Systematic Construction of Deadlock-Free Routing for NoC Using Integer Linear Programming

Shuang Liu and Martin Radetzki
Chair of Embedded Systems
Institute of Computer Architecture and Computer Engineering
University of Stuttgart
Stuttgart, Germany
{shuang.liu, martin.radetzki}@informatik.uni-stuttgart.de

***Abstract*—Network-on-Chip (NoC) presents a promising solution for on-chip communication in highly integrated System-on-Chips (SoCs). This work addresses critical challenges in NoC design, including routing construction, application mapping, and particularly the issue of deadlocks in the widely-used wormhole routing method. In this paper, an Integer Linear Programming (ILP) approach for deadlock-free routing is proposed, applicable to arbitrary network topologies. We systematically analyze deadlock-free routing construction for mesh and torus topologies under uniform random traffic and provide alternative solutions to turn models. In the context of application-specific NoCs, application mapping, and deadlock-free routing are integrated within a single ILP. Through evaluation with several benchmark applications, it is demonstrated that the ILP method consistently delivers optimal solutions and could obtain better results than various heuristic methods within an acceptable time. Fault tolerance is also explored and existing techniques are incorporated into the ILP approach. As an illustrative example, application mapping and a 1-link-fault-tolerant deadlock-free routing for the MP3 application on a mesh network is performed.**

***Index Terms*—Network-on-chip, integer linear programming, deadlock-free, routing construction, application mapping, fault tolerance.**

I. Introduction

With the continuous increase in the integration of components on a single chip, traditional bus-based System-on-Chip (SoC) architectures have encountered performance bottlenecks. In response to this challenge, architectures with Network-on-Chip (NoC) have been proposed [1], [2]. Methodologies from network communication are introduced into on-chip communication. An NoC typically comprises interconnected switches and processing elements (PEs), which are responsible for providing system functionality. PEs are connected to these switches via network interfaces (NIs) [3]. Communication between PEs is achieved through the transmission of data packets.

NoCs can be designed using either regular or application-specific (irregular) topologies. Regular topologies, such as mesh-based topologies, offer the advantage of reusing established topological design, reducing both design time and cost. In contrast, application-specific topologies provide greater flexibility and the potential for enhanced performance. However, they come with the drawback of a more intricate design process, resulting in higher overall design cost.

On a given topology, routing methods determine the path of packets from the source to the destination for each communication flow. These routing methods are typically classified into two categories: deterministic routing and adaptive routing. In deterministic routing, routing paths are pre-determined while adaptive routing takes the current state of the network into account in making routing decisions. For most application-specific NoCs with communication flows known in advance, deterministic routing is often preferred as it can reduce system complexity. The routing can be implemented with routing table, or with more efficient compressed representations such as logic-based distributed routing (LBDR) [4] and its variants. Moreover, as the routing is determined at design time, it may be hard-coded into combinational logic and optimized using standard logic level optimization techniques, and would not require memory. Consequently, finding the optimal routing is a fundamental challenge in NoC design.

In an application-specific NoC, individual PEs may offer varying functions and generate irregular communication flows. Besides routing, the crucial design task is determining the binding between each PE and its respective switch, as this association significantly impacts routing. The synergistic optimization of application mapping and routing can yield a substantial increase in system performance.

Wormhole routing [5], due to its good performance, particularly in term of high throughput and minimal buffering requirements, is widely adopted in NoCs. However, a notable drawback of wormhole routing is the higher risk for deadlocks. Deadlocks occur when no packets can progress towards their destination due to resources occupation. The root cause of these deadlocks is the presence of cyclic dependencies of resources within the routing channels.

In this paper, we introduce an Integer Linear Programming (ILP) method for the construction of deadlock-free routing paths in NoC architectures. The main contributions of the work are:

- Formulating an ILP approach for the optimal construction of deadlock-free routing paths, applicable to a wide range of network topologies.
- Systematically constructing and evaluating deadlock-free routing under uniform random traffic for regular topologies, providing alternative solutions to the well-known

turn models.

