

The Chip Is the Network: Toward a Science of Network-on-Chip Design

Radu Marculescu¹ and Paul Bogdan²

¹ *Department of Electrical and Computer Engineering, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA, radum@ece.cmu.edu*

² *Department of Electrical and Computer Engineering, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA, pbogdan@ece.cmu.edu*

Abstract

In this survey, we address the concept of network in three different contexts representing the deterministic, probabilistic, and statistical physics-inspired design paradigms. More precisely, we start by considering the natural representation of networks as graphs and discuss the main deterministic approaches to Network-on-Chip (NoC) design. Next, we introduce a probabilistic framework for network representation and optimization and present a few major approaches for NoC design proposed to date. Last but not least, we model the network as a thermodynamic system and discuss a statistical physics-based approach to characterize the network traffic. This formalism allows us to address the network concept in the most general context, point out the main limitations of the proposed solutions, and suggest a few open-ended problems.

1

Introduction

Living in a world where the concept of *network* is ubiquitous, makes the quest for a science of network design inevitable. Informally speaking, the science of network design seeks to discover and explain the main properties affecting the network structure and behavior, as well as the mathematical models for predicting their dynamics and guiding their implementation.

The science of networks is primarily based on the graph theory, one of the most successful developments in mathematics ever. Indeed, the graph theory has led to an incredible number of real life applications that continue to grow even today. The graph theory originates in 1736 when Leonhard Euler offered the first rigorous solution to the “Seven Bridges of Konigsberg” problem. More precisely, by abstracting each landmass with a node (or vertex) and each bridge with an edge (link) (see Figure 1.1), Euler was able to show that a path that crosses all seven bridges in the city of Konigsberg only once cannot exist [18]. The newly proposed modeling paradigm based on sets of points linked by edges and represented via adjacency matrices (i.e., a matrix of zeros and ones, where each row in the matrix represents the connectivity of a node) has found many real-world applications. For instance, Cayley

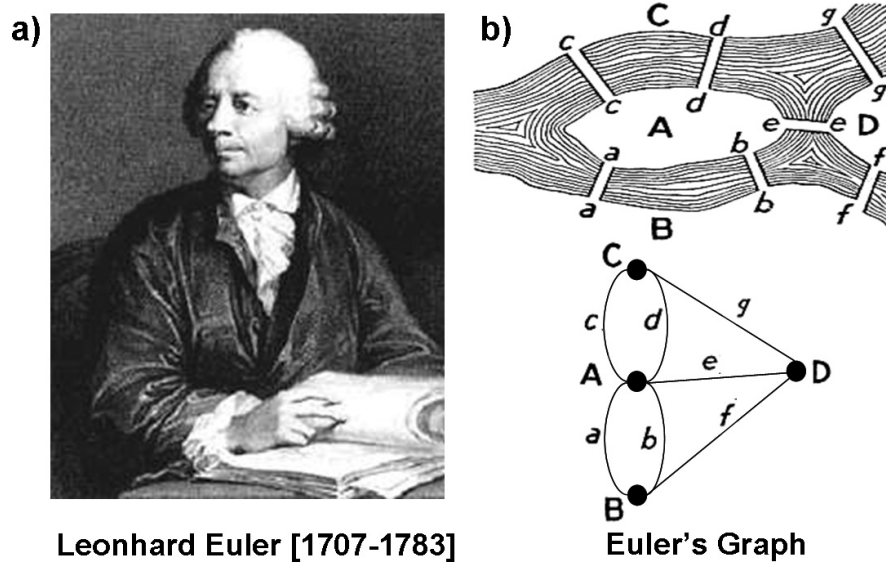


Fig. 1.1 Deterministic perspective on networks: (a) Leonhard Euler invented graph theory while trying to show that a path crossing all bridges in Königsberg does not exist. (b) Abstract representation of the “Seven Bridges of Königsberg” problem: river sides are depicted as nodes and bridges as arcs.

graphs have been applied in the field of chemistry for studying various compound molecules [14]. Other applications of static graphs include the graph coloring with applications to register allocation, scheduling and compiler design; construction of trees with applications in designing the railway system; the problem of finding the optimal cycle in a graph (i.e., the traveling salesman problem) with application to VLSI design; computation of voltages and currents in electric circuits via Kirchhoff’s law [38].

Almost two centuries after the “Seven Bridges of Königsberg” problem was solved, a new discovery in the graph theory, namely the *random graphs* proposed by Erdos and Renyi [68, 69], revolutionized the way we perceive real systems (e.g., rail, road, airplane, electronic networks, etc.). Random graphs are similar to regular graphs with the distinction that any edge between two arbitrary nodes is established with a certain probability. Consequently, the corresponding entry in the adjacency matrix is the probability $p \in [0, 1]$ that two nodes are connected

(instead of a fixed 1 value which would correspond to the classical graph model). This way, a new bridge is created between graph and probability theories [35]; this introduces the possibility of naturally modeling social communication, collaboration or biological networks [62, 140].

Understanding the network behavior requires a deep analysis of the topology and pattern of communication among network components. Thus, a breakthrough in the field of graph theory took place in the 1990s, when physicists questioned the evolution and structure of real networks [8, 71, 78]. The fundamental contribution of this new body of work is the replacement of the static view of random graphs with a dynamic view based on probability distributions for node connectivity in the hope of finding the optimal design of any communication network.

In recent years, the level of understanding of networking concepts needed to design and control complex systems, has reached unprecedented peaks. As such, a holistic approach to the network paradigm is essential. Such an approach involves understanding the theoretical basis (e.g., graph theory, stochastic modeling and analysis), the essential properties (e.g., structure, dynamics, communication paradigm), and the metrics (e.g., energy, fault-tolerance, robustness) which are relevant to designing and characterizing different networks in either engineered or biological systems.

In order to put the network concept into the proper perspective, we need to step back and consider Milner's fundamental work on calculus of communicating systems (also referred to as process algebra) [132], which enables the compact description of communication actions between any two agents. Inspired by the synergy between concurrency and communication, the design of electronic systems has moved recently from computation-based design to communication-based design. Indeed, nowadays, Systems-on-Chips (SoCs) represent true distributed systems at nanoscale where communication aspects dominate. From a technology point of view, this paradigm shift is intended to mitigate the problem of interconnects, keep the design complexity under control, and reduce costs. Since none of the classical architectures based on point-to-point or bus-based communication scales

nically in terms of power and performance figures, a networked architecture based on packet switching has been suggested for future multicore systems [58].

Starting from these overarching ideas, in this survey we address the concept of network in Multiprocessor Systems-on-Chip (MPSoCs) and identify specific design principles and optimization techniques that are relevant to the design automation research community. Understanding the structure and behavior of seemingly different networks is crucial for our ability to master complex behaviors that characterize the emergent application domains. For instance, for SoCs, it has been suggested to replace the global interconnects with packet switching communication via the Network-on-Chip (NoC) paradigm [22, 30, 87, 88, 102] and then avoid the interconnect performance issues [91, 115], while allowing for higher level of fault-tolerance in Deep Submicron (DSM) technologies [24, 167]. Several concrete NoC architectures have been investigated in the literature [61, 116, 198, 201], as well as energy-efficient Globally Asynchronous Locally Synchronous (GALS) designs [20, 21, 42, 181, 212].

The emergent NoC communication paradigm consists of exchanging packets of information among various nodes in the network. As shown in Figure 1.2, each node consists of a core (e.g., DSP or CPU modules, video processors, embedded memory blocks or application specific Processing Elements (PEs)) and a router meant to forward the incoming packets toward the appropriate destination according to the header information. To support the inter-tile communication, each core has embedded input and output buffers to temporarily store the incoming packets from the neighboring nodes in the network. For instance, a packet generated at source (1,1) that needs to be delivered to destination (2,3) via a static XY routing, is first sent from the local PE to its associated router at tile (1,1); then, at each intermediate node, a routing decision is made based on the header information as shown with the dotted arrows in Figure 1.2 for the shortest source–destination path. To avoid the stalling of information flow due to the dependencies on network resources (i.e., shared routing paths), the concept of virtual channels (VCs) has been introduced as well [57, 114, 133, 141]. VCs share dedicated links and provide multiple buffers for each channel.

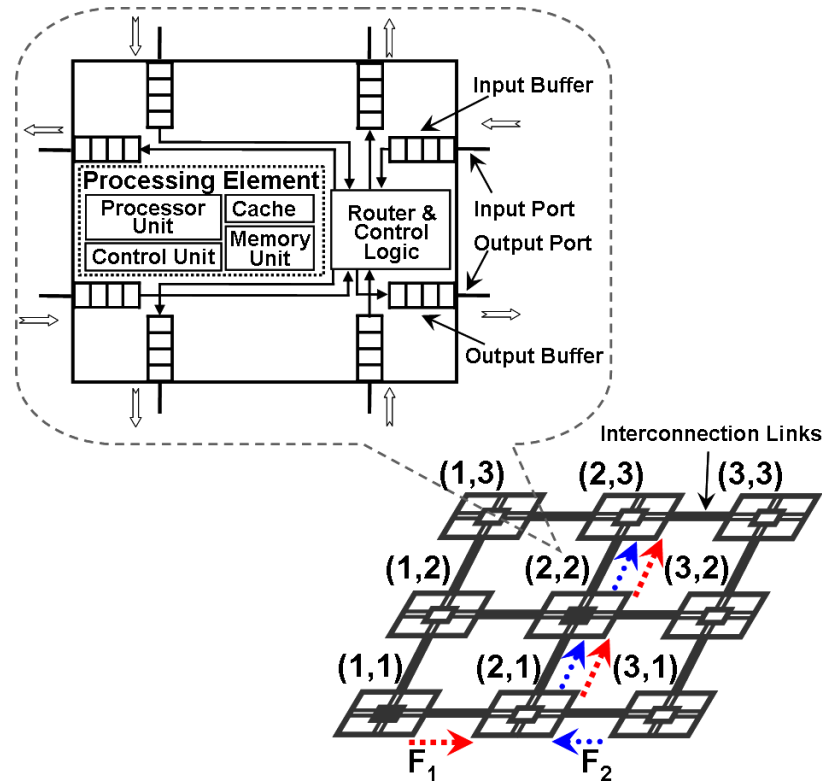


Fig. 1.2 A 3×3 mesh Network-on-Chip (NoC) architecture consisting of Processing Elements (PEs), routers, and interconnection links. Flows F_1 and F_2 are used to depict a stalling problem along the path from node (2,1) to (2,3) which can be solved by utilizing virtual channels.

For instance, the problem of stalling the flow F_1 from (1,1) to (2,3) due to a flow F_2 from (3,1) to (2,3), is solved by reserving the VCs between nodes (2,1), (2,2), and (2,3) for F_1 .

A natural question now is what are the fundamental mathematical techniques that can be used to design, control, and optimize such networks in a rigorous manner. One such powerful technique is the *linear programming* (LP) approach used to solve maximum flow problems [54]. The goal of the linear program is to find a legal flow assignment of the edges of a given graph satisfying the flow conservation constraints. Another mathematical programming approach, namely the *quadratic programming* (QP) was proposed for solving the max-cut

problem [54, 86]. Due to the requirement of discrete values, many real world optimization problems need an integer analysis done either via integer or mixed integer linear programming [156]. For small problems, efficient algorithms such as branch-and-bound or LP-relaxation have been proposed. However, for large problems, the solution is still based on heuristics.

Early studies of the network dynamics use queueing theory first developed by Markov [128] and Erlang [70], formalized by Kolmogorov [112] and Kendall [104], and extensively used in the context of packet switching networks by Kleinrock [110]. Nonetheless, in order to ease the mathematical tractability, most queueing approaches rely on exponential type distributions for the event/job arrival and/or service time. As shown in several studies, this may not be an accurate model for the real network traffic. Consequently, one of our objectives in this survey is to also highlight alternatives to the conventional paradigm of network design. This new vision is based on rigorous developments in the field of statistical physics and information theory that allow us to model the network as a thermodynamical system. The hope is that this new modeling paradigm enables not only capturing the intrinsic interactions among various network components, but also helps proposing more powerful techniques for predicting and optimizing the network.

These ideas are detailed in the remainder of this survey. More precisely, Section 2 presents a graph-based formalism for the network design and the major approaches proposed in a deterministic setup. Section 3 takes a probabilistic view on designing NoC architectures under the Markovian assumption. Finally, Section 4 introduces new problems in a statistical physics-based context for network design and enumerates some preliminary steps taken toward solving these problems.

2

Deterministic Perspective

2.1 Representation

Deterministic approaches for system-level design are based primarily on graph-based modeling of application and architecture. As shown in Figure 2.1, a deterministic description of the application assumes complete information about the interaction among tasks which is captured via various parameters assigned to the edges of the application graph.

Definition 2.1. An *application characterization graph* (ACG), $App = App(V_{App}, E_{App})$, is a directed graph, where each vertex $v_i \in V_{App}$ denotes a computational module of the application referred to as a task, and each directed edge $e_{ij} \in E_{App}$ characterizes the communication process from vertex v_i to vertex v_j , where:

- Each vertex $v_i \in V_{App}$ is annotated with specific task information (e.g., execution time $\epsilon(v_i)$, energy consumption $E(v_i)$, execution deadlines $d(v_i)$, etc.).
- In turn, each arc $e_{ij} \in E_{App}$ is tagged with application-specific information (e.g., communication volume $commvol(e_{ij})$ from vertex v_i to vertex v_j in bits, etc.),

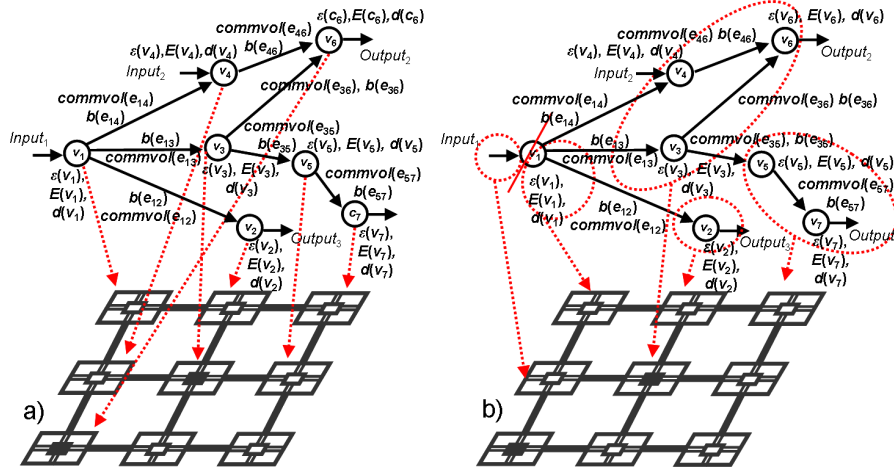


Fig. 2.1 Two instances of *mapping* an Application Characterization Graph (ACG) onto a NoC architecture with fixed parameters: (a) A simple one-to-one mapping of the ACG onto the NoC cores when the platform resources are sufficient to accommodate application requirements. (b) A task division/partitioning (corresponding to vertex v_1 in the ACG), plus clustering and mapping onto a NoC architecture such that the performance and/or energy consumption are optimized. This is typically the case when one needs to map complex applications on platforms with finite resources. In this particular examples, $\epsilon(v_1)$, $E(v_1)$, and $d(v_1)$ represent the execution time, energy consumption, and deadline of task v_1 . Similarly, $commvol(e_{14})$ and $b(e_{14})$ denote the communication volume and bandwidth constraint.

and specific design constraints (e.g., communication bandwidth $b(e_{ij})$ in bits per second, latency requirements $lat(e_{ij})$ in seconds, etc. (see Figure 2.1(a))).

Similarly, a graph-based description of the NoC architecture is as follows:

Definition 2.2. The NoC *architecture* is described by the triple $Arch = Arch(T(U, F), \Omega, S_P)$,¹ where:

- The labeled graph $T(U, F)$ represents the network topology. The routers ($r_i \in U$) and channels ($ch_{ij} \in F$) in the network are given by the sets U and F , respectively, as follows:
 - $\forall ch_{ij} \in F$, $w(ch_{ij})$ gives the channel bandwidth;

¹ For simplicity, we use sometimes the abbreviation $Arch$ instead of the complete definition $Arch(T(U, F), \Omega, S_P)$.

- $\forall r_i \in U, l(r_i, ch_{ij})$ gives the buffer size (depth) of channel ch_{ij} located at router r_i ;
- $Pos(r_i)$ gives the xy coordinates of the router r_i ;
- S_P is the set of Processing Elements (PEs);
 - $card(S_P) = N_{PE}^2$ denotes the maximum number of PEs in the set S_P .
- $\Omega: S_P \times U \rightarrow \{0, 1\}$ is a function that maps each PE $p_k \in S_P$ to a router in $r_j \in U$ (i.e., $\Omega(p_k, r_j) = 1$ if PE p_k is mapped to router r_j , otherwise $\Omega(p_k, r_j) = 0$). Note that, if the communication demands for a subset of PEs from the set S_P are low, then a single router may be assigned to the PEs in order to minimize the area and/or power overhead.

Based on these background definitions, we introduce next the concept of communication paradigm:

Definition 2.3. The *communication paradigm* is defined as

$$\mathfrak{R}\{R(v_k, r_i, ch_{ij}, r_{src}, p_{src}, r_{dst}, p_{dst}, \rho(t)), Sw\} \\ v_k \in V_{App}, r_i, r_{src}, r_{dst} \in U, ch_{ij} \in F, p_{src}, p_{dst} \in S_P, \quad (2.1)$$

where

- $R(v_k, r_i, ch_{ij}, r_{src}, p_{src}, r_{dst}, p_{dst}, \rho(t))$ is the routing policy for a packet (generated when executing task v_k) at router $r_i \in U$, given a source router $r_{src} \in U$ associated to PE p_{src} and destination router $r_{dst} \in U$ associated to PE p_{dst} ;
 - $\rho(t): F \times t \rightarrow [0, 1]$ is the utilization³ of the channels connected to the neighboring routers at time t ;
- Sw specifies the chosen packet switching technique (i.e., a protocol to forward the flit through the channel $ch_{ij} \in F$ of router $r_i \in U$ toward the router $r_j \in U$).

²The abbreviation *card* stands for the cardinality of a given set.

³In the general case, the utilization can be a function of the architectural components so ρ can be also defined on *Arch*.

Note that in the above definition, the static routing functions do not require the utilization information, while adaptive algorithms do consider the level of congestion at the neighboring routers.

Both routers and channels are frequently characterized by power/energy consumption values and performance parameters (e.g., arbitration, routing or switching delay, link speed, etc. [59, 102]). In the case of deterministic approaches, the application and architectural models are described via fixed parameters, typically estimated under worst-case assumptions.

The NoC design starts from the above descriptions, i.e., the computational model of the application and the model of the hardware platform, maps and schedules the application onto the target architecture, while enforcing a set of design constraints. The mapping and scheduling steps implement the application on the given platform by assigning the tasks and communication onto platform resources such that the design specifications are met. In the sequel, we provide a formal description of the mapping and scheduling problems in the deterministic context.

Definition 2.4. The *mapping* function $M : App \times Arch \times \mathfrak{R} \rightarrow \{0,1\}$ seeks to find the topological placement of a selected computational task $v_i \in V_{App}$ on the PE $p_k \in S_P$ connected to router $r_j \in U$, with or without considering the routing protocol $\wp_l \in \mathfrak{R}$ (i.e., $M(v_i, p_k, r_j, \wp_l) = 1$).

For instance, the goal of a mapping action can be, not only to execute a certain application onto a given architecture, but also find the minimum number of PEs which communicate through a deterministic dimension ordered routing protocol [59] such that the total or only the communication energy consumption stays within a certain budget. This implies that a task graph $App(V_{App}, E_{App})$ as in Figure 2.2 is considered to be mapped on the platform resources. If the routing policy (R) is given, then it can be abstracted away from the mapping definition.

Definition 2.5. The *scheduling* function is defined as $S : App \times Arch \times \mathfrak{R} \times t \rightarrow \{0,1\}$ and seeks to determine the start time t of the

selected computational task $v_i \in V_{\text{App}}$ on the PE $p_k \in S_P$ connected to the router $r_j \in U$, by considering the routing protocol $\wp_l \in \mathfrak{R}$ such that the task deadlines are met (i.e., $S(v_i, r_j, p_k, \wp_l, t) = 1$).

Generally speaking, optimal scheduling consists of a set of ordered messages based on their transmission time stamp and, in some cases, the routing paths as well (e.g., under certain circumstances, the scheduling is done by assuming that a minimal routing path offers the best performance). Consequently, the design of the routing protocol requires more attention. Next, we provide a definition for the routing function.

Definition 2.6. The *routing* function/protocol is defined as $R: \text{App} \times \text{Arch} \times \text{Arch} \times \text{Arch} \times \text{Arch} \times \text{Arch} \times \text{Arch} \times [0, 1] \times t \rightarrow \{0, 1\} \times t$ with $R(v_k, r_i, ch_{ij}, r_{src}, p_{src}, r_{dst}, p_{dst}, \rho(t)) = 1$ if the outgoing channel $ch_{ij} \in F$ of router $r_i \in U$ is selected for each packet generated by task $v_k \in V_{\text{App}}$ in the ACG, which is mapped on the PE $p_{src} \in S_P$. In its turn, each packet at source p_{src} needs to be routed from r_{src} to $r_{dst} \in U$, where r_{dst} is mapped to PE $p_{dst} \in S_P$ with/without considering the utilization ($\rho(t)$) of the neighboring routers.

An optimal routing policy R returns the minimum number of channels traversed between any source–destination pair, together with an arbitration scheme, such that livelock and deadlock situations are avoided. In order to avoid congestion, the limited amount of on-chip resources need to be accounted for when designing the routing protocol. Thus, routing is a dynamic function that seeks to find the best policy of sending the packets around the network, given the application characteristics, the network topology, and the PE-to-router mapping, such that the performance, power/energy consumption, and/or reliability metrics are met.

2.2 Problem Formulation

The problem of network design can be formulated as a mathematical program which aims at determining one or more of the unknowns (i.e., topology ($T(U, F)$), mapping (M), scheduling (S),

and/or routing (R) functions) that optimize the objective function $O(\text{Arch}(T(U, F), \Omega, S_P), M, S, R)$:

$$\begin{aligned}
 & \max/\min O(\text{Arch}(T(U, F), \Omega, S_P), M, S, R) \\
 & \text{s.t. for a given application } \text{App}(V_{\text{App}}, E_{\text{App}}) \\
 & \quad \text{and architecture } \text{Arch}(T(U, F), \Omega, S_P) \\
 & g_k(\text{App}, \text{Arch}, M, S, R, t) \leq h_k, \quad k = 1, \dots, N \quad (2.2)
 \end{aligned}$$

where g_k are some functions that represent performance (e.g., bandwidth, throughput, node-to-node latency, average packet latency), power/energy consumption, and/or reliability metrics, while h_k are the specified constraints of the design. The solution of Equation (2.2) is greatly affected by the choice of g_k functions. Also, to obtain efficient NoC implementations, the optimization process should consider accurate area and/or energy models. For instance, Balfour and Dally [15] show how area and energy consumption is affected by the topology, channel width, routing strategy, and buffer size.

In order to design highly optimized NoC architectures, the designer should address some challenging problems like: topology synthesis and floorplanning/placement, resource allocation, binding of computational tasks and communication requirements to the target architecture, task scheduling, etc. [145]. The degree of success in addressing each sub-problem is measured in terms of power/energy consumption and performance figures.

