algorithm for the physical mapping of clustered task graphs onto multiprocessor architectures," in *Proc. 8th Euromicro Workshop Parallel Distrib. Process.*, 2003, pp. 406–413.
[95] A. Hansson, K. Goossens, and A. Rădulescu, "A unified approach to constrained mapping and routing on network-on-chip architectures," in *Proc. 3rd IEEE/ACM/IFIP Int. Conf. Hardw./Softw. Codes. Syst. Synth.*, Jersey City, NJ, USA, Sep. 2005, pp. 75–80.
[96] S. Saeidi, A. Khademzadeh, and A. Mehran, "SMAP: An intelligent mapping tool for network on chip," in *Proc. Int. Symp. Signals, Circuits Syst.*, vol. 1, Jul. 2007, pp. 169–172.
[97] W.-T. Shen, C.-H. Chao, Y.-K. Lien, and A.-Y. Wu, "A new binomial mapping and optimization algorithm for reduced-complexity mesh-based on-chip network," in *Proc. 1st Int. Symp. Netw.-Chip (NOCS)*, May 2007, pp. 317–322.
[98] A. Patooghy, H. Tabkhi, and S. G. Miremadi, "RMAP: A reliability-aware application mapping for network-on-chips," in *Proc. 3rd Int. Conf. Dependability*, Jul. 2010, pp. 112–117.
[99] S. Tosun, "New heuristic algorithms for energy aware application mapping and routing on mesh-based NoCs," *J. Syst. Archit.*, vol. 57, no. 1, pp. 69–78, Jan. 2011.
[100] O. Al-Harbi, "Classifying sentiment of dialectal arabic reviews: A semi-supervised approach," *Int. Arab J. Inf. Technol.*, vol. 16, no. 6, pp. 995–1002, Nov. 2019.
[101] C.-H. Cheng and W.-M. Chen, "Application mapping onto mesh-based network-on-chip using constructive heuristic algorithms," *J. Supercomput.*, vol. 72, no. 11, pp. 4365–4378, Nov. 2016.
[102] P. K. Sharma, S. Biswas, and P. Mitra, "Energy efficient heuristic application mapping for 2-D mesh-based network-on-chip," *Microprocessors Microsyst.*, vol. 64, pp. 88–100, Feb. 2019.
[103] A. Weichslgartner, D. Gangadharan, S. Wildermann, M. Glaß, and J. Teich, "DAARM: Design-time application analysis and run-time mapping for predictable execution in many-core systems," in *Proc. Int. Conf. Hardw./Softw. Codesign Syst. Synth. CODES+ISSS*, vol. 2014, 2014.
[104] S. Sinaei, A. D. Pimentel, and O. Fatemi, "Run-time resource allocation for embedded multiprocessor system-on-chip using tree-based design space exploration," in *Proc. 12th IEEE Int. Conf. Des. Technol. Integr. Syst. Nanosc. Era (DTIS)*, Apr. 2017, pp. 1–6.
[105] T. Schwarzer, S. Roloff, V. Richthammer, R. Khaldi, S. Wildermann, M. Glass, and J. Teich, "On the complexity of mapping feasibility in many-core architectures," in *Proc. IEEE 12th Int. Symp. Embedded Multicore/Many-core Syst.-Chip (MCSoC)*, Sep. 2018, pp. 176–183.
[106] A. Abdi and H. R. Zarandi, "HYSTERY: A hybrid scheduling and mapping approach to optimize temperature, energy consumption and lifetime reliability of heterogeneous multiprocessor systems," *J. Supercomput.*, vol. 74, no. 5, pp. 2213–2238, May 2018.
[107] G. Du, G. Liu, Z. Li, Y. Cao, and D. Zhang, "SSS: Self-aware system-on-chip using a static-dynamic hybrid method," *J. Emerg. Technol. Comput. Syst.* vol. 15, no. 3, pp. 1–26, 2019.
[108] S. Khan, S. Anjum, U. A. Gulzari, F. Ishmanov, M. Palesi, and M. K. Afzal, "An optimized hybrid algorithm in term of energy and performance for mapping real time workloads on 2D based on-chip networks," *Int. J. Speech Technol.*, vol. 48, no. 12, pp. 4792–4804, Dec. 2018.
[109] S. Khan, S. Anjum, U. A. Gulzari, M. K. Afzal, T. Umer, and F. Ishmanov, "An efficient algorithm for mapping real time embedded applications on NoC architecture," *IEEE Access*, vol. 6, pp. 16324–16335, 2018.
[110] Y. Xiao, S. Nazarian, and P. Bogdan, "Self-optimizing and self-programming computing systems: A combined compiler, complex networks, and machine learning approach," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 6, pp. 1416–1427, Jun. 2019.
[111] Z. Qian, D.-C. Juan, P. Bogdan, C.-Y. Tsui, D. Marculescu, and R. Marculescu, "SVR-NoC: A performance analysis tool for network-on-chips using learning-based support vector regression model," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, 2013, pp. 354–357.