- Simultaneously addressing application mapping and deadlock-free routing construction within a single ILP framework.
- Incorporating fault tolerance techniques to achieve application mapping, optimal deadlock-free routing, and fault tolerance within a single ILP.

The remaining paper is organized as follows: Section II reviews related work in this field. Section III provides a detailed description of the method employed in this work. Section IV presents the experimental setup, results, and subsequent discussions. Section V illustrates the incorporation of fault tolerance techniques into our approach. Finally, section VI concludes the work.

II. Related Work

Deadlock-free routing in networks utilizing wormhole routing has been analyzed since the inception of this routing method. In [5], a deadlock-free routing method was presented by introducing virtual channels. Since then, various approaches have been developed. One approach is the creation of virtual channels [6], where multiple packets are allowed to share the same physical channel. Although virtual channels offer flexibility, they increase network complexity and can sometimes result in decreased performance, making them less desirable in many NoC configurations.

Another notable approach is the turn model [7]. This model, including its variants such as XY turn model, odd-even turn model [8], and repetitive turn model [9], prohibits specific turns within the network to prevent cyclic dependencies on resources. Alternatively, forbidden turns can be represented using integer numbering of links [10], allowing only turns to links in either ascending or descending orders.

Similar approaches have been employed in NoCs for constructing both deterministic and adaptive routing. For example, an odd-even turn model has been applied to 2D mesh-based NoCs in [11], and an adaptive routing algorithm for 3D mesh-based NoCs is described in [12]. In [13], the analysis of turns in hexagonal NoCs is performed. However, it is worth noting that forbidding turns without knowledge of the actual communication flows can lead to performance degradation, as it may restrict the optimal routing path between two PEs, even when this is not necessary. A comprehensive discussion of the necessary and sufficient conditions for deadlock can be found in [14], [15].

The synthesis of NoCs has been addressed in many prior works, with a primary emphasis on application mapping. Several of these studies utilize ILP-based methods. Authors of [16] developed an ILP approach to achieve energy-efficient application mapping. In [17], a method for application mapping that includes spare cores for mesh networks is explained. The same authors analyzed a mesh-of-tree-based NoC in [18]. However, these works are primarily based and valid only on a single particular topology. In [19], an ILP-based method for synthesizing application-specific topologies is introduced. Furthermore, in [20], [21], ILP methodologies that consider fault tolerance for application-specific topologies are described. It is worth noting that neither of these works has presented results that are guaranteed to be deadlock-free.

In addition to ILP-based methods, heuristic-based approaches have been utilized for NoC synthesis. For instance, in [22], the authors proposed a heuristic that utilizes ILP relaxation. In [23], an algorithm for mapping applications onto a mesh NoC under bandwidth constraints is presented. Simulated annealing is utilized in [24], while in [25], [26], authors present a particle swarm optimization based method and Kernighan-Lin based partition heuristic for application mapping, respectively. Furthermore, a genetic algorithm is presented in [27]. Unlike ILP-based methods, heuristic-based approaches do not guarantee the delivery of optimal solutions for the problem. Also, these approaches do not guarantee the absence of deadlocks.

III. Method

A. Problem Definition

Definition 1 (Switch Graph G_S): A directed graph $G_S(V_S, E_S)$ represents the topology of the switch network, where each vertex in V_S denotes a switch, and each edge in E_S denotes a link between two switches. Bidirectional links are represented with two directional edges.

Definition 2 (Communication Graph G_C): A weighted directed graph $G_C(V_C, E_C, w)$ represents the communication flow among PEs, where each vertex in V_C denotes a PE, and each directed edge in E_C denotes the communication flow between two PEs. The weight w of the edges indicates the required bandwidth for the communication flow.

The primary goal of this method is to identify the optimal deadlock-free routing paths in G_S for all communication flows within G_C based on selected criteria.