Based on modeling graphs via adjacency matrices, most deterministic approaches to NoC design use traditional mathematical techniques such as linear programming [175], quadratic programming [156, 157], convex optimization, mixed integer linear and/or nonlinear programming (MILP/NLP) [23, 156, 173], evolutionary computing or Genetic Algorithms (GA) [169, 214], branch-and-bound approaches [41], or classical greedy based heuristic algorithms [54, 131]. To solve complex optimization problems many authors rely on various heuristics such as simulated annealing [109], coloring algorithms [82], Fiduccia-Mattheyses [73] or recursive searching algorithms [54].

2.2.1 Topology Synthesis and Mapping Problems

To give a bit of intuition, let us consider first the topology synthesis and mapping problems together. As such, we consider a shortest path routing strategy (R), an earliest-deadline-first scheduling policy (S) and also assume that complete information about the ACG is given. We aim at determining the graph $T(U, F)$ (i.e., the number of routers in the set U and number of links in the set F), the number of PEs in the set S_P , the processor-to-router assignment function Ω , and the mapping function M which assigns the computational tasks to the PEs such that the performance and/or power/energy constraints are satisfied.

Obviously, there is a cost for matching a subset of computational tasks to a certain PE. For instance, Figure 2.1(a) shows a one-to-one mapping of a set of computationally intensive tasks in the ACG onto an NoC architecture. In contrast, Figure 2.1(b) illustrates how multiple tasks can be clustered and mapped on a single NoC tile for improving the system performance (e.g., the clustering of a few highly dependent tasks like v_3 , v_4 , and v_6 on a single PE can overcome the communication penalty). There are also restrictions on the maximum number of computational tasks that can be mapped to a certain PE such that the global or local execution time or the power/energy bounds are not exceeded (see Figure 2.1(b) where the computational task v_1 needs to be split between two PEs).

As shown in Figure 2.2, we aim at finding the best matching between the computational tasks, on one hand, and the processing elements and routers, on the other hand, such that no resource overlap is allowed and the total benefit/penalty is maximized/minimized. Obviously, from a technology point of view it is desirable to minimize the number of PEs and routers as this may translate into energy efficient or high performance hardware platforms. Consequently, based on the ACG information, we decide first on all the possible subsets of partitioned configurations of computational tasks (e.g., an application partitioning method for parallel architecture is discussed in [16]) which we denote as Φ (see the left-hand side of Figure 2.2). Knowing the maximum number of PEs (N_{PE}) available to use, we construct the set Ψ of all possible configurations; this implies that we can use one or more

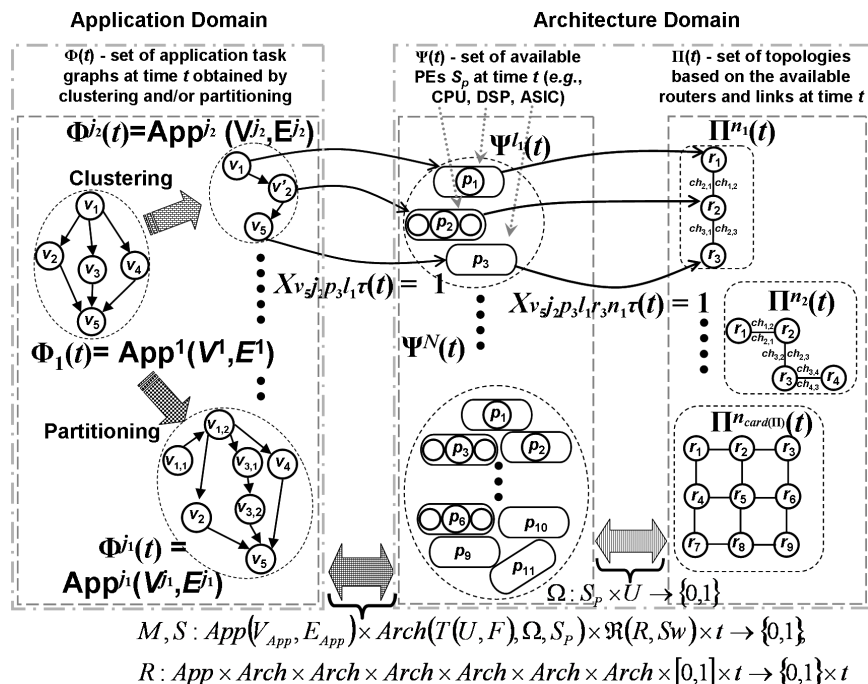


Fig. 2.2 The NoC design problem starts from the description of the application (i.e., a set of task graphs $App(V_{App}, E_{App})$ obtained by clustering/partitioning as in Figure 2.1) and the architecture $Arch = Arch(T(U, F), \Omega, S_P)$. The goal is to find the topological graph $T(U, F)$, the processor-to-router assignment function (Ω), the computational task-to-processing element mapping (M), scheduling (S), and/or the routing (R) function. For instance, the mapping function M can be expressed by variable $X_{ijklmn\tau}$ showing that the vertex $v_i \in V_{App}^j$ from a task graph $App^j(V_{App}^j, E_{App}^j)$ is assigned to processor $p_k \in S_P$ from set Ψ^l and bound to router $r_m \in U^n$ from set Π^n . Note that various PEs (e.g., CPU, DSP, ASIC) in the set of available processing elements $\Psi(t)$ are described via different graphical representations.

PEs to execute an application, but not more than N_{PE} . We also consider the set $\Pi = \{T^j(U^j, F^j) | 1 \leq j \leq card(\Pi), card(U^j) \leq card(S_P)\}$ of all topological configurations consisting of routers interconnected by channels (see the right-hand side of Figure 2.2).

Using the above notation (also summarized in Table 2.1), we try to determine the value of the binary assignment tensor $X_{ijklmn\tau}(t)$, where $X_{ijklmn\tau}(t) = 1$, if node $v_i \in V_{App}^j$ in configuration Φ^j is mapped on the PE $p_k \in S_P^l$ from the set Ψ^l which is connected to the router $r_m \in U^n$ from set Π^n and scheduled to start at clock cycle t (i.e., $\tau = t$); otherwise $X_{ijklmn\tau}(t) = 0$. For instance, the expression $X_{v_5 j_2 p_3 l_1 r_3 n_1 \tau}(t) = 1$ in

Figure 2.2 shows that the task $v_5 \in V_{App}^{j_2}$ in configuration Φ^{j_2} is mapped on PE $p_3 \in S_P^{l_1}$ from Ψ^{l_1} which is connected to the router $r_3 \in U^{n_1}$ from Π^{n_1} . Obviously, each such assignment has associated a certain cost $c_{ijklmn\tau}$. The problem of topology synthesis and mapping can be formulated as follows:

$$\begin{aligned}
& \min_{\Phi^j \in \Phi, \Psi^l \in \Psi, \Pi^n \in \Pi} \sum_{i=1}^{\text{card}(V_{App}^j)} \sum_{k=1}^{\text{card}(S_P^l)} \sum_{m=1}^{\text{card}(U^n)} c_{ijklmn\tau} X_{ijklmn\tau}(t) \\
& \text{s.t. for a given application } A \in App(V_{App}, E_{App}) \\
& \text{and possible configurations } V_{App}^j \in \Phi^j, S_P^l \in \Psi^l, U^n \in \Pi^n \\
& \sum_{i_1, i_2=1}^{\text{card}(E_{App}^j)} b(e_{i_1 i_2}) I(ch_{o_1 o_2}, R(X_{i_1 j k_1 l m_1 n}(t), \\
& \quad X_{i_2 j k_2 l m_2 n}(t))) \leq b_0(ch_{o_1 o_2}) \\
& \text{lat}(X_{i_1 j k_1 l m_1 n}(t), X_{i_2 j k_2 l m_2 n}(t)) \leq \text{lat}_0(e_{i_1 i_2}), \\
& \quad i_1, i_2 = 1, \dots, \text{card}(E_{App}^j) \\
& \sum_{i_1, i_2=1}^{\text{card}(E_{App}^j)} \text{commvol}(e_{i_1 i_2}) E(e_{i_1 i_2}, M(X_{i_1 j k_1 l m_1 n}, X_{i_2 j k_2 l m_2 n})) \leq E_0 \\
& E_{App}^j \in \Phi^j, \forall ch_{o_1 o_2} \in F^n, o_1, o_2 = 1, \dots, \text{card}(U^n), U^n \in \Pi^n. \quad (2.3)
\end{aligned}$$

Table 2.1 List of main variables used in the formulation of topology synthesis and mapping problem (i.e., Equation 2.3). Note that for $\Phi(t)$, $\Psi(t)$, and $\Pi(t)$ sets, the time dependency is intended for capturing both, the static and dynamic nature of mapping, scheduling, and network reconfiguration.

Notation	Description
$\Phi(t)$	The set of possible ACG configurations obtained via clustering and/or partitioning operations at time t .
Φ^j	An ACG configuration from set $\Phi(t)$ obtained via clustering and partitioning operations.
V_{App}^j	The set of vertices in the ACG Φ^j .
E_{App}^j	The set of edges in the ACG Φ^j .
$\Psi(t)$	The set of all possible configurations that can be built with the available PEs.
Ψ^l	A chosen subset of PEs from the available set $\Psi(t)$.
$\Pi(t)$	The set of all possible topological configurations.
Π^n	A topological configuration from set $\Pi(t)$ consisting of routers and channels.
$X_{ijklmn\tau}(t)$	If the binary tensorial variable $X_{ijklmn\tau}(t)$ is equal to 1 then it shows that node $v_i \in V_{App}^j$ in configuration Φ^j is mapped on the PE $p_k \in S_P$ from the set Ψ^l which is connected to the router $r_m \in U^n$ and scheduled to start (i.e., $\tau = t$) at clock cycle t .
$c_{ijklmn\tau}$	The cost associated to a particular assignment $X_{ijklmn\tau}$.

The objective of Equation (2.3) is to determine the value $X_{ijklmn\tau}(t)$ and implicitly the optimal application-to-PE mapping, as well as the best PE-to-router assignment. In general, the costs $c_{ijklmn\tau}$ also depend on variables $X_{ijklmn\tau}$ making the optimization problem nonlinear. For example, the authors in [162] investigate the topology synthesis problem by clustering the communication constraint graph. The authors assume that the cost of implementing a constraint arc is proportional with the Euclidean norm of the distance between two computational modules and a concave function of the required bandwidth. In contrast, if the costs $c_{ijklmn\tau}$ are constant and the constraints are linear in $X_{ijklmn\tau}$ variables, then the topology synthesis can be solved via the LP approach as shown in [192].

Coming back to Equation (2.3), the first constraint computes the traffic on each link in the architecture and ensures that the bandwidth $b_0(ch_{o_1o_2})$ ($\forall ch_{o_1o_2} \in F^n$, $o_1, o_2 = 1, \dots, card(U^n)$) is not exceeded. Note that in Equation (2.3) a simplified notation for the routing function is used. Here, the routing function $R(X_{i_1j_{k_1}l_{m_1}n\tau}(t), X_{i_2j_{k_2}l_{m_2}n\tau}(t))$ encodes the path from source p_{k_1} to destination p_{k_2} which is assumed to be known (i.e., shortest path). The index function $I(ch_{o_1o_2}, R(X_{i_1j_{k_1}l_{m_1}n\tau}(t), X_{i_2j_{k_2}l_{m_2}n\tau}(t)))$ can take only binary values showing that the channel $ch_{o_1o_2} \in F^n$ from configuration Π^n is used when routing packets from source $p_{k_1} \in S_P^l$ to destination $p_{k_2} \in S_P^l$.

The second constraint in Equation (2.3) quantifies the communication latency $lat(X_{i_1j_{k_1}l_{m_1}n\tau}(t), X_{i_2j_{k_2}l_{m_2}n\tau}(t))$ between two tasks $v_{i_1}, v_{i_2} \in V_{App}^j$ from configuration Φ^j which are mapped to PEs $p_{k_1}, p_{k_2} \in S_P^l$ in configuration Ψ^l . The PEs $p_{k_1}, p_{k_2} \in S_P^l$ are connected to routers $r_{m_1}, r_{m_2} \in U^n$ from set Π^n . This source-to-destination latency must not exceed the design specification $lat_0(e_{i_1i_2})$.⁴

The last constraint in Equation (2.3) expresses the fact that the total communication energy needs to stay within a certain bound E_0 . The function $M(X_{i_1j_{k_1}l_{m_1}n\tau}(t), X_{i_2j_{k_2}l_{m_2}n\tau}(t))$ shows that the application vertices $v_{i_1}, v_{i_2} \in V_{App}^j$ are mapped on the PEs $p_{k_1}, p_{k_2} \in S_P^l$; it needs to be determined by finding the binary values $X_{ijklmn\tau}(t)$.

⁴ Communication latency is usually expressed as the number of hops between source and destination; other approximations are possible as well.

The $E(e_{i_1 i_2}, M(X_{i_1 j k_1 l m_1 n \tau}(t), X_{i_2 j k_2 l m_2 n \tau}(t)))$ term represents the average energy consumption of sending one bit of data from vertex $v_{i_1} \in V_{\text{App}}^j$ which is mapped on PE $p_{k_1} \in S_P^l$ to vertex $v_{i_2} \in V_{\text{App}}^j$ which is mapped on $p_{k_2} \in S_P^l$.

In reality, only a subset of the above constraints are actually imposed to lessen the optimization problem and shorten the design time. Note also that to ease the understanding of the topology synthesis and mapping problem, we assume uniform buffer assignment and sufficiently large buffer sizes to accommodate the network traffic.

2.2.2 Routing Problem

Let us analyze now how a routing optimization problem can be formulated when considering an already mapped and scheduled set of applications onto a mesh NoC architecture. More precisely, we aim at determining the shortest path routing function (R) for any source–destination pair, such that the link bandwidth constraints are satisfied (e.g., the volume of traffic along a particular link does not exceed its capacity) and the overall communication energy consumption is minimized. We also assume that if the link bandwidth constraints are satisfied, the packets are successfully routed and transmitted from any source to any destination.

Formally, the link bandwidth constraints can be expressed via a matrix approach as follows:

$$R(t) \cdot \text{commvol} \leq B, \quad (2.4)$$

where $R(t)$ is an unknown matrix with the entries R_{kl} being 0 or 1 depending on whether or not link k is used by the source–destination path l (its size is given by the total number of links and the number of source–destination pairs), each element $\text{commvol}_l(f_{ij})$ in vector commvol gives the communication volume between source $p_i \in S_P$ and destination $p_j \in S_P$, and B is a vector of bandwidth constraints (i.e., each entry B_k in vector B stands for the maximum flow that link k can sustain). Similarly, we can express the energy consumption constraint as follows:

$$E = e^T \cdot R(t) \cdot \text{commvol}, \quad (2.5)$$

where e^T denotes a row vector of per hop energy consumption per transmitted bit; its size is equal to the total number of links. Since we search for a shortest path between a source–destination pair l , we have to minimize the number of links along such a path (i.e., $[1^T \cdot R(t)]_l, \forall l$). Consequently, the fixed routing problem can be formulated as follows:

$$\begin{aligned}
 \min_R \quad & O = e^T \cdot R(t) \cdot commvol + 1^T \cdot R(t) \\
 \text{s.t. for a given application } & A \in App(V_{App}, E_{App}) \\
 & \text{mapped and scheduled on topology } T \in T(U, F) \\
 & R(t) \cdot commvol \leq B.
 \end{aligned} \tag{2.6}$$

An example of a routing problem for irregular topologies is discussed in [37], where the authors try to find a fixed routing function R with minimum hardware cost of routing tables. The authors propose two routing techniques: the turn-table routing and the XY -deviation table. The turn-table routing can be cast as an optimization problem described in Equation (2.6) with the distinction that it does not search for a single shortest path from each source to each destination, but instead for a covering set of paths that minimize the total number of entries in the network turn-tables. This implies that the entries in the columns of matrix R (which are independent of time t in the case of static routing) represent *all* possible paths between all source–destination pairs. The result of the optimization problem ($R_{kl} = 1$) shows that link k was selected for path l in the cover set of selected shortest paths. In contrast, the XY -deviation routing seeks to determine a single path l which minimizes the number of routing steps (i.e., minimum number of 1 entries in the column l of matrix R) and can be expressed as in Equation (2.6) if the energy consumption is ignored.

To ensure fault-tolerance against faults in NoCs, Murali et al. in [134] formulate a multi-commodity flow problem similar to the one presented in Equation (2.6) with the goal of minimizing the traffic on each link by spreading and splitting the traffic through multiple paths across the network.

2.3 Literature Survey

Network-based communication offers the possibility to communicate more efficiently and robustly than via dedicated wires [22, 27, 60, 87, 118], but, at the same time, brings to attention difficult problems such as topology synthesis, router design, bandwidth allocation or channel design, floorplanning and layout design [145]. Consequently, the NoC design problems cannot be solved independently of the application characteristics (e.g., latency, throughput, area, power/energy consumption) as such solutions may be inefficient or impractical. Moreover, the designer needs to address also the fault-tolerant component in DSM technologies via routing approaches [65, 134] or via error control schemes [10, 67, 135].

2.3.1 Communication Infrastructure

The communication infrastructure synthesis aims to find the topology, size of the buffers, width of the channels and floorplanning, such that, for a given application $App(V_{App}, E_{App})$, some performance constraints are satisfied (e.g., network latency, throughput, etc.).

2.3.1.1 Topology Synthesis and Floorplanning

One of the early works designing minimal networks for special purpose computer systems with well behaved communication patterns was proposed in [90]. The proposed topology generation algorithm is based on a recursive bisection technique. More precisely, the approach starts from a set of topologies similarly to set Π in Section 2.2.1; then, based on a contention model extracted by analyzing the communication behavior of the application, it constructs the network via systematic partitioning. Each partitioning step is followed by simulated annealing, which seeks to optimize the number of network partitions by determining the placement of processors linked to the switches and the routing protocol. Finally, a coloring algorithm is used to find the number of links required to support the communication between these partitions. We should note that the algorithm does not guarantee contention free designs. The overall complexity of the topology synthesis algorithm is

$O(N^2KL)$, where N is the number of partitions, K is the number of contention periods, and L is the number of cliques.

Pinto et al. [162] propose a constrained driven communication architecture synthesis of point-to-point links by utilizing a heuristic based on k -way merging. Similarly to Figure 2.2, the authors assume a constrained graph which characterizes the communication between two already mapped vertices $v_{i_1}, v_{i_2} \in V_{\text{App}}^j$ onto processing elements and a communication library. The objective is to determine the assignment function Ω in Equation (2.2) between the mapped application and the communication library. The proposed heuristic approach does not consider explicitly the routing issues and consists of two phases: First, a quadratic programming technique estimates the cost of satisfying a set of arc constraints (this is similar to the bandwidth constraint in Equation (2.3)). Second, one of the hierarchical heuristics (i.e., either divisive clustering or agglomerative clustering) is used to explore efficiently the set of all possible clusterings. From a complexity standpoint, the approach improves over the previous method by requiring $O(n^2)$ steps for the divisive clustering or $O(n^3)$ for the agglomerative clustering, where n is the number of links in graph $T(U, F)$.

In [189], the authors propose a genetic algorithm for application-specific NoC synthesis that corresponds to finding the binary values of $X_{ijklmn\tau}(t)$ variable in Equation (2.3) which optimizes the power consumption and area of the design such that the performance constraints (e.g., required bandwidth in bits per cycle, latency in hops) are met. This technique operates in an iterative manner by generating and selecting a constant number of intermediate solutions that satisfy the fitness criteria. Each solution consists of three levels: the first level represents the number of routers (I), the second defines the mapping of the communication task nodes to the router ports (J is the number of port mappings), the third level specifies the routing protocol (K is the number of mappings of the communication traces on the router ports). The product $I \times J \times K$ shows the space required by the genetic approach. The fitness function is inversely proportional with the linear combination between power, area, and unmapped traces of a given solution. However, the algorithm may pose serious problems in finding the best topology since the accuracy depends on the fitness function.

The work in [146] proposes a branch-and-bound algorithm to decompose the communication requirements of the target application and then construct the set Φ of application configurations as in Section 2.2.1. The algorithm searches the entire design space Π for a specific topology (i.e., determines a specific value for variable $X_{ijklmn\tau}(t)$) which minimizes the communication energy consumption. This means that the third constraint in Equation (2.3) is used as an objective function.

Srinivasan et al. [192] solve the topology synthesis and floorplanning problem via an MILP approach which accounts for link power consumption. In contrast to the formulation presented in Equation (2.3), the authors assume that the application is already mapped on a set of processors and then try to minimize the cumulative traffic flow through the ports of all routers in the network (i.e., the generic costs are replaced by total bandwidth usage) such that the bandwidth, traffic routing and latency constraints are satisfied and resource conflicts avoided. As opposed to the approach in Figure 2.2, the authors assume that all routers are identical and have the same number of bidirectional ports. To determine the minimum number of routers and links in the set $T(U, F)$, in this approach the binary tensor $X_{ijklmn\tau}(t)$ is time independent and split into two variables: a computational module-to-router mapping variable and a router-to-router assignment variable.

In [191], Srinivasan and Chatha propose a heuristic approach to topology synthesis and floorplanning which can be summarized as follows: First, a system-level floorplanning (i.e., an MILP or a slicing tree based recursive partitioning algorithm) takes the communication task graph and the characteristics of the NoC components as inputs and generates the layout that minimizes the power consumption of the interconnects; this is achieved by assuming that the power consumption of the NoC is directly proportional to network bandwidth. Second, a heuristic recursively determines the bandwidth of the physical links, incorporates the routers to the previously obtained topology and maps the communication traces, while minimizing the power consumption and the number of utilized resources via a router merging technique. However, since the merging of routers can have a high impact on network performance, the design optimality is not guaranteed.

Ogras and Marculescu [147] present a low-complexity methodology which automatically synthesizes an irregular topology consisting of inserting few application-specific long-range links to a standard mesh NoC with the objective of maximizing the packet injection rate at which the network saturates. This approach redefines the idea of *small world networks* [207, 75] in the context of application specific NoCs. As such, in contrast to Equation (2.3), the algorithm starts from a regular mesh architecture and greedily inserts long-range links between the tiles with the highest communication frequency. The algorithm stops when the set of available resources becomes empty. This approach can be cast in the spirit of Equation (2.3) by introducing new variables X_{ijk} which indicate that a long-range link i is inserted between the nodes j and k . Although, this approach aims at maximizing the network utilization, it does not guarantee an optimum design. Along the same lines, Chang et al. [47] investigate an NoC design, where the long-range connections are established via wireless communication.

An alternative approach to the above mentioned algorithms is to formulate the topology synthesis as a convex optimization problem [76]. In contrast to the formulation presented in Equation (2.3), the formalism in [76] ignores the information about the possible applications running on the targeted architecture. Instead, it tries to find the minimum number of links needed for maximizing the algebraic connectivity of the graph $T(U, F)$ which corresponds to the second smallest eigenvalue of the graph Laplacian.

Murali et al. [138] propose a heuristic to synthesize the topology and find the floorplanning information. The design flow starts from a set of specified objectives and constraints (e.g., application traffic characteristics, cores size, etc.), varies the operation frequencies and link widths within a prescribed range, the number of switches from one to the total number of cores, and chooses a good topology for a certain frequency, link-width, and switch count. Next, the algorithm floorplans the synthesized topology via the Parquet algorithm [4] and chooses a good topology that optimizes the predefined objectives (e.g., power consumption, area, etc.). Nevertheless, the optimality of the solution is not guaranteed.