[112] M. F. Reza, T. T. Le, B. De, M. Bayoumi, and D. Zhao, "Neuro-NoC: Energy optimization in heterogeneous many-core NoC using neural networks in dark silicon era," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2018, pp. 1–5.
[113] P. Ampadu, "Energy-efficient and high-performance NoC architecture and mapping solution for deep neural networks," in *Proc. IEEE/ACM*, 2019, pp. 1–9.
[114] Y. Xiang and J. Meng, "A cross-layer based mapping for spiking neural network onto network on chip," *Int. J. Parallel, Emergent Distrib. Syst.*, vol. 33, no. 5, pp. 526–544, Sep. 2018.
[115] K.-C. J. Chen, M. Ebrahimi, T.-Y. Wang, and Y.-C. Yang, "NoC-based DNN accelerator: A future design paradigm," presented at the Int. Symp. Netw.-Chip (NOCS), 2019.
[116] X. Liu, W. Wen, X. Qian, H. Li, Y. Chen, "Neu-NoC: A high-efficient interconnection network for accelerated neuromorphic systems," in *Proc. 23rd Asia South Pacific Design Automat. Conf. (ASP-DAC)*, 2018, pp. 141–146.
[117] K.-C. Chen and T.-Y. Wang, "NN-noxim: High-level cycle-accurate NoC-based neural networks simulator," in *Proc. 11th Int. Workshop Netw. Chip Archit. (NoCArc)*, Oct. 2018, pp. 1–5.
[118] K.-C.-J. Chen, T.-Y.-G. Wang, and Y.-C.-A. Yang, "Cycle-accurate NoC-based convolutional neural network simulator," in *Proc. Int. Conf. Omni-Layer Intell. Syst. (COINS)*, 2019, pp. 199–204.
[119] M. O. Agyeman, A. Ahmadinia, and N. Bagherzadeh, "Energy and performance-aware application mapping for inhomogeneous 3D networks-on-chip," *J. Syst. Archit.*, vol. 89, pp. 103–117, Sep. 2018.
[120] B. Li, X. Wang, A. K. Singh, and T. Mak, "On runtime communication and thermal-aware application mapping and defragmentation in 3D NoC systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 12, pp. 2775–2789, Dec. 2019.
[121] A. Alagarsamy, L. Gopalakrishnan, and S.-B. Ko, "KBMA: A knowledge-based multi-objective application mapping approach for 3D NoC," *IET Comput. Digit. Techn.*, vol. 13, no. 4, pp. 324–334, Jul. 2019.
[122] K. Manna and J. Mathew, *Design and Test Strategies for 2D/3D Integration for NoC-based Multicore Architectures*. Springer, 2020.
[123] C. Constantinescu, "Trends and challenges in VLSI circuit reliability," *IEEE Micro*, vol. 23, no. 4, pp. 14–19, Jul./Aug. 2003.
[124] P. P. Pande, C. Grecu, A. Ivanov, R. Saleh, and G. De Micheli, "Design, synthesis, and test of networks on chips," *IEEE Design Test Comput.*, vol. 22, no. 5, pp. 404–413, May 2005.
[125] Y. He and G. Chen, "An inclusive fault model for Network-on-Chip," in *Proc. IEEE 11th Int. Conf. ASIC (ASICON)*, Nov. 2015, pp. 5–8.
[126] B. Aghaei, A. Khademzadeh, M. Reshadi, and K. Badie, "Link testing: A survey of current trends in network on chip," *J. Electron. Test.*, vol. 33, no. 2, pp. 209–225, Apr. 2017.
[127] M. Radetzki, C. Feng, X. Zhao, and A. Jantsch, "Methods for fault tolerance in networks-on-chip," *ACM Comput. Surveys*, vol. 46, no. 1, pp. 1–38, Oct. 2013.
[128] N. K. Baloch, M. I. Baig, and M. Daneshtalab, "Defender: A low overhead and efficient fault-tolerant mechanism for reliable on-chip router," *IEEE Access*, vol. 7, pp. 142843–142854, 2019.
[129] M. Ibrahim, N. Khan, Y. B. Zikria, and S. W. Kim, "An energy efficient and low overhead fault mitigation technique for Internet of Thing edge devices reliable on-chip communication," *Softw., Pract. Exper.*, pp. 1–18, Jan. 2020.
[130] W. Quan and A. D. Pimentel, "A system-level simulation framework for evaluating task migration in MPSoCs," in *Proc. Int. Conf. Compil. Archit. Synth. Embed. Syst.*, 2014, pp. 1–9.
[131] C. Yang and A. Orailoglu, "Predictable execution adaptivity through embedding dynamic reconfigurability into static MPSoC schedules," in *Proc. 5th IEEE/ACM Int. Conf. Hardw./Softw. Codesign Syst. Synth. (CODES+ISSS)*, Sep. 2007, pp. 15–20.