B. ILP Formulation

In NoCs with regular topologies, dedicated ports on the switches are reserved for the links to NIs of PEs. Typically, in mesh-based topologies, one port is allocated on each switch for the link to a PE. In other cases, such as tree-based or hybrid topologies, multiple ports may be available for connecting to multiple PEs. In the context of this ILP, we assume that the bindings between PEs and switches are known. Consequently, the PEs can be identified by the switches to which they are connected.

To construct the ILP, the following variables are defined:

$R_{(i,j)}(u, v)$: a binary variable that indicates whether communication (i, j) in G_C routes through link (u, v) in G_S. $R_{(i,j)}(u, v) = 1$ when the communication routes through the link.

$BW(u, v)$: an integer variable that represents the required bandwidth of link (u, v) in G_S.

$T_{(i,j)}(u, v, w)$: a binary variable that indicates whether the turn from (u, v) to (v, w) in G_S is in the routing path of (i, j) in G_C. $T_{(i,j)}(u, v, w) = 1$ if the turn is part of the routing path. Turns refer to transmissions between different ports within a switch, including straight-through transmissions.

$N(u,v)$: an integer variable used to number each (u,v) in G_S. To prevent deadlock, turns are only allowed from edges with higher numbers to edges with lower numbers, i.e. in descending order. It is also possible to formulate the ILP to allow turns only in ascending order.

Typical performance criteria for NoCs include factors such as latency, bandwidth requirement, production cost, and runtime costs, such as communication cost or energy consumption. These criteria serve as objectives of the ILP. For example, when the goal is to minimize the number of total hops (thereby the average latency), the ILP can be formulated as follows:

$$\text{Minimize: } \sum_{(i,j)\in E_C} \sum_{(u,v)\in E_S} R_{(i,j)}(u,v) \tag{1}$$

Subject to:

$$\sum_{v:(u,v)\in E_S} (R_{(i,j)}(u,v) - R_{(i,j)}(v,u)) = \begin{cases} 1 & \text{if } u = i \\ -1 & \text{if } u = j \\ 0 & \text{if } u \neq i,\ u \neq j, \end{cases} \quad \forall u \in V_S,\ \forall (i,j) \in E_C \tag{2}$$

$$R_{(i,j)}(u,v) + R_{(i,j)}(v,u) \leq 1,\ \forall (u,v) \in E_S,\ \forall (i,j) \in E_C \tag{3}$$

$$BW(u,v) = \sum_{(i,j)\in E_C} (R_{(i,j)}(u,v) \times w(i,j)),\ \forall (u,v) \in E_S \tag{4}$$

$$BW(u,v) \leq BW_{\text{max}},\ \forall (u,v) \in E_S \tag{5}$$

$$T_{(i,j)}(u,v,w) = \min(R_{(i,j)}(u,v), R_{(i,j)}(v,w)),\ \forall v : (u,v),(v,w) \in E_S,\ u \neq w,\ \forall (i,j) \in E_C \tag{6}$$

$$N(u,v) - N(v,w) \geq 1 \text{ if } T_{(i,j)}(u,v,w) = 1,\ \forall v : (u,v),(v,w) \in E_S,\ u \neq w,\ \forall (i,j) \in E_C \tag{7}$$

The constraint (2) defines routing paths for each communication flow. The constraint (3) ensures that no turning backs (i.e. u-turns) are allowed. Constraints (4) and (5) calculate and restrict the required bandwidth of each link in the network, with BW_{max} representing the predefined maximum allowed bandwidth for the links. Constraints (6) and (7) are employed for deadlock avoidance. The constraint (6) indicates whether a turn from switch u through switch v to switch w is in the routing path of communication (i,j). The constraint (7) assigns an integer number to each link, allowing turns only from links with higher numbers to links with lower numbers. This prohibition of cyclic dependencies of resources effectively prevents deadlock. Constraints with min, max, and if are transformed into a set of $\leq$ and $\geq$ constraints by the ILP framework.