In order to solve the floorplanning problem alone, Ye and De Micheli [211] use a QP approach. The physical planning tool addresses the planarization process constrained by aspect ratios, preservation of regularity and hierarchy, and minimization of wire length and power consumption. To address the floorplanning issues, the problem formulation in Equation (2.3) can be augmented with new binary variables which account for the placement of the obtained architecture as given by the $X_{ijklmn\tau}(t)$ variables.

Besides the total area, regularity, power and inter-router distances, a good floorplanner should also consider the channel latency. Consequently, an NoC floorplanner can place the routers and repeaters according to the latency-insensitive design methodology. More precisely, in [43], Carloni et al. propose a pipeline-based design methodology based on a common clock signal and zero-delay channels. This approach consists of relaxing the time constraints during the design phase. Possible time mismatches between the initial time constraints and actual interconnect delays are resolved by partitioning long intermodule wires into segments that obey the time constraints and inserting relay stations which play a similar role to the latches in a pipeline datapath. However, careful analysis is needed to make sure this will not penalize the entire design in terms of area, power or performance.

2.3.1.2 Buffer Sizing

Targeting the optimization of communication of application specific NoCs, Manolache et al. [126] propose a strategy for synthesizing the communication infrastructure of the application specific NoCs and determining the packet release time such that the overall buffer space is minimized. Similarly to the second constraint in Equation (2.3), an off-line packet routing estimates the latency for a source-to-destination pair under imposed buffer capacity constraints. This is done via a greedy heuristic which maps each message to a source-to-destination path only if it minimizes the buffer space that is, the overall buffer space becomes the cost function in Equation (2.2). For each task $v_i \in V_{\text{App}}^j$ mapped on a certain path, a greedy heuristic is used to determine the amount of

time the task can be delayed such that the contention at the destination buffers is minimized and deadlines are met. Although the proposed greedy approach proves to be fast when compared to exhaustive search, it does not guarantee the buffer space optimum solution.

To satisfy the prescribed performance level for multimedia applications, Stuijk et al. [195] propose a technique for finding the optimal trade-off between the network throughput and buffer size for a given Cyclo-Static Data Flow (CSDF) graph. The iterative algorithm starts from an initial set of buffer size distributions and then explores the buffers in the increasing order of their size (i.e., smallest size configurations are considered first). For each distribution size, a dependency graph containing the causal dependencies between the node firings of the communication periodic phases and its corresponding throughput are computed. The loop terminates when the constructed storage distribution offers the maximal throughput and all other storage distributions of equal size have been already evaluated. The next step is to remove all nonminimal storage distribution-throughput pairs. To account for the buffer size of each channel in Equation (2.3), the binary tensorial variable $X_{ijklmn\tau}(t)$ should be augmented with new indices which refer the buffer size for all channels connecting router $r_m \in U^n$ with its neighbors.

One major drawback of the approach in [195] is that some applications have a large number of throughput-buffering Pareto points which can increase the runtime. The paper also presents an approximation algorithm which increases the amount by which the bottleneck channels are sized in order to reduce the number of storage distributions and speed up the search. However, this approach affects the optimality of the solution.

2.3.2 Communication Paradigm

2.3.2.1 Routing

Besides the synthesis of the architecture and the application mapping, the NoC designers face the challenging problem of constructing the routing protocol. Consequently, starting from the application characterization graph and concrete description of the NoC architecture,

one tries to determine a routing protocol such that the performance, power/energy consumption, and/or reliability are optimized. If the workload dynamics and reliability issues are ignored during the optimization problem, then graph-theoretic approaches such as shortest path algorithm [54] for irregular topologies, or dimension ordered routing algorithm [59] for regular topologies can be used. Typically, such approaches seek to minimize the average distance (in number of hops) each packet travels between any source–destination pair.

Addressing only the deterministic/static routing protocols (i.e., time-independent variable R_{kl} in Section 2.2.2) for irregular NoCs, Bolotin et al. [37] propose path selection algorithms which minimize the overall hardware cost of the routing tables. An LP approach to the routing problem has been proposed in [191]. The goal of this approach is to determine the topology, the floorplanning information, and the static routing of communication traces such that the performance (e.g., bandwidth) and power consumption are optimized.

An alternative approach to deterministic routing is to consider the network workload, power or temperature related issues and design an adaptive routing protocol that offers better performance (i.e., determine the *dynamic* variable $R_{kl}(t)$, which denotes that link k should be used by the time-dependent flow along the path l in the network). Catania et al. [45] propose a heuristic based on a prescribed mapped application. This approach first constructs a graph of source–destination pairs and then uses a heuristic to break the cycles in this static communication graph. Again, the optimality of the adaptive routing protocol cannot be guaranteed.

To lower the overhead of implementing adaptive routing, Abad et al. [1] describe a router architecture which consists of two independent rings forcing the packet to circulate through each port reducing the head of line blocking.

In [93], Hu and Marculescu set forth a hybrid strategy (called DyAD) that switches between the deterministic/static dimension ordered (R_{kl}) and adaptive ($R_{kl}(t)$) routing approaches [64]. More precisely, during network operation each router monitors the traffic load on its links. If the traffic load is low, then a deterministic XY protocol is used. On the contrary, if the traffic load is high, then an adaptive

routing (which avoids highly utilized links) is used. Along the same lines, Nilsson et al. [142] propose the so-called deflective routing for regular meshes. This approach aims at determining a dynamic $R_{kl}(t)$ function which routes the incoming packets to one of the free output channel in case of weak congestion.

The work in [176] proposes an oblivious routing algorithm for regular mesh NoCs which generates single or multiple paths between source–destination pairs without considering the actual network traffic.

In order to provide fault-tolerance to the communication protocol, Murali et al. [134] propose an LP approach for finding the multi-path routing strategy; the idea is rooted in [65] in the sense that it exploits path based diversity. The objective in [134] is to minimize the maximum traffic on each link such that the application traffic requirements are satisfied. To cope with transient and permanent link failures, the authors consider the number of times a packet is retransmitted along a certain path and the number of paths along which the packet is sent in the expression of traffic flow constraints. This corresponds to augmenting the variable R_{kl} in Equation (2.6) with new indices (e.g., $R_{klm}(t)$) needed to denote that the packet generated at time t by node $v_k \in V_{\text{App}}$ is sent along path l , m times. To ensure that a packet is sent via multiple paths, a new constraint of the form $e^T \cdot R_{k, :, m}(t) \geq n$, $\forall k \in \text{card}(V_{\text{App}})$ (showing that the packet generated at $v_k \in V_{\text{App}}^j$ is sent along at least n paths) should be added to Equation (2.6).⁵ However, this approach requires some hardware overhead and does not guarantee packet delivery, as faults can occur at node level too.

Using search algorithms, Kim et al. [107] investigate several routing approaches (i.e., minimal adaptive (MINAD), Valiants nonminimal oblivious (VAL), universal globally adaptive load-balanced (UGAL), adaptive Clos (CLOSAD)) in flattened butterfly networks. All these adaptive algorithms use the output channel queue length information to determine the next traversed channel at each intermediate node for any source–destination pair. In all these cases, the routing problem is

⁵The variable $R_{k, :, m}(t)$ shows that the packet generated at time t by node $v_k \in V_{\text{App}}$ is sent m times via multiple paths. The semicolon “:” denotes a vector of 0s and 1s showing whether or not link l is chosen for routing (i.e., $R_{klm}(t) = 1$).

similar to Equation (2.6) with the distinction that the queue sizes are used to determine the dynamic variable $R_{kl}(t)$.

The ultra-low latencies [79], very high frequencies [201], and highly constrained power consumption values make the design of routing protocols for NoCs difficult. Moreover, the packet size influences the power consumption too. Consequently, optimization techniques such as the one suggested in Equation (2.6) are needed to determine the optimal trade-off between power consumption and reliability.

2.3.2.2 Scheduling

The performance of an application running on NoCs depends on the time spent for computation and communication. While the computation time is primarily determined by the design of the IP cores implementing the application, the communication time depends, not only on the routing protocol, but also on the manner in which the communication and tasks are scheduled. Therefore, we review next the solutions proposed so far for application scheduling problem.

The application scheduling problem starts with the communication task graph and the architecture characterization graph and tries to find a scheduling function (S) for each task such that the performance and/or power/energy consumption and/or reliability are optimized. More precisely, the function S returns the start time τ of a mapped task represented by the tensorial variable $X_{ijklmn\tau}(t)$ in Equation (2.3).

The scheduling problem has been first addressed in computer science in the context of real-time operating systems and later on in embedded systems. For instance, in [123], Luo and Jha present a power-aware scheduling algorithm for multi-rate periodic task and aperiodic task graphs in distributed real-time embedded systems [113]. The algorithm schedules first the periodic tasks which have hard deadlines, and then (via an online scheduler) the aperiodic tasks with soft deadlines such that the overall response time is minimized. To lower the power consumption, the online scheduler uses a dynamic voltage scaling and power management technique.

Along the same lines, Shin et al. [184] propose an intra-task dynamic voltage scheduling framework based on static timing analysis, which

aims at minimizing the energy consumption for hard real-time applications by adjusting the voltage and the clock speed within a task. This approach assumes that the Worst-Case Execution Time (WCET) is available *prior* to the run-time execution of the application. In this framework, the energy consumption of each task is dependent on a speed update ratio parameter. To find the speed update ratio parameters, a gradient descent search algorithm is used. Nevertheless, the approach does not guarantee the optimal solution as the sequence of tasks and their worst execution time may not be available in advance.

Poplavko et al. [165] use Synchronous Data Flow (SDF) graphs to model the computation and communication in MPSoCs. This approach divides the application into processes that run on a specific processor and determines the intra-job scheduling. Putting this into perspective, the binary tensorial variable $X_{ijklmn\tau}(t)$ in Equation (2.3) is reduced to the simpler version $X_{ik\tau}(t)$ meaning that the task $v_i \in V_{\text{App}}^j$ mapped on PE $p_k \in S_P^l$ is scheduled to start at time t . The timing analysis of the intra-job scheduling is done via a static-order self-timed scheduling described in [193]. The major drawback of this approach is that it assumes that the jobs do *not* interact with each other. This is not a realistic assumption for networked architectures.

To solve the scheduling problem, Hu and Marculescu [94] propose a heuristic scheduling based on slack-budgeting. Simply put, the idea is to allocate more slack to applications that have a higher impact on performance and energy consumption. The algorithm computes first a weighted cost which is proportional with the execution time and energy consumption for each task in the ACG; this weight is used as a priority in the scheduling algorithm to find the start time τ . To compute the slack to be allocated to the remaining tasks (i.e., the ordering is based on their decreasing weights), the algorithm uses the total execution time derived from the ACG. To avoid missing deadlines, the algorithm iteratively checks the generated schedule and swaps the order of execution of noncritical tasks with critical ones mapped on the same PE (i.e., updating the binary variables $X_{ijklmn\tau}(t)$). If the swapping helps reducing the missing deadline, then the schedule is kept; otherwise it is rejected. To reduce the impact of heavy computational tasks on energy consumption, a task migration strategy is applied if it minimizes

the missing deadlines. Finally, the algorithm does not guarantee the optimum schedule as the ACG can contain dependencies among its tasks.

In [108], Kim et al. propose a scheduling and arbitration algorithm for CDMA-based NoCs. The approach consists of assigning Orthogonal Variable Spreading Factor (OVSF) codes to channels on a time-slot basis, which are obtained via a Dual Round Robin Matching (DRRM) algorithm. The combination of OVSF code-words with DRRM algorithm is meant to guarantee a schedule which minimizes the modulation/demodulation overhead, the average packet latency, and power consumption.

An ILP-based approach to the scheduling problem is proposed in [120]. This approach starts from a task graph description of the application which is mapped onto an NoC and considers both the guaranteed service and the best effort traffic. Then, the packets are scheduled on the links such that the maximum latency of the guaranteed service application is minimized with minimum utilization of network resources. In this case, the optimization problem tries to determine the binary variable $X_{ik\tau}(t)$, which shows that the packet generated at vertex $v_i \in V_{\text{App}}$ (which is already mapped) is scheduled to traverse the k th link when $\tau = t$.

The work in [194] considers the scheduling of multiple communication patterns and proposes several heuristics (i.e., a greedy algorithm, a backtracking approach, and an oracle approach which finds the utilization of each link before scheduling the packets) to solve the scheduling problem while minimizing the resource utilization.

An issue related to scheduling is task migration. In this context, a prediction mechanism should identify not only the start time τ in the binary variable $X_{ijklmn\tau}(t)$, but also the migration time. In [26], Bertozzi et al. propose an user-managed migration strategy based on code checkpoint and user-level middleware support for minimizing the execution time of a task migration event.

Most of the scheduling approaches to NoC design assume that explicit start times and/or WCETs are computed statically for each task $v_i \in V_{\text{App}}$. However, many real-world applications do not fall under the static cyclic scheduling framework. For instance,

Manolache et al. [126] solves via a greedy heuristic the problem of determining the timing of packet release (i.e., finding the binary variable $X_{ijklmn\tau}(t)$) such that the destination contention is minimized and its deadline is not missed.

Regarding the congestion in the NoC architectures, van der Brand [200] propose a Congestion-Controlled Best-Effort (CCBE) strategy which monitors the link utilization and, using a Model Predictive Controller (MPC) [172], determines the maximum load that can be supported by the CCBE connections under deterministic routing. The MPC controller is obtained via a QP approach and takes as constraints the measured link utilizations. The objective of the controller is to determine the set of actions needed such that the link utilization reference is followed. An alternative to the congestion control mechanisms is to use multiple priority levels for providing Quality-of-Service (QoS) requirements (i.e., delay, throughput) [36, 127]. This implies that the urgent traffic with higher priority is routed first.

2.3.3 Architecture Mapping and Optimization

2.3.3.1 Application Mapping

The mapping problem for NoCs has been first addressed in [92] and later extended in [95]. In these papers, the authors propose a branch-and-bound algorithm that maps a set of tasks onto a regular NoC and provides a routing path allocation strategy such that the communication energy is minimized while the bandwidth constraints are satisfied. In this case, the binary tensorial variable $X_{ijklmn\tau}(t)$ in Equation (2.3) is time independent and so it simplifies to a matrix element of the form X_{ik} ; this means that a computational task $v_i \in V_{App}$ is mapped onto a processing element $p_k \in S_P$ under the assumption that each PE is attached to a local router; that is, $card(S_P) = card(U)$. While the branch-and-bound algorithm offers optimal solutions for small examples, its exponential complexity is avoided in [92] by computing some lower and upper bound costs used to legalize the leaf node solutions which correspond to a complete mapping of the computational modules to the NoC tiles.

With the objective of minimizing the overall execution time of the task graph, Lei and Kumar [119] propose a genetic algorithm that maps the parametrized task graph onto a 2D regular NoC architecture. In this framework, the authors assume that the topology $T(U, F)$ is given, and therefore try to find a simplified version of the binary variable $X_{ijklmn\tau}(t)$ which shows on which PE $p_k \in S_P$ a certain task $v_i \in V_{\text{App}}$ is mapped. The complexity of the proposed approach is polynomial $O(\langle N_V \rangle \times N_{\text{PE}}^m)$ on the chromosome size m , where $\langle N_V \rangle$ and N_{PE} represent the average number of tasks mapped on a PE and the number of PEs. Nevertheless, the proposed algorithm does not guarantee optimality.

Murali et al. [137] propose a greedy heuristic (called NMAP) for mapping computational modules onto a mesh NoC. This solution consists of two phases: First, the heuristic finds a good mapping solution X_{ik} , where a task $v_i \in V_{\text{App}}$ is mapped onto a processing element $p_k \in S_P$ through an iterative swapping of vertices which improves the communication energy consumption. Second, in order to satisfy the bandwidth constraints (as in Equation (2.3)), the heuristic calls an LP tool for solving the multi-commodity flow equations [7, 54]. As such, this approach cannot guarantee an optimum solution.

Being inspired by evolutionary computing techniques, Ascia et al. [13] set forth a multi-objective exploration framework which aims at optimizing the performance and power consumption of a mapping solution. Their approach samples the entire Pareto optimal set (i.e., the cross-product among the Φ , Ψ , and Π sets in Figure 2.2) and evaluates the performance and power consumption through a simulation-based approach.

In [170], Rhee et al. present an MILP formulation for the mapping problem which minimizes the hop-count distance between all source–destination pairs, while satisfying the communication flow and link bandwidth constraints. Toward this end, the authors assume a mesh-based topology and each PE bounded to a unique router. Thus, the binary tensorial variable $X_{ijklmn\tau}(t)$ in Equation (2.3) simplifies to X_{ik} (i.e., task $v_i \in V_{\text{App}}$ is mapped onto processing element $p_k \in S_P$). To solve the MILP formulation, the authors resort to LP solver packages and thus, cannot guarantee the solution optimality.

Along the same lines, Srinivasan and Chatha [190] propose a polynomial time heuristic algorithm called MOCA which consists of two phases. First, the algorithm uses the Fiducia–Mattheyses algorithm to construct a partition-based slicing tree (i.e., the application graph is partitioned into clusters with same number of nodes and a minimum cumulative weight of the edges crossing the partitions; this determines a graph $App(V_{App}, E_{App})$ from the set Φ), and then the tree is mapped onto the network routers. Second, using this slicing tree, a unique route is assigned to each communication trace; this is equivalent to solving Equation (2.6) while considering that the bandwidth and energy constraints are satisfied in the first part of the algorithm. Although it is reported that MOCA algorithm is faster than the NMAP, it does not guarantee the optimum solution either.

Another approach, called unified mapping, routing, and slot allocation (UMARS+), is proposed in [85] with the objective of mapping multiple QoS-constrained applications (similarly to the set $\Phi(t)$) onto an NoC architecture (set $\Pi(t)$), statically routing the communication among tiles and allocating the TDMA time-slots on network channels such that the application constraints (i.e., bandwidth, latency) are met. To determine the binary tensorial variable $X_{ijklmn\tau}(t)$ in Equation (2.3), an iterative search picks the most critical flow from the set $\Phi(t)$ (to which a path and a time slot needs to be assigned), checks the availability of the required resources, and allocates the resources such that the QoS constraints are satisfied. The bottleneck of the heuristic is represented by the path search which tries to find R_{kl} in Equation (2.6) that guarantees a certain QoS. Since the algorithm never backtracks to find whether or not another flow assignment may be used to improve the overall solution, the algorithm does not guarantee the optimality of the solution.

To exploit the heterogeneity of various models of operation (i.e., number of applications or use-cases which can be represented as in Figure 2.2), Murali et al. [136] propose an UMARS-inspired heuristic mapping which maps multiple use-cases onto the target NoC architecture while satisfying the constraints of each application. The proposed heuristic starts with an initial target architecture and refines it (i.e., it increases its size) until all applications are mapped while satisfying the

bandwidth and delay constraints (i.e., first and second constraints in Equation (2.3)).

In order to provide a thermally balanced design, Hung et al. [99] use a genetic algorithm to solve the mapping problem (i.e., find the variables X_{ik}) by encoding the IP virtualization and placement information into chromosomes. The algorithm starts with a random configuration and iterates based on the fitness of the chromosomes. The fitness function associated to each chromosome is directly proportional with the inverse of the communication cost quantified as the number of hops between the source and the destination and the communication volume.

In [49], Chou et al. propose a run time mapping algorithm consisting of two phases. First, the algorithm determines via a maze routing approach the set of neighboring resources (called near convex region) where the incoming application can be mapped. Then, a recursive heuristic is used to allocate the nodes of the incoming application communication graph (i.e., the $\Phi(t)$ set) onto the available resources (the topology is fixed to a mesh NoC and so $\Psi(t)$ denotes the PEs available at time t) together with voltage level assignment. The goal of the heuristic is to minimize the inter-node communication energy. However, the obtained solution is not optimal since the heuristic may be trapped in a local minimum.

2.3.3.2 Techniques for Power, Energy, and Thermal-based Optimization and Management

All the approaches discussed so far rely on the accuracy of estimating various metrics used during the optimization process. One of the earliest attempts of estimating the power consumption of multi-chip interconnection networks is addressed in [159]. In that paper, the authors quantify the message latency as a function of network topology, switching technique and power budget. The message latency can be substituted as an objective in Equation (2.2) with the goal of determining the network topology that satisfies a specified power budget for a certain set of applications Φ .

Several other studies analyze the power consumption of NoCs. For instance, a framework for quantifying the power consumption on switch

fabrics is presented in [210]. In [204], Wang et al. analyze the power consumption of network microarchitectures, while in [205] the impact of process technologies on network energy consumption is investigated. A network simulation environment meant to report the power consumption is described in [206].

An architectural methodology for estimating the leakage power consumption based on technology dependent variables has been proposed in [48]. In [17], a cycle accurate RTL model is used to evaluate the dynamic and leakage power consumption of NoCs. Eisley and Peh [66] propose a power analysis framework by estimating the link utilization based on message flows. All these methodologies can provide insights into formulating the optimization problem in Equation (2.2).

In terms of approaches available for optimization and management, Shin and Kim [183] consider a periodic task graph description of the application (similar to $App(V_{App}, E_{App})$ in Definition 2.1) and present an offline genetic algorithm that explores the design space in Figure 2.2. This approach statically assigns the speed of links by minimizing the energy consumption (the objective in Equation (2.2)), while guaranteeing the timing constraints. To minimize the leakage power, the authors also propose to turn links off when no communication is scheduled.

To minimize the power consumption, Sun et al. [196] propose an iterative heuristic to system-level thermal optimization of 3D-MPSoCs via task scheduling and voltage scaling. Mapping this to our framework and determine the voltage supply at which a certain PE should run at, the binary tensorial variable $X_{ijklmn\tau}(t)$ should be augmented with a new index which picks the right voltage from the given set of values in accordance with the specified design constraints. We denote by $\Xi = \{V_{dd}^o, 1 \leq o \leq \text{card}(\Xi)\}$ the set of available supply voltages at which each PE $p_k \in S_P^l$ can run. In this context, the optimization problem in Equation (2.3) aims at determining the tensorial variable $X_{ijkolmn\tau}(t)$ for which task $v_i \in V_{App}^j$ in configuration Φ^j is mapped on the PE $p_k \in S_P^l$ from the set Ψ^l . The PE $p_k \in S_P^l$ has assigned the supply voltage V_{dd}^o from set Ξ and is connected to the router $r_m \in U^n$ from set Π^n and scheduled to start at clock cycle t .

To tackle the problem of run-time power optimization, Shang et al. [178] propose a history-based dynamic voltage scaling algorithm

that adjusts the link frequencies and voltages in accordance to their utilization. To account for frequency and voltage assignment, new binary variables representing the assigned frequency and voltage should be added into the problem formulation presented in Equation (2.3).

In [179], Shang et al. consider an interconnection network as in Definition 2.2 and present a distributed run-time mechanism (i.e., the Power Herd approach), which seeks to steer the routing decision (i.e., variable $R_{kl}(t)$) and regulate the network power consumption such that it stays within the specified budget. Along the same lines, Jin et al. [103] propose a credit-based peak power control strategy that regulates each flow injection rate at the source, for both real-time and best effort traffic, such that the performance degradation and overall power consumption are minimized.