[132] A. Tran and B. Baas, "Noctweak: A highly parameterizable simulator for early exploration of performance and energy of networks on-chip," VLSI Comput. Lab, Univ. California, Oakland, CA, USA, Jul. 2012, pp. 1–12.
[133] H.-S. Wang, X. Zhu, L.-S. Peh, and S. Malik, "Orion: A power-performance simulator for interconnection networks," in *Proc. 35th Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Jan. 2002, pp. 294–305.
[134] A. B. Kahng, B. Li, L.-S. Peh, and K. Samadi, "ORION 2.0: A fast and accurate NoC power and area model for early-stage design space exploration," in *Proc. Design, Autom. Test Eur. Conf. Exhib.*, Apr. 2009, pp. 423–428.

WAQAR AMIN received the B.Sc. degree in computer engineering and the M.S. degree from the University of Engineering and Technology at Taxila (UET Taxila), Pakistan, in 2007 and 2014, respectively, where he is currently pursuing the Ph.D. degree. He has vast experience of research and development in various embedded systems companies. He worked on the development of many GSM and 3G systems. His research interests include fault-tolerant systems, Network on Chip (NoC), self-healing, and reconfigurable systems designs. He is currently working on the low-cost application mapping on NoC.

FAWAD HUSSAIN received the B.Sc. degree in computer engineering, the M.Sc. degree in electrical engineering, and the Ph.D. degree in computer engineering from the University of Engineering and Technology (UET), Taxila, Pakistan, in 2005, 2009, and 2015, respectively. He is currently working as an Assistant Professor with the Computer Engineering Department, UET. His research interest includes speech and audio processing, computer vision, human activity recognition, and emotion recognition. He is currently leading research for the M.S. and Ph.D. students in the mentioned areas of interest.