In [106], Kim et al. propose a heuristic for shutting down the links with low utilization instead of using dynamic voltage scaling techniques. The heuristic uses two hardware modules, namely the shutdown decision module and the output port selection module, to decide whether or not there is a minimal set of links providing connectivity with minimum performance degradation such that one can power down the links with the least utilization. If there is such a set, then a shortest path search algorithm is used to determine the adaptive routing strategy ($R_{kl}(t)$). If there is no minimal set ensuring the connectivity, then the heuristic powers up the link having the maximum number of appearances in the routing tables.

Li et al. [121] present a compiler-driven approach to communication link voltage management. Their strategy is to extract the data communication patterns from the application and determine the utilization patterns of the links. This information is used to select the voltage/frequency level of each link and insert control code instruction for achieving these levels.

To minimize the energy consumption and overcome the difficulties of a single clock signal distributed across the chip, Ogras et al. [150] propose a design methodology that iteratively partitions the NoC architecture into multiple voltage-frequency islands such that the performance constraints (e.g., deadlines, throughput) are satisfied. This methodology can be incorporated into our general framework of

network design by including in the binary tensorial variable $X_{ijklmn\tau}(t)$ indices accounting for each selected voltage-frequency island. Let us assume that $\Xi = \{(V_{dd}^o, V_{th}^o), 1 \leq o \leq \text{card}(\Xi)\}$ represents the set of available supply (V_{dd}) and threshold (V_{th}) voltages at which each PE $p_k \in S_P$ can run. In this context, the optimization problem formulated in Equation (2.3) aims at determining the tensorial variable $X_{ijklmn\tau}(t)$ for which task $v_i \in V_{App}^j$ in configuration Φ^j is mapped on the PE $p_k \in S_P$ from the set Ψ^l . The PE $p_k \in S_P$ has assigned the voltage tuple (V_{dd}^o, V_{th}^o) from set Ξ and it is connected to the router $r_m \in U$ from set Π^n and scheduled to start at clock cycle t .

Another metric of interest for NoC design is the thermal dissipation. In [180], Shang et al. develop an architectural thermal model for characterizing the thermal profile of the MIT RAW chip. Based on this model, the authors also propose the so-called Thermal Herd mechanism which dynamically predicts and regulates the network traffic, and uses a proactive/reactive routing protocol (variable $R_{kl}(t)$ in Equation (2.4)) to minimize the impact on performance.

Murali et al. [139] propose a convex optimization approach to thermal-aware processor frequency assignment problem. Their approach seeks to determine the operating frequencies of the processors such that the performance (i.e., total amount of instructions that can be executed in a certain time interval) is maximized and the temperature and power budget constraints are met.

2.4 Implementation Issues

2.4.1 Prototypes

To close the topology synthesis loop, an NoC design has to be implemented and accurately evaluated via prototyping. Several NoC implementation prototypes have been presented in both academia [83, 116, 198] and industry [161, 201]. For instance, Guerrier and Greiner [83] present an architectural study of on-chip interconnect networks arguing why bus-based designs are not a viable solution for future computational platforms.

In [116], Lee et al. present a comprehensive evaluation (in terms of performance area and energy consumption) of three communication

architectures (i.e., bus-based, Point-to-Point (P2P), and NoC communication architectures) targeting multimedia applications (i.e., MPEG-2 encoder). Direct measurements obtained from a Field Programmable Gate Array (FPGA) implementation of the three communication architectures show that the NoC performs better in terms of area, performance, and energy consumption.

Along the same lines, Angiolini et al. [11] compare a traditional multi-layer bus-based design (i.e., advanced microcontroller bus architecture (AMBA)) with an NoC architecture (i.e., xpipes [101]) and show that the NoC approach has higher processing speed and is more scalable.

Gratz et al. [79] present the design, implementation, and evaluation of a 4×10 2D mesh on-chip network with four VCs which provides low latencies between I/O units, L2 caches, and TRIPS processors.

To solve the wire-delay problem, Taylor et al. [198] propose the RAW microprocessor architecture which uses a scalable Instruction Set Architecture (ISA) and provides a software interface to on-chip resources.

A 32-port on-chip network implemented in 130 nm process and using a credit-based flow control is presented in [2]. Along the same lines, Bartic et al. [19] describe the implementation of a flexible FPGA-based NoC which allows to dynamically change the packet routing. In [122], Liang et al. present an adaptive SoC which supports software for application mapping and compile-time scheduled communication.

Lee et al. [117] present a hierarchical star-connected on-chip network supporting 11.2 GB/s bandwidth, operating at 1.66 GHz and implemented in 180 nm process.

To offer higher performance and lower energy consumption, Ogras and Marculescu [147] investigate the impact of adding long-range links to a 2D mesh NoC on an FPGA prototype. To avoid livelocks and deadlocks due to link insertions, a static routing strategy is implemented via lookup tables. The FPGA prototype shows that a 4×4 mesh NoC with four long-range links has higher throughput and shorter average packet latency when compared with a simple 4×4 mesh NoC.

More recently, Intel presented an 80-tile 2D mesochronous NoC implemented in 65 nm process, operating at 4 GHz and providing a

256 GB/s bisection bandwidth [201]. Targeting a 16 GFLOPS performance for LAPACK algorithms, each PE consists of two blocks of a 9-stage pipelined single precision Floating Point Multiply Accumulator (FPMAC), a 3KB single cycle instruction memory, and 2KB local data memory. Each PE is connected to a 5-port wormhole switched router with mesochronous interfaces and round robin arbitration logic. To save power, each tile is partitioned into 21 sleep regions and each router is powered down when the network traffic allows it. The 80-core NoC architecture achieves a peak performance of 1TFLOP at 1V supply voltage.

Kim et al. [105] address both architectural and circuit-level techniques for NoC implementation. Their analysis of the NoC topology confirms the arguments set forth in Equation (2.3). More precisely, for localized traffic, the energy consumption and area of mesh NoCs is larger than for customized hierarchical NoCs. This fact was also exploited in [147]. Concerning the circuit level techniques, the authors discuss the On-chip Serialization and De-serialization (SERDES) mechanism which reduces the link width and several low power strategies (i.e., low voltage swing interconnects, partial activation mechanism for crossbar switches).

Along the same lines, Shacham et al. [177] discussed the possibility of designing hybrid NoCs which combines the photonic link transmission with an electronic control layer.

While considering the topology synthesis together with floorplanning and prototyping offers the possibility of evaluating the power-performance characteristics, it is also important to test and verify the designed NoCs to ensure the targeted functionality. Several built-in NoC self-testing mechanisms have been proposed in the literature [6, 9, 28, 80, 160]. More precisely, Ahn and Kang [6] present a test scheduling algorithm which generates the test packet routing and test pattern generation with multiple test clocks for minimizing test time.

In [9], Amory et al. describe a test strategy for the control logic of routers and input FIFOs. To test the interconnected routers, a series of test vectors are applied from a single pin in the network interface and broadcasted to all routers. Next, a comparator block checks the test

responses of the routers and identifies a defective router by recording iteratively the output of each router. Finally, an IEEE 1500 compliant wrapper is used to explore the whole NoC and to provide same test stimuli for all routers. Along the same lines, Grecu et al. [80] propose a test strategy of the routers and FIFO buffers.

Petersen and Oberg [160] present a Built in Self Test (BIST) mechanism for the 2D mesh NoC switches and links at a maximum of few thousand clock cycles. The proposed test method detects and localizes the faults. Later on, this information is used to reconfigure the NoC architecture.

Regarding NoC verification, Goossens et al. [77] propose to monitor the NoC traffic based on the information about the communication requirements of the application. In [153], Ost et al. present the MAIA framework which verifies the NoC design for certain traffic patterns and load conditions. Focusing on NoC verification as well, Borriore [39] propose an ACL2 theorem prover which verifies whether the design specifications are met by the NoC design starting from a LISP description of the implementation. Finally, Salaun et al. [174] describe a model checking inspired verification methodology for asynchronous architectures.

2.4.2 Tools

To assess and evaluate the performance of NoC architectures several Computer-Aided Design (CAD) tools and design flows have been proposed by both academia [25, 52, 143, 144, 208] and industry [12, 100, 187].

One quick solution for architectural exploration and evaluating its critical performance metrics (e.g., average packet latency, throughput) is to rely on producer–communication–consumer abstraction models and build flit-accurate NoC simulators. Several simulation based architectural explorations are presented in [111, 154, 155, 205].

Among the proposed NoC simulators, Wormsim [208] offers the possibility of simulating a wide range of NoC architectures (i.e., mesh and torus topologies, deterministic and adaptive routing algorithms) for specific design parameters (e.g., channel buffer size, routing engine

delay, crossbar arbitration delay, etc.). It also supports both synthetic traffic models (e.g., uniform, hotspot, transpose) and input-trace based models of specific applications.

Targeting a simulation environment for NoCs, Nostrum [144] provides a SystemC implementation of various layers (e.g., physical, data link, network, transport, and application). The simulator allows to adjust the size of the topology (i.e., mesh, torus, ring, tree), choose the flow control scheme and routing protocol, and set the network traffic. The performance of a given design is reported in terms of average packet latency and network throughput.

Along the same lines, Nirgam [143] is a discrete event cycle accurate simulator implemented in SystemC which allows to investigate the impact of various design parameters (e.g., topology, buffer parameters, virtual channels, switching techniques, clock frequency, routing algorithms) on network design. In addition to these features, the designer can investigate the performance (i.e., average packet latency, average flit latency, average throughput for each channel) of various applications with synthetic traffic models (e.g., constant bit rate, bursty, input trace based).

Targeting primarily the communication infrastructure synthesis, COSI offers the possibility of constructing a network from a library of building blocks with specific communication protocols and comparing their performance [52, 163]. The main feature of COSI is that it allows to synthesize a general communication infrastructure based on the available library elements and their associated performance metrics.

The NetChip tool [25] offers the possibility of designing an NoC architecture by first evaluating the performance metrics for each topology in the library and then by generating SystemC simulations.

Analytical power/energy models, as well as, simulation environments for on-chip networks have been proposed in [17, 46, 48, 66, 152, 159, 179]. For instance, Orion is a dynamic power simulator which allows to obtain on-chip network power estimates for 70, 50, and 35 nm technologies based on switching energy model [152]. The Orion methodology consists of dividing the entire power estimation

function into atomic components (e.g., an atomic component contains the gate ends of NMOS and PMOS transistors of an inverter together with their drain ends), estimating the equivalent capacitances of each atomic component, identifying the on-chip network operations and estimating the switching activity based on information about these operations.

SonicsSX Smart Interconnect also offers NoC design capabilities [187]. Targeting high quality and increased video processing SonicsSX supports 2D data transactions and data busses of up to 256 bits wide. This solution is based on interleaved multichannel technology (IMT) allowing multi DRAM channel access and a peak interconnect performance of up to 16 GB per second for each port. In addition to these features, the tool offers also the possibility of desinging Globally Asynchronous Locally Synchronous (GALS) on-chip networks which can provide low power and high performance solutions.

Focusing on compatibility issues of the communication standards for NoCs, Arteris solution [12] simplifies the well-known Open System Interconnection (OSI) model to only three network layers: transaction, transport, and physical. The transaction layer connects each IP in the set S_P to the network via a Network Interface Unit (NIU). The transport layer defines the packet format and priority (depending on targeted QoS), the routing function R and the switching technique Sw . The physical layer focuses on implementation issues like raw bandwidth, matching clocks, off-chip communication, etc.

Starting from a high level description of the application and architecture domains, the iNoCs tool [100] generates a set of possible solutions depending on the chosen cost function (e.g., low power, higher performance, small area, etc.). The tool instantiate an on-chip network consisting of IP cores, switches, and network interfaces for various communication protocols, links, and frequency adapters. The adapters are meant to support multi-clock and GALS approaches which can lower the energy consumption.

As a final remark, one key issue for future CAD tools is to address the NoC design problem in all generality. For instance, the optimal topology synthesis cannot be done without floorplanning and

application mapping information. At the same time, routing and scheduling cannot be addressed without considering traffic patterns, power or energy consumption, reliability issues, etc. One step toward incorporating the network traffic into the NoC design flow is discussed via the probabilistic framework in Section 3.

3

Stochastic Perspective

3.1 Representation

Generally speaking, deterministic approaches suffer from the fact that they typically rely on worst-case type of information about the application and its behavior, while neglecting various fluctuations in the system that can occur due to changes in workload, parameter variation, or malfunctioning of hardware components. This lack of generality in capturing the information about the operating environment typically results in underestimating or overestimating the expected values of performance and energy metrics, and thus increased costs and wasted resources. For instance, a packet flow in the network and its latency along a certain path cannot be viewed as being independent of other flows since packet flows do interact causing congestion at high packet injection rates. On the other hand, in order to guarantee a certain performance level, the network design problem should consider evaluating and satisfying certain QoS metrics like average packet delay, maximum source-to-destination packet delay, packet loss rate, etc. The major difficulty in doing so is that the network traffic cannot be characterized via individual source models, but instead should reflect some

variability rooted in the complex interactions among various network components.

As expected, complete information about the application and/or architecture cannot be provided at design time. To cope with the lack of information in NoC design, the variables used in Section 2 to describe the application, architecture, performance, and various power/energy metrics need to be replaced by *probability distributions* and their corresponding moments (e.g., mean, variance). Under such a major paradigm shift, most modeling and evaluation methodologies proposed to date resort to Markovian models [128] which assume that the next state of the system can be predicted based on the current state or a few states preceding it. Since its inception, this theory became very popular and led to the prominence of Markov and queueing approaches as *de facto* tools in telecommunications and computer engineering [34, 171]. For instance, early efforts in estimating the average packet delay view the entire network as a collection of $M/M/1$ queues (see Figure 3.1) [89, 110]. The $M/M/1$ model is based on the assumption of Poisson-distributed packet arrival times and exponential distribution of packet lengths; this was meant to account for queueing and transmission delays.

There are two main problems with this approach. First, the arrival times are not exponentially distributed for many practical applications. Second, often times, the resulting Markov chains explode in the number of states needed to accurately estimate the stationary distribution of a given network. Nevertheless, stochastic approaches represent an improvement over the deterministic ones as they offer more realistic optimization techniques. Let us see how such a formalism can be constructed by giving first the definition of a probability distribution function $P_Z(z)$.

Definition 3.1. A *probability distribution* $P_Z : Q \rightarrow [0, 1]$ is a function defined over a finite set of states Q with values in the $[0, 1]$ interval which satisfies the following relation:

$$\int_{q \in Q} P_Z(z) dz = 1. \quad (3.1)$$

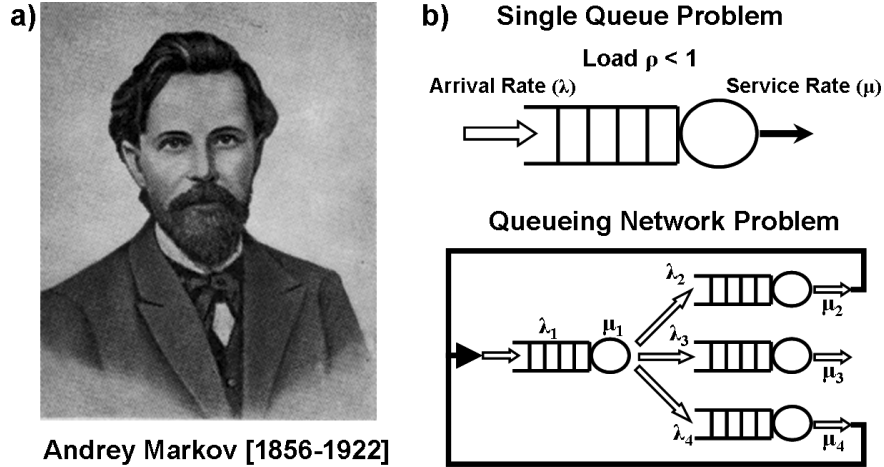


Fig. 3.1 Stochastic perspective: (a) Andrey Markov developed the Markov chain theory. (b) A single queue and a network of queues. The single queue formalism shows that jobs arrive with rate λ and stay in the queue until they are serviced with rate μ . The order in which queued jobs are served define the queueing discipline (e.g., First Come First Served (FCFS), priority based job selection, random job selection, etc.). The load ρ of the queues is assumed to be smaller than 1 to avoid congestion. The queueing network generalizes the concept of a single queue by encapsulating the relationship among multiple queues (i.e., the jobs serviced at queue 1 are distributed based on their priority to three different queues for further processing).

We go now into more details and provide next a formal definition of the application characterization graph under the Markovian assumption (i.e., exponential type distributions).

Definition 3.2. A *probabilistic application characterization graph* (PACG), $App = App(V_{App}, E_{App}, P_Q, P_\lambda, P_\mu, P_\epsilon, P_d)$, is a *directed* graph, where

- V_{App} is the set of vertices (i.e., each vertex $v_i \in V_{App}$ denotes a computational module of the application referred to as a computational task);
- $E_{App} \subset V_{App} \times V_{App}$ is the set of edges (i.e., each directed arc $e_{ij} \in E_{App}$ represents the communication from vertex v_i to vertex v_j)
 - each arc $e_{ij} \in E_{App}$ is tagged with application-specific information (e.g., probability distribution of the

packet communication volume $commvol(e_{ij})$ flowing from source v_i to destination v_j), and specific design constraints (e.g., utilization parameter of an arc $\rho(e_{ij})$, first and second moment of the latency requirements $lat(e_{ij})$);

- $P_Q : V_{App} \times V_{App} \times E_{App} \times P_Z(V_{App}) \times P_Z(E_{App}) \times R^+ \rightarrow [0,1]$ is a time dependent probability which activates (or characterizes) each transition in the PACG;
- P_λ and P_μ are the probability distributions associated to the packet generation (λ) and consumption (μ) at each vertex $v_i \in V_{App}$;
- P_ϵ is the probability distribution of the execution or computation time (ϵ) at vertex $v_i \in V_{App}$;
- $P_d(v_i)$ represents probability distribution function of the deadlines (d) for each vertex $v_i \in V_{App}$ as shown in Figure 3.2.

Similarly, a probabilistic description of the NoC architecture is as follows:

Definition 3.3. The NoC *architecture* is defined by the tuple $Arch = Arch(T(U,F), \Omega, S_P, P_r, P_{ch}, P_{st}, P_{freq})$, where the components have the following semantics:

- The network topology is viewed as a labeled graph $T(U,F)$, where the routers and channels in the network are given by the sets U and F , respectively, as follows:
 - $\forall ch_{ij} \in F$, $w(ch_{ij})$ gives the channel bandwidth;
 - $\forall r_i \in U$, $l(r_i, ch_{ij})$ gives the buffer size (depth) of channel ch_{ij} located at router r_i ;
 - $Pos(r_i)$ gives the xy coordinates of the router r_i ;
 - $P_r(r_i)$ is the probability of failure of router r_i ;
 - $P_{ch}(r_i, ch_{ij})$ is the probability of successful transmission of a packet routed at r_i via channel ch_{ij} ;
 - $P_{st}(r_i)$ gives the distribution of the packet service time (st) at the router r_i ;

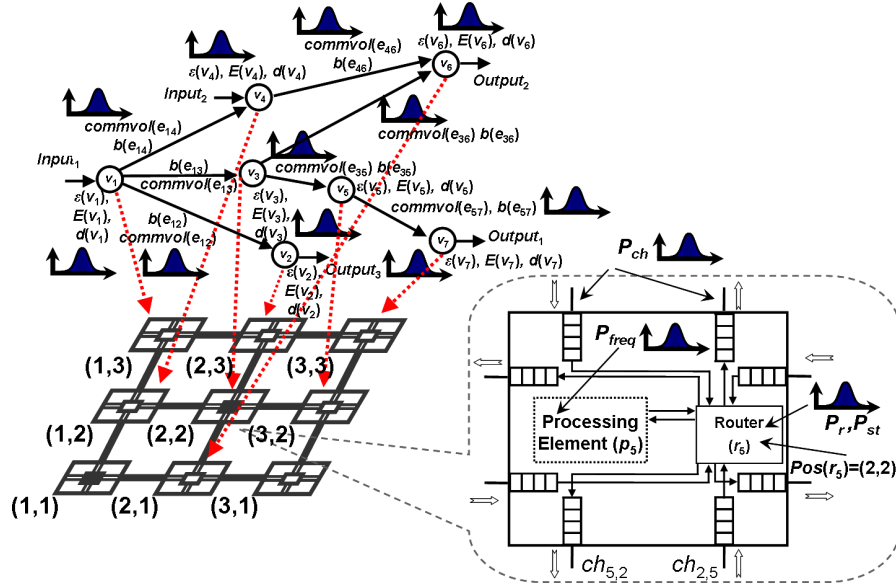


Fig. 3.2 A probabilistic perspective to NoC design: Both application and architecture parameters are modeled via a graph where vertices and arcs are characterized by random variables typically assumed to exhibit short-range dependence. For instance, $\epsilon(v_1)$ and $d(v_1)$ denotes the random variables associated with the execution time and deadline of task v_1 .

- S_P represents the set of Processing Elements (PEs);
 - the clock frequency ($freq$) of each $p_k \in S_P$ is characterized by a probability density function $P_{freq}(freq|p_k)$;
- $\Omega : S_P \times U \rightarrow \{0,1\}$ is a function that maps a PE $p_k \in S_P$ to a router $r_i \in U$ (i.e., $\Omega(p_k, r_i) = 1$).

For clarity purposes, we also provide the definition of the communication paradigm:

Definition 3.4. The *communication paradigm* is represented as

$$\mathfrak{R}\{R(v_k, r_i, ch_{ij}, r_{src}, p_{src}, r_{dst}, p_{dst}, \rho(t)), Sw\}$$

$$v_k \in V_{App}, r_i, r_{src}, r_{dst} \in U, ch_{ij} \in F, p_{src}, p_{dst} \in S_P, \quad (3.2)$$

where

- $R(v_k, r_i, ch_{ij}, r_{src}, p_{src}, r_{dst}, p_{dst}, \rho(t))$ is the routing policy¹ at router r_i for a packet generated during the execution of task v_k . The task v_k is mapped on PE p_{src} which is connected to router r_{src} . The packet is intended for a destination p_{dst} which is mapped to router r_{dst} .
 - $\rho(t): F \times t \rightarrow [0, 1]$ denotes the utilization of channels connected to the neighboring routers at time t ;
- Sw specifies the chosen packet switching technique (i.e., a protocol to forward the flit through the channel $ch_{ij} \in F$ of router $r_i \in U$ toward the router $r_j \in U$).

Similarly to the deterministic case, static routing does not require the utilization information, while adaptive routing checks the congestion of the immediate routers and picks the ones having the smallest utilization $\rho(t)$ at time t . To provide fault-tolerant communication between source–destination pairs, various routing algorithms consider also the state of the routers and links across the network. For instance, the stochastic communication protocol in [31, 65] chooses to forward the incoming packet to several neighbors to overcome the possibility of link or node failures.

3.2 Problem Formulation

The stochastic formulation of the network design problem is as follows:

$$\begin{aligned}
 & \min O(Arch(T(U, F), \Omega, S_P, P_r, P_{ch}, P_{st}, P_{freq}), M, S, R) \\
 & \text{s.t. for a given application } App(V_{App}, E_{App}, P_Q, P_\lambda, P_\mu, P_\epsilon, P_d) \\
 & \text{and architecture } Arch(T(U, F), \Omega, S_P, P_r, P_{ch}, P_{st}, P_{freq}) \\
 & \quad g_k(App, Arch, M, S, R, t) \leq h_k, \quad k = 1, \dots, N \quad (3.3)
 \end{aligned}$$

¹In the stochastic case, the routing function is defined as $R: App \times Arch \times Arch \times Arch \times Arch \times Arch \times Arch \times [0, 1] \times t \rightarrow \{0, 1\} \times t$ and considers the probability distributions associated with the correct functionality of routers and channels when determining the output channel ch_{ij} at router r_i . However, in Equation (3.9), we use a shorter form of the routing function which gives the set of paths between a source destination pair.

where the g_k functions depend on random variables characterizing the application and architecture, and h_k are the constraints that need to be satisfied. In other words, the goal here is to determine a certain NoC architecture (i.e., specific topology, channel bandwidth, buffer sizes, etc.), a mapping, scheduling and/or a routing function such that the constraints, which can represent performance (e.g., bandwidth, throughput, node-to-node latency, chip area), power/energy consumption, and/or reliability metrics, are satisfied.

The main distinction between the deterministic formulation in Equation (2.2) and the current stochastic framework is that now we have to deal with *uncertainty* about the application and/or architecture, typically captured via exponential type distributions. For instance, the transition probability from state ξ_n (e.g., *Read*) to a state ξ_m (e.g., *Produce*) from space Q (e.g., $Q = \{\textit{Produce}, \textit{Wait to write}, \textit{Read}, \dots\}$) can be written as $Pr\{S(t + \delta t) = \xi_m | S(t) = \xi_n\} = 1 - e^{-\lambda(\xi_n \rightarrow \xi_m)\delta t}$, where $\lambda(\xi_n \rightarrow \xi_m)$ is the corresponding rate. This type of transition rates can be used to build a generator matrix W for the entire system, and, if the associated Markov chain is ergodic, then the performance of any design can be evaluated at equilibrium by solving the matrix equation $W\pi = \pi$ (see the discrete time Markov chain chapter in [34]). Indeed, solving this equation gives the steady-state probability distribution π which can be further used to evaluate concrete metrics (e.g., resource utilization, power consumption, etc.) on system behavior.

3.2.1 Topology Synthesis, Mapping and Scheduling Problems

To be more precise, let us formulate the topology synthesis, mapping and scheduling problems assuming that we know the application characterization graph and PEs communicate via a shortest path routing protocol. More precisely, given the maximum number of PEs, the goal of topology synthesis and mapping problems is to determine the number of links and routers, and the configurations in which these links connect the routers such that the application performance is maximized and/or energy consumption is minimized. Due to practical

considerations driven by technology, we are limited in the maximum number of PEs (N_{PE}) that can be used.

We denote by $\Psi = \{\Psi^j(S_P^j) | 1 \leq j \leq \text{card}(\Psi), \text{card}(S_P^j) \leq N_{PE}\}$ the set of processing elements (see Figure 3.3). We also build a set of graphs $\Pi = \{T^j(U^j, F^j) | 1 \leq j \leq \text{card}(\Pi)\}$ depicting all possible topological configurations of routers and links. Similarly, we denote by $\Phi = \{App^j(V_{App}^j, E_{App}^j) | 1 \leq j \leq \text{card}(\Phi)\}$ the set of graphs consisting of all possible application task graphs obtained by clustering and/or partitioning.

Next, we define the probability $X_{ijklmn\tau}(t) \in [0, 1]$ of mapping the task $v_i \in V_{App}^j$ from configuration Φ^j , on PE $p_k \in S_P^l$ from set Ψ^l which is attached to router $r_m \in U^n$ from topological configuration Π^n and scheduled to start its execution at time t as shown in Figure 3.3. The problem can be formulated as follows:

$$\begin{aligned}
& \min c_1^T \cdot PET + c_2^T \cdot IPCT + c_3^T \cdot PEC + c_4^T \cdot CEC + LC \\
& \text{s.t. for a given set of application configurations } \Phi(t) \\
& \text{available sets of processors } \Psi(t) \text{ and topologies } \Pi(t) \\
& P\{(PET + IPCT) > lat_0\} \leq \zeta, \quad \zeta > 0 \text{ (infinitesimally small)} \\
& P\{\sum_{w=1}^{\text{card}(SD^n)} f_{w|ch_{o_1o_2}} > b_{o_1o_2}\} \leq \zeta, \quad \zeta > 0, \forall ch_{o_1o_2} \in F^n, \\
& \quad o_1, o_2 = 1, \dots, \text{card}(U^n), \quad n = 1, \dots, \text{card}(\Pi) \\
& E[\epsilon(v_{i_1}) | X_{i_1jklmn\tau}] + \text{Var}[\epsilon(v_{i_1}) | X_{i_1jklmn\tau}] + E[\tau_{\text{start}}(v_{i_1}) | X_{i_1jklmn\tau}] \\
& \quad \leq E[\tau_{\text{start}}(v_{i_2}) | X_{i_2jklmn\tau}], \text{ if } X_{i_1jklmn\tau}, X_{i_2jklmn\tau} > 0 \\
& \quad E[\tau_{\text{start}}(v_{i_1}) | X_{i_1jklmn\tau}] + lat_{i_1i_2} + E[\epsilon(v_{i_1}) | X_{i_1jk_1l_m_1n\tau}] \\
& + \text{Var}[\epsilon(v_{i_1}) | X_{i_1jk_1l_m_1n\tau}] \leq E[\tau_{\text{start}}(v_{i_2}) | X_{i_1jk_1l_m_1n\tau}, X_{i_2jk_2l_m_2n\tau}], \\
& \quad \text{if } X_{i_1jk_1l_m_1n\tau}, X_{i_2jk_2l_m_2n\tau} > 0 \forall i_1, i_2 \in \Phi^j, i_1 \neq i_2, \quad (3.4)
\end{aligned}$$

where $c_1^T, c_2^T, c_3^T, c_4^T$ are normalized cost functions (vectors) depending on probabilities $X_{ijklmn\tau}(t)$, PET denotes the processor execution time (see also Equation (3.5)), $IPCT$ represents the inter-processor communication time (Equation (3.6)), PEC defines the processor energy consumption (Equation (3.11)), CEC is the communication

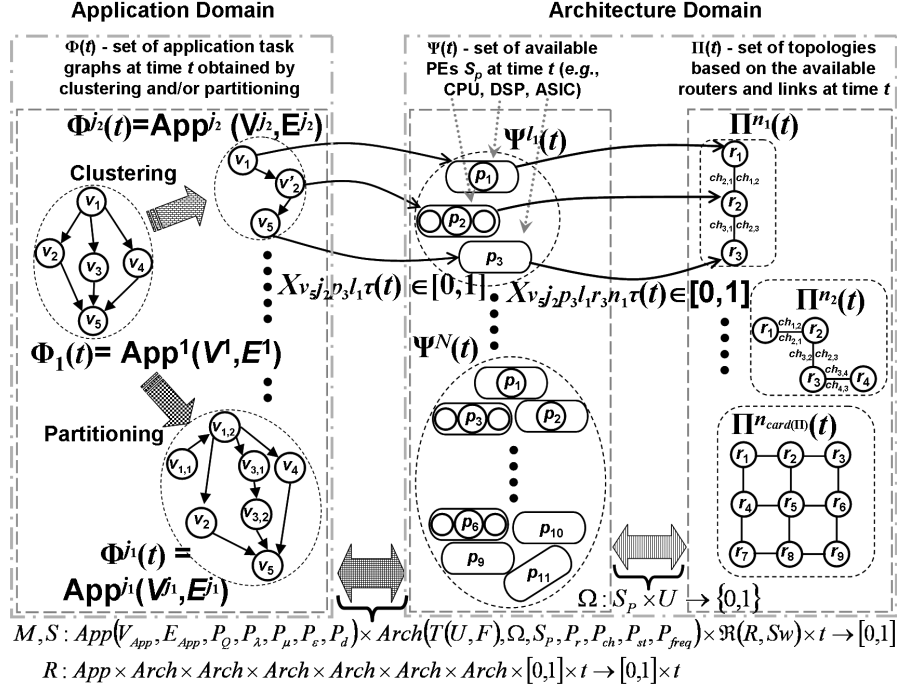


Fig. 3.3 The stochastic approach models the NoC traffic as a network of queues and replaces the deterministic binary variables (e.g., $X_{v_5 j_2 p_3 l_1 r_3 n_1 \tau}(t) \in \{0,1\}$) with probability variables (e.g., $X_{v_5 j_2 p_3 l_1 r_3 n_1 \tau}(t) \in [0,1]$). The optimum NoC design corresponds to the highest $X_{ijklmn\tau}(t) \in [0,1]$ probability.

energy consumption (Equation (3.12)) and LC is the link capacity (Equation (3.13)) assignment supporting the inter-processor traffic. We discuss next the semantics of variables, parameters, and constraints appearing in the optimization problem in Equation (3.4). Note that SD^n in the second constraint denotes the set of all source–destination pairs for the Π^n topology.

The vector of Processor Execution Time (PET) can be written as follows:

$$PET = \begin{bmatrix} \dots \\ \sum_{i=1}^{\text{card}(V_{\text{App}}^j)} \int_0^{\epsilon_{\max}} \epsilon(v_i) P_\epsilon(\epsilon(v_i), t) X_{ijklmn\tau}(t) d\epsilon \\ \dots \end{bmatrix}, \quad (3.5)$$

with $\epsilon(v_i)$ being the random variable associated to the execution-time of task $v_i \in V_{\text{App}}^j$ from graph Φ^j which runs on PE $p_k \in S_P^l$ from set Ψ^l based on a time-dependent probability $P_\epsilon(\epsilon(v_i), t)$.

The vector of Inter-Processor Communication Time (IPCT) is given by

$$IPCT = \begin{bmatrix} \dots \\ \sum_{\substack{i_1, i_2=1 \\ i_1 \neq i_2}}^{\text{card}(V_{\text{App}}^j)} \frac{E[z_{e_{i_1 i_2}} | \Psi^j(S_P^j), \Pi^n = T^n(U, F)] \cdot \text{lat}_{i_1 i_2}}{TCV} \\ \dots \end{bmatrix}, \quad (3.6)$$

where the average communication volume for a given edge $e_{i_1 i_2}$ in the application task graph $\Phi^j(V_{\text{App}}^j, E_{\text{App}}^j)$ is as follows:

$$\begin{aligned} & E[z_{e_{i_1 i_2}} | \Psi^j(S_P^j), \Pi^n = T^n(U, F)] \\ &= \int_0^\infty d\xi \int_0^{z_{\max}} dz_{e_{i_1 i_2}} z_{e_{i_1 i_2}} P_Z(z_{e_{i_1 i_2}}, \xi) X_{i_1 j k_1 l m_1 n \tau}(\xi), X_{i_2 j k_2 l m_2 n \tau}(\xi), \end{aligned} \quad (3.7)$$

and $z_{e_{i_1 i_2}}$ is a random variable associated to the communication volume between tasks v_{i_1} and v_{i_2} from configuration $\Phi^j(V_{\text{App}}^j, E_{\text{App}}^j)$ which are mapped on processors $p_{k_1}, p_{k_2} \in S_P^l$ from the set Ψ^l assigned to routers $r_{m_1}, r_{m_2} \in U^n$ from set Π^n , TCV denotes the total communication volume and is given by

$$TCV = \sum_{\substack{i_1, i_2=1 \\ i_1 \neq i_2}}^{\text{card}(V_{\text{App}}^j)} E[z_{e_{i_1 i_2}} | \Psi^j(S_P^j), \Pi^n = T^n(U, F)], \quad (3.8)$$

and $\text{lat}_{i_1 i_2}$ is the communication latency (i.e., the amount of time from the moment when a packet is generated at the source, until the last flit of the packet reaches the final destination) between two mapped and scheduled tasks v_{i_1} and v_{i_2} from configuration $\Phi^j(V_{\text{App}}^j, E_{\text{App}}^j)$. The latency $\text{lat}_{i_1 i_2}$, derived under the $M/G/1$ assumptions and a First Come First Served (FCFS) policy at the router level, is given by

$$\begin{aligned} \text{lat}_{i_1 i_2} &= E[ms | X_{i_1 j k_1 l m_1 n \tau}(t)] - 1 \\ &+ \sum_{o_1, o_2=1}^{\text{card}(R(X_{i_1 j k_1 l m_1 n \tau}(t), X_{i_2 j k_2 l m_2 n \tau}(t)))} (W_{o_1 o_2} + E[st]), \end{aligned} \quad (3.9)$$

where ms is a random variable associated with the size of the message generated due to task v_{i_1} at source p_{k_1} , $W_{o_1o_2}$ is the average waiting time for messages traversing channel $ch_{o_1o_2} \in F^n$ from the set of paths returned by the routing function $R(X_{i_1j k_1 l m_1 n \tau}(t), X_{i_2j k_2 l m_2 n \tau}(t))$. Note that, for the sake of simplicity, we use a shorter notation for the routing function $R(X_{i_1j k_1 l m_1 n \tau}(t), X_{i_2j k_2 l m_2 n \tau}(t))$ which gives the set of paths between a source $X_{i_1j k_1 l m_1 n \tau}(t)$ and a destination $X_{i_2j k_2 l m_2 n \tau}(t)$. For the $M/G/1$ model, the average waiting time is given by the following formula:

$$W_{o_1o_2} = \frac{E[\lambda_{o_1o_2} | X_{i_1j k_1 l m_1 n \tau}(t), X_{i_2j k_2 l m_2 n \tau}(t)] \cdot (E[st]^2 + Var(st)^2)}{2 \cdot (1 - E[\lambda_{o_1o_2} | X_{i_1j k_1 l m_1 n \tau}(t), X_{i_2j k_2 l m_2 n \tau}(t)] E[st])}, \quad (3.10)$$

where $\lambda_{o_1o_2}$ is the random variable associated to the arrival process at channel $ch_{o_1o_2} \in F^n$, st is the service time random variable ($E[st]$ and $Var(st)$ are the average and variance of the service time, respectively).

The PEC cost is also expressed as a vector, where each element represents the processor energy consumption of each mapped application graph on a set of PEs and a topological configuration, as follows:

$$PEC = \begin{bmatrix} \dots \\ \sum_{i=1}^{\text{card}(V_{App}^j)} \int_0^{\epsilon_{max}} E(v_i) X_{ijklmn\tau}(\xi) d\xi \\ \dots \end{bmatrix}, \quad (3.11)$$

where $E(v_i)$ is the energy consumed for executing task v_i from configuration $\Phi^j(V_{App}^j, E_{App}^j)$ which is mapped on $p_k \in S_P^l$ from set Ψ^l .

Similarly, the CEC term from Equation (3.4) is formulated as a vector whose elements encapsulate the communication energy consumption of a mapped application task graph (e.g., $\Phi^j(V_{App}^j, E_{App}^j)$) onto a set of PEs and a topological configuration:

$$CEC = \begin{bmatrix} \dots \\ \sum_{i_1, i_2=1}^{\text{card}(V_{App}^j)} \int_0^{z_{max}} z_{e_{i_1 i_2}} P_Z(z_{e_{i_1 i_2}}) E_{e_{i_1 i_2}}^{bit} X_{i_1j k_1 l m_1 n \tau} X_{i_2j k_2 l m_2 n \tau} dz_{e_{i_1 i_2}} \\ \dots \end{bmatrix}, \quad (3.12)$$

where $z_{e_{i_1 i_2}}$ is a random variable associated to the communication volume (characterized by a probability distribution $P_Z(z_{e_{i_1 i_2}})$) between tasks v_{i_1} and v_{i_2} from configuration $\Phi^j(V_{\text{App}}^j, E_{\text{App}}^j)$ which are mapped on processors $p_{k_1}, p_{k_2} \in S_P^l$ from set Ψ^l assigned to routers $r_{m_1}, r_{m_2} \in U^n$ from set Π^n and $E_{e_{i_1 i_2}}^{\text{bit}} = E(X_{i_1 j k_1 l m_1 n \tau}(t), X_{i_2 j k_2 l m_2 n \tau}(t))$ is the average energy consumption for sending one bit between vertices v_{i_1} and v_{i_2} .

Finally, the vector of link capacity assignment term can be written as follows:

$$LC = \begin{bmatrix} \dots \\ \sum_{o_1, o_2=1}^{\text{card}(U^n)} c_{o_1 o_2} \cdot E[Y_{o_1 o_2} - E[f_{o_1 o_2} | Y_{o_1 o_2}]] \\ \dots \end{bmatrix}, \quad (3.13)$$

where $Y_{o_1 o_2}$ is a random variable associated with the link capacity, $f_{o_1 o_2}$ represents the amount of data that link $ch_{o_1 o_2} \in F$ has to sustain, and $c_{o_1 o_2}$ is a design cost (e.g., area) associated with the implementation of a link of capacity $Y_{o_1 o_2}$.

The first constraint in Equation (3.4) states that the probability of having a total average execution time for the application (which is approximated as the sum between the average processor execution time and the average inter-processor execution time) to be larger than a predefined value lat_0 must not exceed a certain QoS parameter ζ , which in general is chosen to be very small. For instance, if the designer targets guaranteed service, then the value of ζ should be very small (e.g., 10^{-10}) but this value should be carefully chosen as may impact the overall runtime of the optimization problem in Equation (3.4)). Alternatively, if a best effort service is targeted then a higher ζ value in the order of 10^{-3} can be chosen.

Similarly, the probability that the flow $\sum_{w=1}^{\text{card}(SD^n)} f_w | ch_{o_1 o_2}$ for each link $ch_{o_1 o_2}$ to outrun a certain bandwidth $b_{o_1 o_2}$ must be under the limits defined by the QoS parameter ζ .

The third and fourth constraints in Equation (3.4) ensure the validity of the mapped and scheduled application (i.e., whether or not the data dependencies are satisfied to avoid resource conflicts). More precisely, if the computational tasks v_{i_1} and v_{i_2} from configuration

$\Phi^j(V_{\text{App}}^j, E_{\text{App}}^j)$ are mapped on the same processor $p_k \in S_P^l$ from the set Ψ^l (i.e., $X_{i_1 j k l m n \tau}(t), X_{i_2 j k l m n \tau}(t) > 0$) and v_{i_2} is *dependent* on the completion of task v_{i_1} (i.e., $E[\tau_{\text{start}}(v_{i_2})|X_{i_1 j k l m n \tau}, X_{i_2 j k l m n \tau}]$), then the stochastic program in Equation (3.4) must satisfy the following relation

$$E[\epsilon(v_{i_1})|X_{i_1 j k l m n \tau}(t)] + \text{Var}[\epsilon(v_{i_1})|X_{i_1 j k l m n \tau}(t)] + E[\tau_{\text{start}}(v_{i_1})|X_{i_1 j k l m n \tau}] \leq E[\tau_{\text{start}}(v_{i_2})|X_{i_2 j k l m n \tau}], \quad (3.14)$$

which shows that the two computational tasks cannot be executed by processor $p_k \in S_P$ at the same time. The start time of task v_{i_1} is given by the following relation:

$$E[\tau_{\text{start}}(v_{i_1})|X_{i_1 j k l m n \tau}] = \int_0^{\text{lat}_0} \tau_{\text{start}}(v_{i_1}) P_{\tau}(\tau_{\text{start}}(v_{i_1})) X_{i_1 j k l m n \tau} d\tau. \quad (3.15)$$

The average and variance of the execution time of task v_{i_1} are given by the following expressions:

$$\begin{aligned} E[\epsilon(v_{i_1})|X_{i_1 j k l m n \tau}(t)] &= \int_0^{\epsilon_{\text{max}}} \epsilon(v_{i_1}) P_{\epsilon}(\epsilon(v_{i_1}), t) X_{i_1 j k l m n \tau}(t) d\epsilon \\ \text{Var}[\epsilon(v_{i_1})|X_{i_1 j k l m n \tau}(t)] &= \int_0^{\epsilon_{\text{max}}} (\epsilon(v_{i_1}) - E[\epsilon(v_{i_1})])^2 P_{\epsilon}(\epsilon(v_{i_1}), t) X_{i_1 j k l m n \tau}(t) d\epsilon, \end{aligned} \quad (3.16)$$

while the expected start time of task v_{i_2} is as follows:

$$E[\tau_{\text{start}}(v_{i_2})|X_{i_2 j k l m n \tau}] = \int_0^{\text{lat}_0} \tau_{\text{start}}(v_{i_2}) P_{\tau}(\tau_{\text{start}}(v_{i_2})) X_{i_2 j k l m n \tau} d\tau. \quad (3.17)$$

If the two computational tasks do not depend on each other, then a specific scheduling policy (e.g., earliest deadline first, shortest job first, etc.) can be enforced by the designer. However, in our mathematical formulation, we have considered that each task comes with a specified start time and consequently it has to satisfy this constraint.

Similarly, the fourth constraint in Equation (3.4) must be satisfied if the computational tasks v_{i_1} and v_{i_2} from configuration $\Phi^j(V_{\text{App}}^j, E_{\text{App}}^j)$

are mapped on two different processors $p_{k_1}, p_{k_2} \in S_P^l$ from set Ψ^l assigned to routers $r_{m_1}, r_{m_2} \in U^n$ from set Π^n and the task v_{i_2} depends on the completion of task v_{i_1} , where $lat_{i_1 i_2}$ is the average communication latency between the two mapped vertices of graph $\Phi^j(V_{App}^j, E_{App}^j)$.

Most of the stochastic models used for performance evaluation rely on Markovian assumptions and can be categorized in few classes of mathematical tools like discrete or continuous Markov chains [130] (including stochastic Petri nets (SPN), stochastic automata networks (SAN)), queueing theory (e.g., queueing networks) [34, 171]. In the next section, we review the main proposed approaches that employ the above mentioned mathematical formalism.

3.3 Literature Survey

3.3.1 Communication Infrastructure

A fundamental problem in communication infrastructure synthesis is finding the right buffer assignment that maximizes the performance of the network and minimizes the power consumption. Consequently, we describe next some work on performance modeling of wormhole switching technique which lies at the basis of NoC design [3, 5, 56].

Dally [56] considers the problem of constructing wire efficient networks by analyzing the impact of network dimension in k -ary n -cubes interconnection architectures under the assumption of constant wire bisection, wormhole switching, dimensional ordered routing scheme, and fix channel delay. To derive the network throughput, the author assumes that packets are injected from each node according to a Poisson distribution and follow a uniform traffic model. The main finding of this analysis is that low dimensional networks (e.g., torus) with wide communication channels offer higher performance when compared to high dimensional networks with same bisection width.

In contrast, Agarwal investigate the network latency in k -ary n -cubes interconnection architectures under wormhole switching technique, while taking into account both switch and wire delays [5]. Similarly to Equation (3.9), the latency is proportional with the size of the packet and the delay encountered in traversing the intermediate

nodes between a source and a destination. Note that the waiting time in queues is modeled as an $M/G/1$ queueing system.

In [3], Adve and Vernon model a wormhole k -ary n -cube network as a closed queueing system and use mean value analysis to determine the average packet latency under deterministic routing and arbitrary source–destination probability distributions. The major drawback of this analysis is that it is applicable only to networks with infinite buffers; this is unrealistic for NoCs where small buffers are desirable due to tight area and power constraints.

One of the early work on performance analysis of wormhole switching technique in k -ary n -cubes networks is presented in [63]. The objective is to estimate the average packet latency and channel waiting time. The authors assume that the packets in queues are served in a FCFS manner. Similarly to Equation (3.9), the authors view the architecture as a network of queues, where each channel is characterized by an $M/G/1$ model.