SHERAZ ANJUM received the M.Sc. degree in electronics from Quaid-E-Azam University, Islamabad, Pakistan, in 1999, the M.Sc. degree in computer engineering from the University of Engineering and Technology, Taxila, Pakistan, in 2005, and the Ph.D. degree in microelectronics and solid-state electronics from the Institute of Microelectronics, Graduate University of Chinese Academy of Sciences, Beijing, China, in 2008. From 1999 to 2001, he worked for the Barani Institute of Information Technology as a Research Associate. From 2001 to 2005 and from 2008 to 2012, he served at the COMSATS Institute of Information Technology, Wah Cantt Campus, Pakistan, as a Lecturer and as an Assistant Professor, respectively. He is currently working as an Associate Professor with the COMSATS University Islamabad, Wah Cantt Campus. He has more than 20 years of teaching and research experience. His research interests include but not limited to design and analysis of networks on chip architectures and algorithms, multiprocessor heterogeneous computing, reconfigurable architectures, and advance computer architectures. From 2010 to 2018, he served in the TPC of the international conference FIT. He is serving as a reviewer of many international journals. Since 2015, he is a member of IET and an executive member of IET Islamabad local network. He is registered as a Chartered Engineer with ECUK and as a Registered Engineer with Pakistan Engineering Council.



SARZAMIN KHAN received the B.E. degree in electronic engineering and the M.S. degree in electrical engineering from the NED University of Engineering and Technology, Karachi, Pakistan, in 2001 and 2003, respectively, and the Ph.D. degree in electrical engineering from COMSATS University Islamabad, Pakistan, in 2018. He has more than ten years of industrial experience at different positions. His research interests include design and analysis of analog and digital systems, artificial intelligence, the IoT, communication systems, System-on-Chip, Network-on-Chip, embedded systems, and optimization techniques. He is serving as a Reviewer and Author of *Future Generation Computer Systems* (Elsevier), IEEE Access, *PLOS One*, *Journal of Artificial Intelligence* (Springer), and *Journal of IET Computers and Digital Techniques*.

NAVEED KHAN BALOCH received the B.Sc. degree in computer engineering and the M.S. degree from the University of Engineering and Technology, Taxila, Pakistan, in 2007, where he is currently pursuing the Ph.D. degree. He has worked in multinational companies as an Embedded System Designer, from 2007 to 2010. He joined UET as a Lecturer. He has published many research articles in his field and have experience in embedded system designing, fault tolerant systems, reconfigurable computing, and he is currently working on self-healing digital systems. He is also working as an Assistant Professor with the Computer Engineering Department, UET Taxila. During his tenure in the academia, he did many collaborations with industry and foreign universities in the field of on-chip networks, embedded vision, and reconfigurable computing.

ZULQAR NAIN (Student Member, IEEE) received the B.S. degree in computer science from the Virtual University of Pakistan, in 2015, and the M.S. degree from COMSATS University Islamabad, in 2018. He is currently pursuing the Ph.D. degree with the WINLab, Department of Information and Communication Engineering, Yeungnam University, South Korea. His research interests include routing in NoC, fault-tolerant routing in NoC, the IoT, machine learning, and wireless NoCs.

SUNG WON KIM received the B.S. and M.S. degrees from the Department of Control and Instrumentation Engineering, Seoul National University, South Korea, in 1990 and 1992, respectively, and the Ph.D. degree from the School of Electrical Engineering and Computer Sciences, Seoul National University, in August 2002. From January 1992 to August 2001, he was a Researcher with the Research and Development Center, LG Electronics, South Korea. From August 2001 to August 2003, he was a Researcher with the Research and Development Center, AL Tech, South Korea. From August 2003 to February 2005, he was a Postdoctoral Researcher with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, USA. In March 2005, he joined the Department of Information and Communication Engineering, Yeungnam University, Gyeongsangbuk-do, South Korea, where he is currently a Professor. His research interests include resource management, wireless networks, mobile computing, performance evaluation, and machine learning.