Along the same line, Hu and Kleinrock [97] propose a finite size buffer model for wormhole routing and deterministic deadlock free routing schemes in arbitrary topologies. The goal is to estimate the output link contention delay and buffer queueing delay. To better characterize the blocking situations due to finite size buffers and estimate the buffer queueing delay, the authors characterize each buffer via an $M/G/1/K$ model with finite capacity (see Equation (3.10)). Besides the assumption of a deadlock free routing, the authors also assume that the buffers at destination are infinite and the arrival process and size distribution of packets are Poisson distributed.

A performance model for wormhole switching technique in 2D torus network, together with adaptive routing and finite size buffers, is presented in [51]. In that paper, the authors model each link as an $M/G/1$ queue with multiple classes of flow (described in [197]) and present a closed form formula for quantifying the mean latency time.

Addressing the buffer sizing problem for NoCs, Hu et al. [96] present an algorithm that allocates resources within a budget such that the average packet latency (which can be obtained from Equation (3.6) by assuming a mesh NoC architecture for an already mapped application) is minimized for deterministic and oblivious routing approaches. The

algorithm is based on the $M/M/1/K$ model described in [89] and adds more buffering resources only to the highly utilized channels. However, this approach does not guarantee an optimal configuration with the smallest average packet latency.

Along the same lines, an $M/G/1/K$ approach (see [197] for mathematical details) is used in [148] to model the on-chip routers under wormhole switching technique, deterministic routing with arbitrary size messages and finite buffers. The authors provide a mathematical framework for estimating the average packet latency, throughput, utilization of buffers, and average latency per router under Markovian assumptions.

Many applications, however, generate bursty traffic for which buffer size is hard to be predicted; consequently, the problem of buffer sizing requires more advanced models. Varatkar and Marculescu addressed this problem by identifying self-similar effects in on-chip multimedia traffic [202]. This means that simple models based on Poisson modeling are inaccurate in this context. Consequently, the problem formulation described in Equation (3.4) should be adapted to include some slack variables to ensure that the deadlines are met and the network traffic does not fall in the congestion regime. Moreover, the long-range dependency makes the statistical analysis of communication traces impractical and unrealistic, as the widely used confidence interval method relies on the assumption that the data is normally distributed; this is clearly not always true. Motivated in part by these facts, the work in [188] proposes an empirical model for capturing the spatiotemporal characteristics of the network traffic. The authors show how their approach can be used to generate synthetic traces, but no definite algorithm for optimizing the architecture is discussed.

Regarding the link capacity allocation, Guz et al. [84] present an algorithm for sizing the virtual channels in NoC under QoS constraints and wormhole routing. Assuming a certain NoC topology, a static routing approach, fixed link capacities and communication requirements, their first analytical step is to estimate source-to-destination delay via an $M/M/1$ model. In contrast to Equation (3.4), where the inter-processor communication is minimized such that the bandwidth constraints are satisfied, the next step in the algorithm is to reduce, via a

greedy approach, the capacity of each link such that the total allocated link capacity is minimized and the delay constraints are met. Due to the nature of greedy searching, several optimal configurations can be missed and the optimum solution cannot be guaranteed.

To leverage the level of network congestion (i.e., the head-of-line blocking at router-level), Huang et al. [98] propose a queueing-based algorithm for allocating the Virtual Channels (VCs). The authors assume that the network topology ($T(U, F)$), the mapping (M) of a PACG onto NoC architecture and the static routing (R) functions are given and seek to find the number of VCs allocated to each link. To evaluate the level of congestion, a queueing network consisting of $M/M/1$ models is used. The proposed algorithm proceeds in a greedy fashion by increasing the number of VCs for each link as long as the average packet latency is minimized. Nevertheless, the proposed algorithm does not guarantee the optimal VC allocation.

Along the same lines, Faruque and Henkel [72] propose a VC allocation algorithm based on Poisson traffic assumptions. The algorithm first maps the ACG onto an NoC architecture via an ant colony optimization approach. Based on the assumption that packet arrival follows a Poisson distribution, the next step is to estimate VC size for each output port of a router. Again, the proposed approach does not guarantee the optimality. The main drawback of the proposed methodology is that it relies on Markovian assumptions for multimedia applications which, in fact, is contradicted by the experimental results in [202].

3.3.2 Communication Paradigm

Addressing the problem of mapping and scheduling of new functionality on distributed embedded systems, Pop et al. [164] propose a heuristic consisting of two steps: First, the algorithm searches for a mapping (i.e., the probability $X_{ijklmn\tau}(t)$ is obtained via an iterative search) with a valid schedule satisfying the prescribed hard deadlines. Second, using simulated annealing (or an iterative approach) the algorithm minimizes the objective function which consists of a weighted linear combination of the available slack sizes and penalty for not meeting the expected processor time and bus bandwidth. At the same time, the

newly obtained mapping should not affect the already running applications. This approach is considered to be applicable to a design with nonpreemptive static scheduling and a TDMA-based communication scheme. Although the simulated annealing approach produces nearly optimal results, it requires very large computational times which is not realistic for design space exploration. The second approach, based on an iterative heuristic, is faster at providing good results, but there is no way to guarantee optimality.

To efficiently utilize the bandwidth offered by NoC architecture, a predictive closed loop flow control mechanism is presented in [149]. Based on an ON/OFF traffic modeling of the sources (and in the spirit of the second constraint in Equation (3.4)), the algorithm predicts the possible congestion in the network and then controls the packet injection rates at source-level such that a certain network throughput is maintained. Note that the authors assume that the application is mapped and the decision on flow-control is taken at runtime.

Concerning the thermal issues, Coskun et al. [55] propose two heuristics for task scheduling onto MPSoCs while seeking to optimize the thermal profile with minimal impact on performance. The first approach schedules the incoming task to the coolest processor (Coolest) or to the processors that have idle neighbors (Coolest-FLP). The second approach (Adaptive-Random) updates continuously the probabilities of sending the workload to the processors at each time step based on the temperature history analysis. The processor probabilities of receiving new tasks are incremented or decremented based on whether or not their temperature exceeds the prescribed threshold. More precisely, the processors for which temperature exceeds a certain threshold have zero probability of receiving new tasks. The amount by which the probability is increased is inversely proportional with the average temperature below the threshold. However, this does not guarantee an optimal schedule.

To overcome the problem of transient link failures, Manolache et al. [125] propose a heuristic strategy that statically assigns multiple copies of each message to be sent on the network links. The core of the heuristic is based on the message arrival probability which is defined as the expected fraction of successfully transmitted messages for a pair

of communicating tasks. The algorithm identifies for each pair of communicating tasks a set of candidate communication supports (i.e., sets of consecutive links between the source and the destination) such that a lower bound on the message arrival probability is satisfied. Among all the candidate communication supports, the algorithm chooses the one that offers the minimum response time; this is similar to the first constraint in Equation (3.4). If no solution is found, then the algorithm terminates with an error message stating that the application cannot be scheduled. This approach can be improved if the formalism described in Equation (3.4) is used for scheduling the application onto a reconfigurable architecture.

The process of shrinking transistor sizes, reducing the interconnect features and increasing the operating frequencies of CMOS circuits leads to higher soft-error rates [53, 185] and an increasing number of timing violations [182]. Consequently, the design of a fault-tolerant communication protocol is proposed in [31], where Bogdan et al. quantify the efficiency of information spreading through the network via a coverage metric. The approach is based on a probabilistic broadcasting scheme where each node randomly sends the newly received packets to its neighbors. A continuous-time Markov chain is used to model the information diffusion over a mesh network, and based on a master equation, the authors propose a mean-field analysis describing the number of nodes that receive the disseminated message (i.e., node coverage). However, besides node coverage, the NoC design and optimization requires specific information about the number of communication rounds needed for a message to propagate between a source–destination pair of nodes. To address this issue, Bogdan et al. [33] estimate the hitting time for the stochastic communication. The analysis is based on modeling the process of information diffusion as a collection of branching random walks. Of course, the number of random walks associated with packet forwarding can increase or decrease at each node based on the link successful transmission or the rate of nodes routing failure.

To sustain the network functionality and to lessen the impact of faults on the running application, a probabilistic broadcasting scheme as in [31, 65], together with a reconfigurable approach, is presented in [166]. Puente et al. [166] propose the use of conventional hardware

to dynamically detect errors in k -ary n -cube topologies. However, the proposed approach suffer from the fact that packets that are in transit through failing nodes are lost. Another drawback of this approach is that it assumes that only permanent failures can affect the routers. Obviously this is not always the case, since routers can also be affected by transient errors. For instance, transient errors can happen during an emergency state when routers try to communicate their state, thus affecting the information about the network connectivity and resources. Consequently, an optimal routing protocol should not only consider the bandwidth and energy constraints as in Equation (3.4), but also consider the signal-to-noise ratio induced by the physical substrate errors.

3.3.3 Power, Energy, and Thermal-based Optimization and Management Issues

The work in [186] tackles the power management problem for NoCs and proposes a stochastic control loop model which is meant to overcome the inherent challenges of the network communication and reliability. The authors build a queueing network model for the NoC architecture, where each core is modeled as a renewal stochastic process. The length of the idle time and the transition time between cores active, idle, and sleep states are distributed according to a Pareto and a uniform distribution, respectively, while the workload is modeled via an exponential distribution. The stochastic control loop model consists of the NoC architecture under investigation, the power controller and the estimator. The role of the power controller is to set certain commands to the core influencing its performance and energy consumption. The role of the estimator is to observe the requests coming into the core queue from network traffic and the state of the core. Based on this information, the estimator computes the parameters of the power management policy. To provide a fast closed-loop power management, the authors use the Lagrangian theory to obtain the dual problem instead of solving the LP formulation.

To estimate the energy consumption of a packet-switched NoC architecture, Chan et al. [46] employ a linear regression method to model the relationship between NoC communication events and

energy consumption (obtained from a gate-level simulator). Consequently, each variable corresponds to the energy consumption of an NoC component (e.g., arbiter, crossbar). To estimate the energy consumption of each component, a macro-model of the NoC components is built based on the technology libraries and traffic patterns. Nevertheless, the macro-modeling is highly dependent on the considered traffic patterns since they can exercise differently various circuit parts. For instance, the estimated energy consumption values are highly affected by the packet injection and consumption rates.

To address the problem of low energy hard real-time task scheduling, Gruian [81] uses stochastic data to derive efficient voltage assignments. The author assumes independent tasks running on a single processor characterized by a variable speed which can be adjusted at run-time. To obtain the voltage assignment for each task, the author uses the probability distribution of the execution pattern for a given task. The proposed offline task scheduling algorithm works iteratively on each task and based on exact timing analysis, derives offline voltage scaling factors. In contrast, the online approach uses the variations in the execution length of various tasks to stretch some of the tasks (i.e., when there are no tasks in the queue, the executing tasks can be stretched and run at a lower voltage until the arrival of a new task) and provide lower energy consumption. Based on task priority, the online algorithm distributes the available processor times resulted from the high priority tasks. A major drawback of the proposed approaches is that the authors assume independent task sets.

To support the communication among multiple voltage-frequency NoC islands, Ogras et al. [151] present a feedback-based control methodology for controlling the speed of each domain in the presence of parameter and workload variations and to reduce the communication energy consumption. A state-space model based on the utilization of the inter-domain queues is built and analyzed to see whether all or only a subset of queues can be controlled. Nevertheless, for any subset of controllable queues, the proposed approach can control only well-behaved workloads. As such, it is necessary to identify which subset of controllable queues brings more benefits and also to dynamically switch from one set of controllable queues to another in response to network

traffic variation. To account for the speed at which each PE is running, the probabilities $X_{ijklmn\tau}(t)$ in Equation (3.4) have to be augmented with indices representing the assigned supply and threshold voltages. Then, the role of the controller is to find the voltages which ensure that the utilization of the queues is within some margins.

One picture that clearly emerges from this analysis is that probabilistic approaches can better describe the distributed computational processes mapped onto an NoC architecture. At the same time, it is unlikely that with the increasing complexity of the emergent applications the OS can easily cluster or partition and estimate their timing constraints. Consequently, despite their inherent limitations due to the Markovian assumption, stochastic approaches are more promising for mapping and scheduling decisions. Moreover, with the technology advances and novel devices at nanoscale, we can build reconfigurable architectures with stochastically evolvable topologies and routing protocols. To overcome the main limitations of Markovian assumptions, we introduce next a statistical physics inspired traffic analysis and review the major approaches proposed in the literature concerning the complexity of networked architectures.

4

Statistical Physics Perspective

4.1 Representation

In general, given an arbitrary network topology, the routing function between any source–destination pair, and the maximum available buffer space, predicting the point of maximum information transfer and/or the point of network congestion is a difficult problem. This is mainly because the network behavior under various traffic conditions follows a nonequilibrium process. Consequently, it is hard to suggest how to design the network such that congestion is avoided.

Statistical physics can help us describe any network through a wave-function. However, this is not an easy task since it needs to solve the Schrodinger equation [40, 129]. However, an important observation can be made: due to the inherent uncertainty in the system behavior, one cannot completely know the network wave function as a function of time and thus, need to consider many possible network configurations. Statistical physics and information theory can help us find these time-dependent wave functions and the most likely network states that can be further used as a foundation for network optimization.

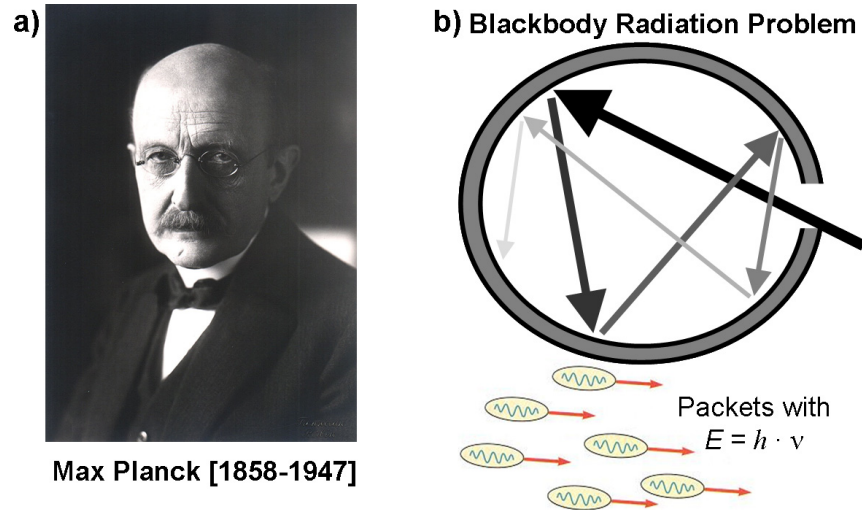


Fig. 4.1 Statistical physics perspective: (a) Max Planck put forth the revolutionary hypothesis that electromagnetic energy is emitted in a quantized form $E = h\nu$ and solved the black-body radiation problem. (b) A black-body is an ideal system that absorbs incident electromagnetic radiation. The black-body system is considered isolated and thus has a definite energy. Planck postulated that photons absorbed at one frequency may be emitted at another frequency, but in different numbers and at nonzero temperature.

The main difference between the statistical physics framework and the previously discussed stochastic approaches is that the first one does not make any Markovian assumption about the dynamical processes involved in network behavior; instead, it tries to model the interactions among various components, while taking into consideration the long-term memory effects (see Figure 4.1). Nevertheless, the statistical physics and information theory inspired approaches to network optimization are still in their infancy. Thus, we limit ourselves to describe next some of the statistical physics inspired approaches proposed to date, hoping that this will stimulate more research in future years.

4.2 Literature Survey

Traditional queueing or Markov approaches used for modeling and analyzing the network performance are primarily based on Poisson traffic. While being attractive for its analytical tractability in terms of closed

form performance metrics (e.g., mean and variance of queue length, waiting time, length of busy/idle periods, packet loss, etc.), the Poisson traffic model fails to capture some important network characteristics like self-similarity or long-range dependence [158]. In fact, it has been shown that transport in many complex systems (e.g., Internet, biological networks, etc.) can be defined as a stochastic process whose auto-correlation function decays as a power law [124, 168]. This phenomenon is called long-range dependence.

Simply speaking, the main difficulty entailed by the ubiquity of long-range memory properties is that their autocorrelation function decays much slower than the exponential Poisson decay. For instance, the network measurements done at Bellcore on Internet traffic exhibit self-similarity characteristics [158]. Similar conclusions were reached by analyzing the bursty traffic between on-chip modules in an MPEG-2 video application [202]. In conclusion, the network traffic characteristics (and in particular, self-similarity) are crucial for performance analysis, as increased burstiness is conducive to inefficient resource utilization. Besides performance, various QoS parameters, such as available bandwidth, average packet latency, packet loss probability, can be highly affected too. To overcome these difficulties, any performance analysis or buffer assignment algorithm needs to account for network traffic characteristics.

Along these lines, Bogdan and Marculescu [32] propose a paradigm shift in NoC design based on the analogy between the network and thermodynamic gas behaviors. This becomes possible based on the observation that each buffer of the NoC, at any point in time, can be characterized by a particular energy level. More precisely, the main idea is that packets in the network move from one node to another in a manner that is similar to particles moving in a Bose gas and migrating between various energy levels as a result of temperature variations. The approach in [32] models the buffers in the NoC architecture as competitive systems and constructs a Virtual Random Growing Network (VRGN) corresponding to the information flow through the network. More precisely, each node in the VRGN corresponds to a buffer in the NoC architecture. The application mapped onto the NoC architectures dictates not only the network traffic, but also the *IN* and *OUT* degree

of each node in the VRGN. Based on the fitness model proposed in [29], Bogdan and Marculescu assign two energy levels corresponding to the incoming and outgoing packets at each node in the VRGN.

To make things clearer, a schematic representation of the VRGN evolution is given in Figure 4.2. The packets in the real NoC correspond

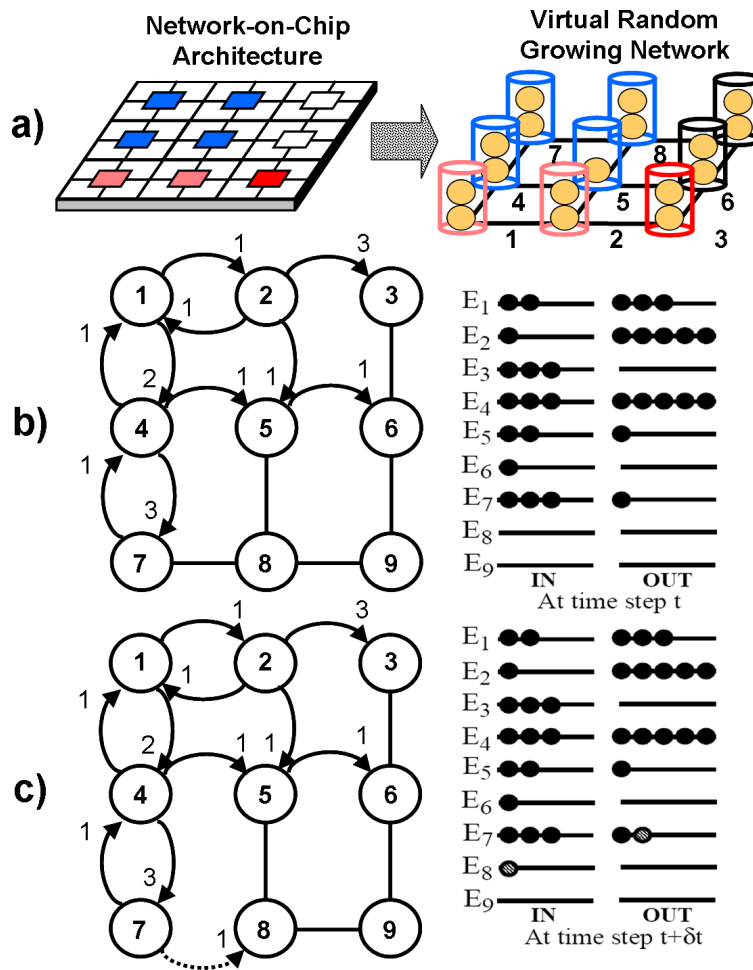


Fig. 4.2 (a) NoC architecture seen as a thermodynamic gas where a *Virtual Growing Random Network* (VRGN) is used to model the information flow through the network and the competitiveness among packets for buffer resources. (b) Snapshot of the VRGN at time t and the corresponding *IN* and *OUT* energy levels. (c) The state of the VRGN at time $t + \delta t$ when a packet is routed from node 7 to node 8 increasing the *OUT* and *IN* degree of nodes 7 and 8, respectively. (Figure taken from Bogdan and Marculescu [32].)

to particles on various energy levels in a thermodynamic system. The configuration of various energy levels shown in Figure 4.2(b) gives a snapshot of VRGN state. The numbers on the directed links of the left most graphs indicate the number of packets exchanged between those nodes. For example, in Figure 4.2(b), the *IN* and *OUT* degree of node 7 is 3 and 1, respectively. Within the infinitesimal time interval $(t, t + \delta t)$, node 7 may send one packet to node 8 (i.e., discovering this node) and thus increasing its *OUT* degree. This is illustrated in Figure 4.2(c) by a change in the energy level of E_7 .

Buffers in real NoCs are finite and therefore packet interaction causes congestion. As such, besides the growing process mentioned earlier, a link rewiring process takes place in the VRGN. The main claim in [32] is that the NoC traffic follows *mixed statistics*, i.e., Bose–Einstein (BE) statistics for the growing process before criticality, and Fermi–Dirac (FD) statistics for the reverse process. In terms of quantum-like behavior, in the bosonic phase each energy level can accommodate infinitely many particles, while during the fermionic phase each energy level is constrained. The analysis shows that the buffer occupancy follows a power law distribution.

Recently, Cohen et al. [50] propose a statistical approach to the traffic load distribution which determines how much capacity provisioning is required to accommodate 90% of the traffic patterns. The main assumption is that the traffic load distributions can be approximated by Gauss distributions. However, the proposed approach requires to first estimate the traffic load distributions; this is a difficult task, given the complex behavior exhibited by network traffic.

Along these lines and in contrast to the topology synthesis solutions discussed in Section 2.3.1.1, Teuscher proposes a randomized heuristic for constructing a nature-inspired 3D network topology with bounded connectivity for each node; this approach should benefit from the emerging self-assembling molecular electronics [199]. The approach starts from the Watts–Strogatz model of small-worlds [207] and varies a set of parameters (e.g., connectivity, number of switches, distribution of long versus short-range connections, etc.) such that the communication characteristics are optimized and robustness against link and/or node failures is ensured. Although the proposed approach can provide

insights into designing cheaper self-assembling and low energy consumption NoC architectures, it must rely on specific mapping and scheduling strategies, as well as a fault-tolerant communication protocol (such as the one presented in [31]) to lessen the design complexity.

Varatkar et al. [203] use robust statistical signal theory for designing robust and energy efficient sensor NoCs (SNoC). Similarly to the stochastic communication protocol in [31], the nodes in the SNoC communicate with each other stochastically. Due to the presence of nanometer nonidealities (e.g., particle hits, process variation), a robust low-power computation is achieved via an estimation approach. The authors assume that the sensors output errors are distributed according to a Gaussian distribution. A so-called fusion node operating at a much lower frequency than the actual sensors, collects the computed results from all (statistically similar) sensors and performs a robust estimation.

Most of the above approaches help us understand the interactions among network components, explain their functionality and propose algorithms for solving a concrete problem: how we should design the networks of the future? At the same time, biological systems prove to be highly efficient and robust to failures through an evolutionary process. One common feature to most of the complex natural and biological systems is the power-law signature. Inspired from statistical physics concepts, Carlson and Doyle [44] propose the *highly optimized tolerance* (HOT) approach which aims to relate the complexity and power law mechanism observed in nature with the effort of designing high performance engineered systems. More precisely, the authors claim that the inputs of any complex system may lead to outputs obeying power laws as a consequence of a global collective optimization process aimed at high efficiency, performance, and robustness. Through deliberated design or natural selection, HOT systems (e.g., internet, power/electrical networks, forest ecosystems) are robust to common perturbations, but fragile to unanticipated or rare events. The theoretical premises of HOT formalism are based on edge of chaos and self-organized criticality concepts, such as bifurcations, fractal structures, and critical phase transitions.

Another fundamental problem in the field of network design is related to the concept of network capacity, namely, how much information transfer can a network sustain. For instance, an analytical framework for quantifying the capacity can help us compare various platforms and choose the one that offers the best performance to reliability trade-off. Traditional methods to the capacity assessment problem of wired networks rely on queueing approaches. Nevertheless, most queueing approaches relying on Markovian assumptions can only overestimate the capacity of a network, while network traffic exhibits long-range memory effects.

Based on information theory concepts and on geometrical considerations, Kumar et al. [209] determine the capacity of a wireless network, while taking into account the architectural features of the protocol stack (e.g., spatial distribution of nodes, strategies for sharing the wireless medium, signal attenuation with distance, spatiotemporal scheduling of transmissions, etc.). More recently, Franceschetti et al. were able to derive the wireless network capacity as being proportional with the inverse square-root of n , where n is the number of users in a certain region, by using Maxwell's laws for wave propagation and Shannon's theory [74]. The main feature of the models proposed in [74] is that they consider the technology implications side (i.e., Maxwell's equations of wave propagation) of the wireless networks in addition to the informational aspects. Consequently, in these models, the authors quantify the interference among scheduled transmissions.

Nevertheless, the theory presented in [209] or [74] are not readily applicable to NoC design. For instance, such a complete theory of network capacity should suggest what is the best communication protocol for a given set of specifications (e.g., application constraints, power/energy consumption, robustness, and reliability issues). Much of the difficulties encountered in network capacity analysis are related to the intrinsic characteristics of the traffic (e.g., self-similarity, long-range dependence). Moreover, future wireless NoC architectures may also suffer from the attenuation problem as one cannot afford high power transmission on a chip. Toward this end, Zhao and Wang [213] investigate the benefits of a wireless NoC architecture and present a

synchronous and distributed medium access control protocol for solving the channel contention among RF nodes.

The availability of diverse nano devices drives the quest for designing not only low-power hybrid NoC (i.e., nodes communicating locally via wires and globally via wireless), but also for building reconfigurable computing architectures that are fault-tolerant to transient and permanent errors by adjusting their computational functionality on the fly. In this context, all the above design issues (mapping, scheduling, routing, etc.) will likely migrate from deterministic to statistical approaches where modeling space (e.g., *Which PE does what?*) and time (e.g., *For how long?*) are of crucial interest for building high performance computing platforms.

Conclusions

In this survey, we have reviewed the major design methodologies that have had a profound effect on designing the future NoC architectures. More precisely, we have addressed the problem of NoC design in the deterministic context, where the application and the architecture are modeled as graphs with worst-case type of information about the parameters of the components influencing the network traffic. Rather than simply enumerating the proposed approaches, we have taken a formal approach and also discussed the main features of each proposed solution.

One step further was made by considering the design of NoCs with partial information available (primarily under the Markovian assumption) about the application and the architecture. Similarly to the deterministic context, we have discussed various probabilistic approaches to NoC design and pointed out their advantages and limitations.

Last but not least, we have looked at emerging approaches inspired from statistical physics and information theory. Such approaches can address the problem of network design in the most general sense. In this framework, the network stops being seen as a static graph, but instead it is modeled as a thermodynamic system where packets flow from

sources to destinations similarly to particles migration between various energy levels. In this context, the fundamental question of what is the information capacity of a network can be addressed via evolutionary equations borrowed from statistical physics.

Besides the fundamental problems discussed under the above design paradigms, the NoC research community should also address several open-ended problems, such as programmability of reconfigurable and bio-inspired architectures or design under uncertainty and parameter variations, which can extend our vision outside the Boolean/von Neumann/Turing legacy.

Acknowledgments

This research is supported in part by Marco Gigascale Systems Research Center (GSRC) one of the five research centers funded under the Focus Center Research Program, a Semiconductor Research Corporation (SRC) program, by SRC contracts 2008-HJ-1823, 2008-HJ-1800, and in part by NSF Grant CCF-0702420.

The authors thank collaborators in the Core and Alternative themes of GRSC for stimulating discussions on this topic. The authors would also like to thank anonymous reviewers, as well as Prof. Petru Eles of Linkoping University and Prof. Diana Marculescu of Carnegie Mellon University for prompt and valuable feedback in early stages of this survey.

Finally, the authors would like to thank the Editor-in-Chief, Professor Sharad Malik, and the editors James Finlay and Mike Casey for their help in making this paper possible.

References

- [1] P. Abad, V. Puente, J. A. Gregorio, and P. Prieto, "Rotary router: An efficient architecture for CMP interconnection networks," *ACM SIGARCH Computer Architecture News*, vol. 35, no. 2, May 2007.
- [2] A. Adriahtenaina and A. Greiner, "Micro-network for SoC: Implementation of a 32-Port SPIN network," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, March 2003.
- [3] V. S. Adve and M. K. Vernon, "Performance analysis of mesh interconnection networks with deterministic routing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 5, no. 3, pp. 225–246, March 1994.
- [4] S. N. Adya and I. L. Markov, "Fixed-outline floorplanning: Enabling hierarchical design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 6, pp. 1120–1135, December 2003.
- [5] A. Agarwal, "Limits on Interconnection Network Performance," *Transactions on Parallel and Distributed Systems*, vol. 2, no. 4, October 1991.
- [6] J. Ahn and S. Kang, "Test scheduling of NoC-based SoCs using multiple test clocks," *Electronics and Telecommunications Research Institute Journal*, vol. 28, no. 4, pp. 475–485, August 2006.
- [7] R. Ahuja, T. Magnanti, and J. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.
- [8] R. Albert and A.-L. Barabasi, "Statistical mechanics of complex networks," *Reviews of Modern Physics*, vol. 47, p. 74, cond-mat/0106096, 2002.
- [9] A. M. Amory, E. Briao, E. Cota, M. Lubaszewski, and F. G. Moraes, "A scalable test strategy for network-on-chip routers," in *Proceedings of IEEE International Test Conference*, November 2005.

- [10] F. Angiolini, D. Atienza, S. Murali, L. Benini, and G. De Micheli, "Reliability support for on-chip memories using networks-on-chip," in *Proceedings of the International Conference on Computer Design (ICCD)*, October 2006.
- [11] F. Angiolini, P. Meloni, S. Carta, L. Benini, and L. Raffo, "Contrasting a NoC and a traditional interconnect fabric with layout awareness," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, March 2006.
- [12] Arteris [online]. Available: <http://www.arteris.com>.
- [13] G. Ascia, V. Catania, and M. Palesi, "Multi-objective mapping for mesh-based NoC architectures," in *Proceedings of the 2nd IEEE/ACM/IFIP International Conference on Hardware-Software Codesign and System Synthesis (CODES+ISSS)*, pp. 182–187, September 2004.
- [14] A. T. Balaban, *Chemical Applications of Graph Theory*. Academic Press, 1976.
- [15] J. Balfour and W. J. Dally, "Design tradeoffs for tiled CMP on-chip networks," in *Proceedings of the 20th International Conference on Supercomputing (ICS)*, pp. 187–198, June 2006.
- [16] M. Ball, C. Cifuentes, and B. Deepankar, "Partitioning of code for a massively parallel machine," in *Proceedings of the 13th International Conference on Parallel Architecture and Compilation Techniques (PACT)*, pp. 225–236, September 2004.
- [17] N. Banerjee, P. Vellank, and K. S. Chatha, "A power and performance model for network-on-chip architectures," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, p. 21250, February 2004.
- [18] A.-L. Barabási, *Linked: How Everything is Connected to Everything Else*. Plume Books, 2004.
- [19] T. A. Bartic, J. y. Mignolet, V. Nollet, T. Marescaux, D. Verkest, S. Vernalde, and R. Lauwereins, "Highly scalable network on chip for reconfigurable systems," in *Proceedings of the International Symposium on System-on-Chip*, pp. 79–82, November 2003.
- [20] P. Beerel and M. E. Roncken, "Low power and energy efficient asynchronous design," *Journal of Low Power Electronics*, vol. 3, no. 3, pp. 234–253, December 2007.
- [21] E. Beigne, F. Clermidy, S. Miermont, and P. Vivet, "Dynamic voltage and frequency scaling architecture for units integration with a GALS NoC," in *Proceedings of the 2nd ACM/IEEE International Symposium on Network-on-Chip (NOCS)*, pp. 129–138, 2008.
- [22] L. Benini and G. De Micheli, "Networks on chips: A new SoC paradigm," *IEEE Computer*, vol. 35, no. 1, January 2002.
- [23] D. Bertsekas and R. Gallager, *Data Networks*. Prentice Hall, 1992.
- [24] D. Bertozzi, L. Benini, and G. De Micheli, "Error control schemes for on-chip communication links: The energy-reliability tradeoff," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 6, pp. 818–831, June 2005.
- [25] D. Bertozzi, A. Jalabert, S. Murali, R. Tamhankar, S. Stergiou, L. Benini, and G. De Micheli, "NoC synthesis flow for customized domain specific multiprocessor systems-on-chip," *IEEE Transactions on Parallel and Distributed Systems*, vol. 16, no. 2, pp. 113–129, February 2005.

- [26] S. Bertozzi, A. Acquaviva, D. Bertozzi, and A. Poggiali, "Supporting task migration in multi-processor systems-on-chip: A feasibility study," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, March 2006.
- [27] P. S. Bhojwani and R. N. Mahapatra, "Interfacing cores with on-chip packet-switched networks," in *Proceedings of the International Conference on VLSI Design*, 2003.
- [28] P. S. Bhojwani and R. N. Mahapatra, "A robust protocol for concurrent on-Line test (COLT) of noc-based systems-on-a-chip," in *Proceedings of Design Automation Conference*, June 2007.
- [29] G. Bianconi and A.-L. Barabási, "Bose-Einstein condensation in complex networks," *Physical Review Letters*, vol. 86, pp. 5632–5635, 2001.
- [30] T. Bjerregaard and S. Mahadevan, "A survey of research and practices of network-on-chip," *ACM Computing Surveys*, vol. 38, no. 1, March 2006.
- [31] P. Bogdan, T. Dumitras, and R. Marculescu, "Stochastic communication: A new paradigm for fault-tolerant networks-on-chip," *Hindawi VLSI Design*, February 2007.
- [32] P. Bogdan and R. Marculescu, "Quantum-like effects in network-on-chip buffers behavior," in *Proceedings of the 44th Design Automation Conference (DAC)*, pp. 266–267, ACM, NY, 2007.
- [33] P. Bogdan and R. Marculescu, "Hitting time analysis for stochastic communication," in *Proceedings of the International Conference on Nano-Networks*, Boston, September 2008.
- [34] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, *Queueing Networks and Markov Chains: Modeling and Performance evaluation with Computer Science Applications*. John Wiley and Sons, 2006.
- [35] B. Bollobas, *Random Graphs*. Cambridge University Press, second ed., 2001.
- [36] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, "QNoC: QoS architecture and design process for network on chip," *Journal of Systems Architecture (EUROMICRO)*, vol. 50, nos. 2–3, pp. 105–128, February 2004.
- [37] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, "Routing table minimization for irregular mesh NoCs," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, April 2007.
- [38] A. J. Bondy and U. S. R. Murty, *Graph Theory with Applications*. John Wiley and Sons, 2005.
- [39] D. Borrione, "A generic model for formally verifying NoC communication architectures: A case study," in *Proceedings of the First International Symposium Networks-on-Chip (NOCS)*, pp. 127–136, 2007.
- [40] S. Brandt and H. D. Dahmen, *The Picture Book of Quantum Mechanics*. Springer, 2001.
- [41] M. J. Brusco and S. Stahl, *Branch-and-Bound Applications in Combinatorial Data Analysis*. New York, NY: Springer, 2005.
- [42] G. Campobello, M. Castano, C. Ciofi, and D. Mangano, "GALS networks on chip: A new solution for asynchronous delay-insensitive links," in *Proceedings of Design, Automation and Test in Europe Conference (DATE)*, March 2006.

- [43] L. P. Carloni, K. L. McMillan, and A. L. Sangiovanni-Vincentelli, "Theory of latency-insensitive design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 9, pp. 1059–1076, September 2001.
- [44] J. M. Carlson and J. Doyle, "Highly optimized tolerance: Robustness and design in complex systems," *Physical Review Letter*, vol. 84, no. 11, 2000.
- [45] V. Catania, R. Holsmark, S. Kumar, and M. Palesi, "A methodology for design of application specific deadlock-free routing algorithms for NoC systems," in *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, October 2006.
- [46] J. Chan and S. Parameswaran, "NoCEE: Energy macro-model extraction methodology for network on chip routers," in *Proceedings the of International Conference on Computer Aided Design (ICCAD)*, November 2005.
- [47] M. F. Chang, J. Cong, A. Kaplan, M. Naik, G. Reinman, E. Socher, and S. W. Tam, "CMP network-on-chip overlaid with multi-band RFinterconnect," in *Proceedings of the International Symposium on High-Performance Computer Architecture (HPCA)*, February 2008.
- [48] X. Chen and L.-S. Peh, "Leakage power modeling and optimization in interconnection networks," in *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED)*, August 2003.
- [49] C.-L. Chou, U. Y. Ogras, and R. Marculescu, "Energy- and performance-aware incremental mapping for networks-on-chip with multiple voltage levels," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 10, October 2008.
- [50] I. Cohen, O. Rottenstreich, and I. Keslassy, "Statistical approach to NoC design," in *Proceedings of the Second ACM/IEEE International Symposium Networks-on-Chip (NOCS)*, Newcastle, UK, April 2008.
- [51] M. Colajanni, B. Ciciani, and F. Quaglia, "Performance analysis of wormhole switching with adaptive routing in a two-dimensional torus," in *Proceedings of the 5th International Euro-Par Conference on Parallel Processing, Lecture Notes in Computer Science*, 1999.
- [52] Communication Synthesis Infrastructure [Online]. Available: [http:// embedded.eecs.berkeley.edu/cosi/](http://embedded.eecs.berkeley.edu/cosi/).
- [53] C. Constantinescu, "Impact of deep submicron technology on dependability of VLSI circuits," in *Proceedings of the 2002 International Conference on Dependable Systems and Networks (DSN)*, pp. 205–209, June 23–26 2002.
- [54] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. MIT Press and McGraw-Hill, Second ed., 2001.
- [55] A. K. Coskun, T. S. Rosing, and K. Whisnant, "Temperature aware task scheduling in MPSoCs," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, April 2007.
- [56] W. J. Dally, "Performance analysis of k-ary n-cube interconnection networks," *IEEE Transactions on Computers*, vol. 39, no. 6, June 1990.
- [57] W. J. Dally, "Virtual-channel flow control," *IEEE Transactions on Parallel and Distributed Systems*, vol. 3, no. 2, pp. 194–205, March 1992.

- [58] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proceedings of the Design Automation Conference (DAC)*, June 2001.
- [59] W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2004.
- [60] G. De Micheli and L. Benini, *Networks on Chips: Technology and Tools*. Morgan Kaufmann, 2006.
- [61] J. Dielissen, A. Radulescu, K. Goossens, and E. Rijpkema, "Concepts and implementation of the Philips network-on-chip," in *Proceedings of the IP-based SoC Design*, November 2003.
- [62] S. N. Dorogovtsev and J. F. F. Mendes, *Evolution of Networks: From Biological Networks to the Internet and WWW*. Oxford University Press, 2003.
- [63] J. Draper and J. Ghosh, "A comprehensive analytical model for wormhole routing in multicomputer systems," *Journal of Parallel and Distributed Computing*, vol. 23, no. 2, November 1994.
- [64] J. Duato, S. Yalamanchili, and L. Ni, *Interconnection Networks: An Engineering Approach*. San Mateo, CA: Morgan Kaufmann, 2002.
- [65] T. Dumitras, S. Kerner, and R. Marculescu, "Towards on-chip fault-tolerant communication," in *Proceedings of the 2003 Conference on Asia South Pacific Design Automation (ASP-DAC)*, Kitakyushu, Japan, January 21–24 2003.
- [66] N. Eisley and L.-S. Peh, "High-level power analysis for on-chip networks," in *Proceedings of the 2004 International Conference on Compilers, Architecture, and Synthesis For Embedded Systems (CAES)*, Washington DC, USA, September 22–25 2004.
- [67] A. Ejlali, B. M. Al-Hashimi, P. Rosinger, and S. G. Miremadi, "Joint consideration of fault-tolerance, energy-efficiency and performance in on-chip networks," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, April 2007.
- [68] P. Erdos and A. Renyi, "On random graphs," *Publicationes Mathematicae Debrecen*, vol. 6, p. 290, 1959.
- [69] P. Erdos and A. Renyi, "On the evolution of random graphs," *Publicationes Mathematicae Institut Hungary Academic Science*, vol. 5, p. 17, 1960.
- [70] A. K. Erlang, "The theory of probabilities and telephone conversations," *Nyt Tidsskrift for Matematik B*, vol. 20, 1909.
- [71] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," *Computer Communication Review*, vol. 29, p. 251, 1999.
- [72] M. A. Faruque and J. Henkel, "Minimizing virtual channel buffer for routers in on-chip communication architectures," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, 2008.
- [73] C. M. Fiduccia and R. M. Mattheyses, "A linear-time heuristic for improving network partitions," in *Proceedings of the Design Automation Conference (DAC)*, 1982.
- [74] M. Franceschetti, M. D. Migliore, and P. Minero, "The capacity of wireless networks: Information-theoretic and physical limits," *IEEE Transaction on Information Theory*, 2007.

- [75] H. Fuks and A. Lawniczak, "Performance of data networks with random links," *Mathematics and Computers in Simulation*, vol. 51, pp. 101–117, December 1999.
- [76] A. Ghosh and S. Boyd, "Growing well-connected graphs," in *Proceedings of IEEE Conference on Decision and Control*, December 2006.
- [77] K. Goossens, J. Dielissen, O. P. Gangwal, S. G. Pestana, A. Radulescu, and E. Rijpkema, "A design flow for application-specific networks-on-chip with guaranteed performance to accelerate SoC design and verification," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, March 2005.
- [78] R. Govindan and H. Tangmunarunkit, "Heuristics for internet map discovery," *IEEE — INFOCOM*, 2000.
- [79] P. Gratz, C. Kim, R. McDonald, S. W. Keckler, and D. C. Burger, "Implementation and evaluation of on-chip network architectures," in *Proceedings of the International Conference on Computer Design (ICCD)*, October 2006.
- [80] C. Grecu, P. P. Pande, B. Wang, A. Ivanov, and R. Saleh, "Methodologies and algorithms for testing switch-based NoC interconnects," in *Proceedings of the 20th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, October 03–05 2005.
- [81] F. Gruian, "Hard real-time scheduling for low energy using stochastic data and DVS processors," in *Proceedings of the International Symposium on Low-Power Electronics and Design*, August 2001.
- [82] Q. P. Gu and S. Peng, "Wavelengths requirement for permutation routing in all-optical multistage interconnection networks," in *Proceedings of the 14th International Symposium on Parallel and Distributed Processing*, pp. 761–768, 2000.
- [83] P. Guerrier and A. Greiner, "A generic architecture for on-chip packet switched interconnections," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, 2000.
- [84] Z. Guz, I. Walter, E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, "Efficient link capacity and QoS design for wormhole network-on-chip," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, March 2006.
- [85] A. Hansson, K. Goossens, and A. Radulescu, "A unified approach to mapping and routing on a network-on-chip for both best-effort and guaranteed service traffic," *Hindawi VLSI Design*, May 2007.
- [86] A. K. Hartmann and H. Rieger, *New Optimization Algorithms in Physics*. Wiley-VCH, 2004.
- [87] A. Hemani, A. Jantsch, S. Kumar, A. Postula, and J. Oberg, "Network-on-chip: An architecture for billion transistor era," in *Proceedings of the IEEE NorChip Conference*, November 2000.
- [88] J. Henkel, W. Wolf, and S. Chakradhar, "On-chip networks: A scalable communication-centric embedded system design paradigm," in *Proceedings of the 17th International Conference on VLSI Design*, pp. 845–851, January 2004.
- [89] F. S. Hillier and G. J. Lieberman, *Introduction to Operations Research*. New York: McGraw-Hill, Ch.15, Sixth ed., pp. 661–732, 1995.

- [90] W. H. Ho and T. M. Pinkston, "A methodology for designing efficient on-chip interconnects on well-behaved communication patterns," in *Proceedings of the International Symposium on High-Performance Computer Architecture*, February 2003.
- [91] M. Horowitz, R. Ho, and K. Mai, "The future of wires," *Proceedings of IEEE*, vol. 89, no. 4, pp. 490–504, April 2001.
- [92] J. Hu and R. Marculescu, "Energy-aware mapping for tile-based NOC architectures under performance constraints," in *Proceedings of the Conference on Asia South Pacific Design Automation (ASP-DAC)*, Kitakyushu, Japan, January 21–24 2003.
- [93] J. Hu and R. Marculescu, "DyAD — Smart routing for networks-on-chip," in *Proceedings of the Design Automation Conference (DAC)*, June 2004.
- [94] J. Hu and R. Marculescu, "Communication and task scheduling of application-specific networks-on-chip," *IEE Proceedings of Computers and Digital Techniques*, vol. 152, no. 5, pp. 643–651, September 2005.
- [95] J. Hu and R. Marculescu, "Energy- and performance-aware mapping for regular NoC architectures," *IEEE Transactions on Computer-Aided Design Integrated Circuits and Systems*, vol. 24, no. 4, pp. 551–562, April 2005.
- [96] J. Hu, U. Y. Ogras, and R. Marculescu, "System-level buffer allocation for application-specific networks-on-chip router design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 12, pp. 2919–2933, December 2006.
- [97] P. Hu and L. Kleinrock, "An analytical model for wormhole routing with finite size input buffers," in *Proceedings of the 15th International Teletraffic Congress*, June 1997.
- [98] T.-C. Huang, U. Y. Ogras, and R. Marculescu, "Virtual channels planning for networks-on-chip," in *Proceedings of the 8th International Symposium on Quality Electronic Design (ISQED)*, San Jose, March 2007.
- [99] W. Hung, C. Addo-Quaye, T. Theocharides, Y. Xie, V. Narayanan, and M. J. Irwin, "Thermal-aware IP virtualization and placement for networks-on-chip architecture," in *Proceedings of the IEEE International Conference on Computer Design (ICCD)*, October 11–13 2004.
- [100] iNoCs Structured Interconnects [Online]. Available: <http://www.inocs.com>.
- [101] A. Jalabert, S. Murali, L. Benini, and G. De Micheli, "xpipescompiler: A tool for instantiating application specific networks on chip," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, 2004.
- [102] A. Jantsch and H. Tenhunen, *Networks-on-Chip*. Norwell, MA: Kluwer, 2003.
- [103] Y. Jin, E. J. Kim, and K. H. Yum, "Peak power control for a QoS capable on-chip network," in *Proceedings of the International Conference on Parallel Processing (ICPP)*, June 14–17 2005.
- [104] D. G. Kendall, "Some problems in the theory of queues," *Journal of Royal Statistical Society Series B*, vol. 13, pp. 230–264, 1951.
- [105] D. Kim, K. Kim, J. Kim, S. Lee, and H. Yoo, "Solutions for real chip implementation issues of NoC and their application to memory-centric NoC," in *Proceedings of the First International Symposium on Networks-on-Chip (NOCS)*, pp. 30–39, May 2007.

- [106] E. J. Kim, K. H. Yum, G. M. Link, C. R. Das, N. Vijaykrishnan, M. Kandemir, and M. J. Irwin, "Energy optimization techniques in cluster interconnects," in *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED)*, Seoul, Korea, August 25–27 2003.
- [107] J. Kim, W. J. Dally, and D. Abts, "Flattened butterfly: A cost-efficient topology for high-radix networks," in *Proceedings of the 33rd Annual International Symposium on Computer Architecture (ISCA)*, June 2007.
- [108] M. Kim, D. Kim, and G. E. Sobelman, "Adaptive scheduling for CDMA based networks-on-chip," in *Proceedings of the IEEE Northeast Workshop on Circuits and Systems*, pp. 100–103, May 2005.
- [109] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
- [110] L. Kleinrock, *Queueing Systems*. New York, NY: Wiley Interscience, 1976.
- [111] T. Kogel, M. Doerper, A. Wiefierink, R. Leupers, G. Ascheid, and H. Meyr, "A modular simulation framework for architectural exploration of on-chip interconnection networks," in *Proceedings of the 1st IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, Newport Beach, CA, USA, October 01–03 2003.
- [112] A. N. Kolmogorov, *Foundations of the Theory of Probability*. New York, NY: Chelsea Publishing Company, Second english ed., 1956.
- [113] C. M. Krishna and K. G. Shin, *Real-time Systems*. McGraw-Hill, 1997.
- [114] A. Kumar, L.-S. Peh, P. Kundu, and N. K. Jha, "Express virtual channels: Towards the ideal interconnection fabric," in *Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA)*, June 2007.
- [115] R. Kumar, V. Zyuban, and D. M. Tullsen, "Interconnections in multi-core architectures: Understanding mechanisms, overheads and scaling," in *Proceedings of the 32nd Annual International Symposium on Computer Architecture (ISCA)*, June 2005.
- [116] H. G. Lee, N. Chang, U. Y. Ogras, and R. Marculescu, "On-chip communication architecture exploration: A quantitative evaluation of point-to-point, bus and network-on-chip approaches," *ACM Transactions on Design Automation of Electronic Systems*, vol. 12, no. 3, August 2007.
- [117] K. Lee, S. J. Lee, S. E. Kim, H. M. Choi, D. Kim, S. Kim, M. W. Lee, and H. J. Yoo, "A 51 mW 1.6 GHz on-chip network for low-power heterogeneous SoC platform," in *Proceedings of the IEEE International Solid-State Circuits Conference (ISSCC)*, pp. 152–518, February 15–19 2004.
- [118] T. Lehtonen, P. Liljeberg, and J. Plosila, "Fault tolerance analysis of NoC architectures," in *Proceedings of the International Symposium on Circuits and Systems (ISCAS)*, pp. 361–364, May 27–30 2007.
- [119] T. Lei and S. Kumar, "A Two-step genetic algorithm for mapping task graphs to a network on chip architecture," in *Proceedings of the Euromicro Symposium on Digital System Design*, 2003.
- [120] L. F. Leung and C. Y. Tsui, "Optimal link scheduling on improving best-effort and guaranteed services performance in network-on-chip system," in *Proceedings of the Design Automation Conference (DAC)*, July 2006.

- [121] F. Li, G. Chen, and M. Kandemir, "Compiler-directed voltage scaling on communication links for reducing power consumption," in *Proceedings of the International Conference on Computer-Aided Design (ICCAD)*, 2005.
- [122] J. Liang, A. Laffely, S. Srinivasan, and R. Tessier, "An architecture and compiler for scalable on-chip communication," *IEEE Transactions on Very-Largescale Integration (VLSI) Systems*, vol. 12, no. 7, pp. 711–726, July 2004.
- [123] J. Luo and N. K. Jha, "Power-conscious joint scheduling of periodic task graphs and aperiodic tasks in distributed real-time embedded systems," in *Proceedings of the International Conference on Computer-Aided Design (ICCAD)*, November 2000.
- [124] B. B. Mandelbrot, *The Fractal Geometry of Nature*. San Francisco: Freeman Company, 1982.
- [125] S. Manolache, P. Eles, and Z. Peng, "Fault and energy-aware communication mapping with guaranteed latency for applications implemented on NoC," in *Proceedings of the Design Automation Conference (DAC)*, July 2005.
- [126] S. Manolache, P. Eles, and Z. Peng, "Buffer space optimisation with communication synthesis and traffic shaping for NoCs," in *Proceedings of the Design Automation and Test in Europe Conference*, Munich, Germany, March 6–10 2006.
- [127] T. Marescaux and H. Corporaal, "Introducing the superGT network-on-chip," in *Proceedings of the Design Automation Conference (DAC)*, June 2007.
- [128] A. A. Markov, "Extension of the law of large numbers to dependent events," *Bulletin Society Physics and Mathematics Kazan*, vol. 15, no. 2, no. 2, pp. 135–156, (in Russian), 1906.
- [129] A. Messiah, *Quantum Mechanics: Two Volumes Bound as One*. Courier Dover Publications, 1999.
- [130] S. Meyn and R. Tweedie, *Markov Chains and Stochastic Stability*. Springer Verlag, 2005.
- [131] Z. Michalewicz and D. B. Fogel, *How to Solve it: Modern Heuristics*. Springer, 2000.
- [132] R. Milner, *Communication and Concurrency*. Upper Saddle River, NJ, USA: Prentice-Hall, 1989.
- [133] R. Mullins, A. West, and S. Moore, "Low-latency virtual-channel routers for on-chip networks," in *Proceedings of the 31st Annual International Symposium on Computer Architecture*, Munchen, Germany, June 19–23 2004.
- [134] S. Murali, D. Atienza, L. Benini, and G. De Micheli, "A method for routing packets across multiple paths in NoCs with in-order delivery and fault-tolerance guarantees," *Hindawi VLSI Design*, vol. 2007, no. 37627, p. 11, May 2007.
- [135] S. Murali, L. Benini, T. Theocharides, N. Vijaykrishnan, M. J. Irwin, and G. De Micheli, "Analysis of error recovery schemes for networks on chip," *IEEE Design and Test of Computers*, vol. 22, no. 5, pp. 434–442, September 2005.
- [136] S. Murali, M. Coenen, A. Radulescu, K. Goossens, and G. De Micheli, "A methodology for mapping multiple use-cases onto networks on chips," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, Munich, Germany, March 06–10 2006.

- [137] S. Murali and G. De Micheli, "Bandwidth-constrained mapping of cores onto NoC architectures," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, February 16–20 2004.
- [138] S. Murali, P. Meloni, F. Angiolini, D. Atienza, S. Carta, L. Benini, G. De Micheli, and L. Raffo, "Designing application-specific networks on chips with floorplan information," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, San Jose, California, November 05–09 2006.
- [139] S. Murali, A. Mutapcic, D. Atienza, R. Gupta, S. Boyd, and G. De Micheli, "Temperature-aware processor frequency assignment for MPSoCs using convex optimization," in *Proceedings of the 5th IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, Salzburg, Austria, September 30–October 03 2007.
- [140] M. Newman, A.-L. Barabasi, and D. J. Watts, *The Structure and Dynamics of Networks*. Princeton University Press, 2006.
- [141] C. A. Nicopoulos, D. Park, J. Kim, V. Narayanan, M. S. Yousif, and C. Das, "ViChaR: A dynamic virtual channel regulator for network-on-chip routers," in *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*, December 09–13 2006.
- [142] E. Nilsson, M. Millberg, J. Oberg, and A. Jantsch, "Load distribution with the proximity congestion awareness in a network on chip," *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, vol. 1, March 03–07 2003.
- [143] NIRGAM [Online]. Available: <http://nirgam.ecs.soton.ac.uk/>.
- [144] Nostrum [Online]. Available: <http://www.imit.kth.se/info/FOFU/Nostrum/>.
- [145] U. Y. Ogras, J. Hu, and R. Marculescu, "Key research problems in NoC design: A holistic perspective," in *Proceedings of the 3rd IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, Jersey City, NJ, USA, September 19–21 2005.
- [146] U. Y. Ogras and R. Marculescu, "Energy- and performance-driven NoC communication architecture synthesis using a decomposition approach," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, March 07–11 2005.
- [147] U. Y. Ogras and R. Marculescu, "It's a small world after all: NoC performance optimization via long-range link insertion," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Special Section on Hardware/Software Codesign and System Synthesis, vol. 14, no. 7, pp. 693–706, July 2006.
- [148] U. Y. Ogras and R. Marculescu, "Analytical router modeling for networks-on-chip performance analysis," in *Proceedings of Design, Automation and Test in Europe Conference (DATE)*, pp. 1096–1101, Nice, France, April 16–20 2007.
- [149] U. Y. Ogras and R. Marculescu, "Analysis and optimization of prediction-based flow control in networks-on-chip," *ACM Transactions on Design Automation of Electronic Systems*, vol. 13, no. 1, January 2008.
- [150] U. Y. Ogras, R. Marculescu, P. Choudhary, and D. Marculescu, "Voltage-frequency island partitioning for GALS-based networks-on-chip," in *Proceedings of the IEEE/ACM Design Automation Conference*, San Diego, June 2007.

- [151] U. Y. Ogras, R. Marculescu, D. Marculescu, and E.-G. Jung, "Design and management of voltage-frequency island partitioned networks-on-chip," to appear in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Special Section on Networks-on-Chip, 2009.
- [152] Orion [Online]. Available: <http://www.princeton.edu/peh/orion.html>.
- [153] L. Ost, A. Mello, J. Palma, F. Moraes, and N. Calazans, "MAIA: A framework for networks on chip generation and verification," in *Proceedings of Asia South Pacific Design Automation Conference (ASPDAC)*, January 2005.
- [154] G. Palermo and C. Silvano, "PIRATE: A framework for power/performance exploration of network-on-chip architectures," in *Proceedings of the 14th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS), Lecture Notes in Computer Science*, Springer, pp. 521–531, Santorini, Greece, September 15–17 2004.
- [155] P. P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, "Performance evaluation and design trade-offs for network-on-chip interconnect architectures," *IEEE Transactions on Computers*, vol. 54, no. 8, pp. 1025–1040, August 2005.
- [156] P. Pardalos and M. Resende, *Handbook of Applied Optimization*. Oxford University Press, 2002.
- [157] P. M. Pardalos, F. Rendl, and H. Wolkowicz, "The quadratic assignment problem: A survey and recent developments," *Quadratic Assignment and Related Problems*, vol. 16, pp. 1–42, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 1994.
- [158] K. Park and W. Willinger, *Self-Similar Network Traffic and Performance Evaluation*. Wiley Interscience, 1999.
- [159] C. S. Patel, S. M. Chai, S. Yalamanchili, and D. E. Schimmel, "Power constrained design of multiprocessor interconnection networks," in *Proceedings of the IEEE International Conference on Computer Design (ICCD)*, pp. 408–416, October 12–15 1997.
- [160] K. Petersen and J. Oberg, "Toward a scalable test methodology for 2D-mesh network-on-chips," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, Nice, France, April 16–20 2007.
- [161] D. Pham, S. Asano, M. Bolliger, M. N. Day, H. P. Hofstee, C. Johns, J. Kahle, A. Kameyama, J. Keaty, Y. Masubuchi, M. Riley, D. Shippy, D. Stasiak, M. Suzuoki, M. Wang, J. Warnock, S. Weitzel, D. Wendel, T. Yamazaki, and K. Yazawa, "The design and implementation of a first-generation CELL processor," *Proceedings of the International Solid-State Circuits Conference (ISSCC)*, vol. 1, pp. 184–592, February 10–15 2005.
- [162] A. Pinto, L. P. Carloni, and A. L. Sangiovanni-Vincentelli, "Efficient synthesis of networks on chip," in *Proceedings of the 21st International Conference on Computer Design (ICCD)*, October 13–15 2003.
- [163] A. Pinto, L. P. Carloni, and A. L. Sangiovanni-Vincentelli, "COSI: A framework for the design of interconnection networks," *IEEE Design and Test of Computers*, pp. 402–415, September/October 2008.
- [164] P. Pop, P. Eles, T. Pop, and P. Zebo, "An approach to incremental design of distributed embedded systems," in *Proceedings of the 38th Conference on Design Automation (DAC)*, Las Vegas, Nevada, United States, June 2001.

- [165] P. Poplavko, T. Basten, M. Bekooij, J. van Meerbergen, and B. Mesman, "Task-level timing models for guaranteed performance in multiprocessor networks-on-chip," in *Proceedings of the International Conference on Compilers, Architecture and Synthesis For Embedded Systems (CASES)*, San Jose, California, USA, October 30–November 01 2003.
- [166] V. Puente, J. A. Gregorio, F. Vallejo, and R. Beivide, "Immunet: A cheap and robust fault-tolerant packet routing mechanism," in *Proceedings of the 31st Annual International Symposium on Computer Architecture*, pp. 198–209, June 19–23 2004.
- [167] A. Pullini, F. Angiolini, D. Bertozzi, and L. Benini, "Fault tolerance overhead in network-on-chip flow control schemes," in *Proceedings of the 18th Annual Symposium on Integrated Circuits and System Design (SBCCI)*, Florianopolis, Brazil, September 04–07 2005.
- [168] G. Rangarajan and M. Ding, *Processes with Long-Range Correlations: Theory and Applications*. Berlin Heidelberg: Springer-Verlag, 2003.
- [169] C. R. Reeves and J. E. Rowe, *Genetic Algorithms. Principles and Perspectives: A Guide to GA Theory*. Springer, 2003.
- [170] C. Rhee, H. Y. Jeong, and S. Ha, "Many-to-many core-switch mapping in 2-D mesh NoC architectures," in *Proceedings of the IEEE International Conference on Computer Design (ICCD)*, pp. 438–443, October 11–13 2004.
- [171] S. M. Ross, *Simulation*. Academic Press, Second ed., 1997.
- [172] J. Rossiter, *Model Based Predictive Control, A Practical Approach*. CRC Press, 2002.
- [173] S. M. Sait and H. Youssef, *VLSI Physical Design Automation: Theory and Practice*. McGraw-Hill Inc, 1994.
- [174] G. Salaun, W. Serwe, Y. Thonnart, and P. Vivet, "Formal verification of CHP specifications with CADP illustration on an asynchronous network-on-chip," in *Proceedings of the 13th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, March 12–14 2007.
- [175] A. Schrijver, *Theory of Linear and Integer Programming*. John Wiley and Sons, 1998.
- [176] D. Seo, A. Ali, W. Lim, N. Rafique, and M. Thottethodi, "Near-optimal worst-case throughput routing for two-dimensional mesh networks," in *Proceedings of the 32nd Annual International Symposium on Computer Architecture (ISCA)*, June 04–08 2005.
- [177] A. Shacham, K. Bergman, and L. P. Carloni, "The case for low-power photonic networks-on-chip," in *Proceedings of the 44th Annual Conference on Design Automation (DAC)*, San Diego, California, June 04–08 2007.
- [178] L. Shang, L.-S. Peh, and N. K. Jha, "Dynamic voltage scaling with links for power optimization of interconnection networks," in *Proceedings of the 9th International Symposium on High-Performance Computer Architecture (HPCA)*, February 08–12 2003.
- [179] L. Shang, L.-S. Peh, and N. K. Jha, "PowerHerd: Dynamically satisfying peak power constraints in interconnection networks," in *Proceedings of the ACM International Symposium on Supercomputing*, pp. 98–108, June 2003.

- [180] L. Shang, L.-S. Peh, A. Kumar, and N. K. Jha, "Thermal modeling, characterization and management of on-chip networks," in *Proceedings of the 37th Annual IEEE/ACM International Symposium on Microarchitecture*, Portland, Oregon, December 04–08 2004.
- [181] A. Sheibanyrad, I. M. Panades, and A. Greiner, "Systematic comparison between the asynchronous and the multi-synchronous implementations of a network-on-chip architecture," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, Nice, France, April 16–20 2007.
- [182] B. Shim and N. R. Shanbhag, "Energy-efficient soft-error tolerant digital signal processing," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 14, no. 4, pp. 336–348, April 2006.
- [183] D. Shin and J. Kim, "Power-aware communication optimization for networks-on-chips with voltage scalable links," in *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, September 08–10 2004.
- [184] D. Shin, J. Kim, and S. Lee, "Intra-task voltage scheduling for low-energy, hard real-time applications," *IEEE Design and Test*, vol. 18, no. 2, pp. 20–30, March/April 2001.
- [185] P. Shivakumar, M. Kistler, S. Keckler, D. Burger, and L. Alvisi, "Modeling the effect of technology trends on soft error rate of combinational logic," in *Proceedings of the International Conference on Dependable Systems and Networks (DSN)*, June 23–26 2002.
- [186] T. Simunic and S. Boyd, "Managing power consumption in networks on chip," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, March 04–08 2002.
- [187] SONICS Smart Interconnects [Online]. Available: <http://www.sonicsinc.com>.
- [188] V. Soteriou, H.-S. Wang, and L.-S. Peh, "A statistical traffic model for on-chip interconnection networks," in *Proceedings of the 14th IEEE International Symposium on Modeling, Analysis, and Simulation*, September 11–14 2006.
- [189] K. Srinivasan and K. Chatha, "ISIS: A genetic algorithm based technique for custom on-chip interconnection network synthesis," in *Proceedings of the 18th International Conference on VLSI Design (VLSID) Held Jointly with 4th International Conference on Embedded Systems Design*, January 03–07 2005.
- [190] K. Srinivasan and K. S. Chatha, "A technique for low energy mapping and routing in network-on-chip architectures," in *Proceedings of the 2005 International Symposium on Low Power Electronics and Design (ISLPED)*, San Diego, CA, USA, August 08–10 2005.
- [191] K. Srinivasan and K. S. Chatha, "A low complexity heuristic for design of custom network-on-chip architectures," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, Munich, Germany, March 06–10 2006.
- [192] K. Srinivasan, K. S. Chatha, and G. Konjevod, "Linear programming based techniques for synthesis of network-on-chip architectures," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 4, pp. 407–420, April 2006.

- [193] S. Sriram and S. S. Bhattacharyya, *Embedded Multiprocessors: Scheduling and Synchronization*. Marcel Dekker Inc., 2002.
- [194] 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," *Journal of Systems Architecture: The EUROMICRO Journal*, vol. 54, no. 3–4, March 2008.
- [195] S. Stuijk, M. Geilen, and T. Basten, "Throughput-buffering trade-off exploration for cyclo-static and synchronous dataflow graphs," *IEEE Transactions on Computers*, vol. 57, no. 10, p. 28, October 2008.
- [196] C. Sun, L. Shang, and R. P. Dick, "Three-dimensional multi-processor system-on-chip thermal optimization," in *Proceedings of the 5th IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, pp. 117–122, Salzburg, Austria, September 30–October 03 2007.
- [197] H. Takagi, *Queueing Analysis — A Foundation of Performance Evaluation*. North-Holland: Elsevier, 1991.
- [198] M. B. Taylor, J. Kim, J. Miller, D. Wentzloff, F. Ghodrati, B. Greenwald, H. Hoffman, P. Johnson, W. Jae-Wook Lee, A. Ma, A. Saraf, M. Seneski, N. Shnidman, V. Strumpfen, M. Frank, S. Amarasinghe, and A. Agarwal, "The Raw microprocessor: A computational fabric for software circuits and general purpose programs," *IEEE Micro*, vol. 22, no. 2, pp. 25–35, March/April 2002.
- [199] C. Teuscher, "Nature-inspired interconnects for self-assembled largescale network-on-chip designs," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 17, no. 2, pp. 26–106, 2007.
- [200] J. W. van den Brand, C. Ciordas, K. Goossens, and T. Basten, "Congestion-controlled best-effort communication for networks-on-chip," in *Proceedings of Design Automation and Test in Europe Conference (DATE)*, April 2007.
- [201] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, P. Iyer, A. Singh, T. Jacob, S. Jain, S. Venkataraman, Y. Hoskote, and N. Borkar, "An 80-Tile 1.28TFLOPS network-on-chip in 65nm CMOS," in *Proceedings of IEEE International Solid-State Circuits Conference (ISSCC)*, February 2007.
- [202] G. Varatkar and R. Marculescu, "On-chip traffic modeling and synthesis for MPEG-2 video applications," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, no. 1, pp. 108–119, January 2004.
- [203] G. Varatkar, S. Narayanan, N. R. Shanbhag, and D. L. Jones, "Trends in energy-efficiency and robustness using stochastic sensor network-on-a-chip," in *Proceedings of the 18th ACM Great Lakes Symposium on VLSI*, Orlando, Florida, USA, May 04–06 2008.
- [204] H. Wang, L.-S. Peh, and S. Malik, "Power-driven design of router microarchitectures in on-chip networks," in *Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture*, December 03–05 2003.
- [205] H. Wang, L.-S. Peh, and S. Malik, "A technology-aware and energy-oriented topology exploration for on-chip networks," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, March 07–11 2005.

- [206] H. Wang, X. Zhu, L.-S. Peh, and S. Malik, "Orion: A power-performance simulator for interconnection networks," in *Proceedings of the 35th Annual International Symposium on Microarchitecture*, pp. 294–305, November 2002.
- [207] D. J. Watts, *Small Worlds: The Dynamics of Networks Between Order and Randomness*. Princeton, NJ: Princeton University Press, 1999.
- [208] Wormsim [Online]. Available: <http://www.ece.cmu.edu/~sld/>.
- [209] F. Xue and P. R. Kumar, "Scaling laws for ad-hoc wireless networks: An information theoretic approach," *Foundations and Trends in Networking, Now Publishers Inc.*, vol. 1, no. 2, pp. 145–270, 2006.
- [210] T. T. Ye, L. Benini, and G. De Micheli, "Analysis of power consumption on switch fabrics in network routers," in *Proceedings of the 39th Conference on Design Automation (DAC)*, pp. 524–529, New Orleans, Louisiana, USA, June 10–14 2002.
- [211] T. T. Ye and G. De Micheli, "Physical planning for multiprocessor networks and switch fabrics," in *Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures and Processors*, pp. 97–107, June 2003.
- [212] Z. Yu and B. Baas, "Implementing tile-based chip multiprocessors with GALS clocking styles," in *Proceedings of the International Conference on Computer Design*, pp. 174–179, October 2006.
- [213] D. Zhao and Y. Wang, "SD-MAC: Design and synthesis of a hardware-efficient collision-free QoS-aware MAC protocol for wireless networks-on-chip," *IEEE Transactions on Computers*, vol. 57, no. 9, pp. 1230–1245, May 2008.
- [214] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength pareto evolutionary algorithm," *Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, pp. 95–100, Greece, Athens 2001.