The constraint (5) requires a definition of the maximum allowed bandwidth. Normally, this is obtained from the specification of the yet to be designed NoC. The optimal value can also be acquired directly from the ILP by replacing the objective (1) to an objective minimizing the required maximum bandwidth:

$$\text{Minimize: } BW_{\text{max}} = \max_{(u,v)\in E_S} (BW(u,v)) \tag{8}$$

However, the number of hops is no longer restricted by the ILP. This may lead to results containing routing paths with a very large latency, even if the required bandwidth is minimized. To address this, the objective (1) has to be transformed into a constraint:

$$\text{s.t. } \sum_{(u,v)\in E_S} R_{(i,j)}(u,v) \leq L_{\text{max}},\ \forall (i,j) \in E_C \tag{9}$$

where L_{max} is the predefined maximum allowed number of hops for each communication flow $(i,j) \in E_C$. L_{max} should be specified as part of the system requirements.

Some ILP solvers, such as Gurobi [28], support multi-objective optimization given that priorities are assigned to each objective. This functionality enables finding solutions with the smallest average latency under minimized bandwidth requirements.

C. Application Mapping

The assumption that bindings between PEs and switches are known is not always valid, especially in the context of application-specific NoCs, where optimal bindings can be challenging to determine. In this work, application mapping and routing are integrated within a single ILP.

A binary variable $Bd(i,u)$ is defined, indicating whether the NI of PE i in V_C is (bidirectionally) linked to switch u in V_S. $Bd(i,u) = 1$ when PE i and switch u are bound.

While the same objectives can be used, for application-specific NoCs, more complex models are typically involved. An example of a straightforward model used to calculate the runtime communication cost CC found in [26] is

$$\begin{aligned} CC &= \sum_{(i,j)\in E_C} (\text{Number of hops}(i,j) \times w(i,j)) \\ &= \sum_{(i,j)\in E_C} (\sum_{(u,v)\in E_S} R_{(i,j)}(u,v) \times w(i,j)) \end{aligned} \tag{10}$$

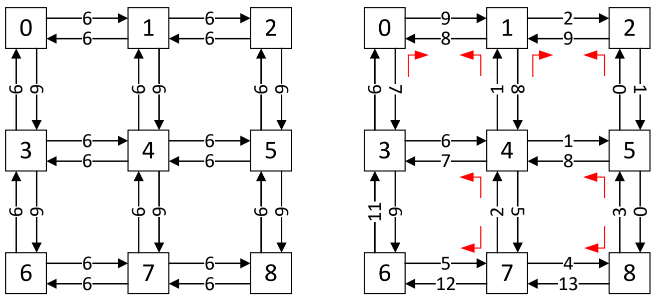
There are other models used to estimate the energy consumption of the NoC, such as those employed in [29], [30]. These models can also serve as objectives of the ILP. The constraints from (3) to (7) remain applicable. Only constraint (2) needs to be replaced, and two more constraints will be added. As a result, the ILP is formulated as follows:

$$\text{Minimize: } CC \tag{11}$$

Subject to:

$$\sum_{v:(u,v)\in E_S} (R_{(i,j)}(u,v) - R_{(i,j)}(v,u)) = Bd(i,u) - Bd(j,u),\ \forall u \in V_S,\ \forall (i,j) \in E_C \tag{12}$$

$$\sum_{u\in V_S} Bd(i,u) = 1,\ \forall i \in V_C \tag{13}$$



(a) Bandwidth requirement

(b) Numbering of links and forbidden turns

Fig. 1. Deadlock-free routing under uniform random traffic on a 3×3 mesh

$$\sum_{i \in V_C} Bd(i,u) \leq 1,\ \forall u \in V_S \tag{14}$$

and (3) to (7)

The constraint (12) defines the communication flow for each $(i,j) \in E_C$. If switch u is bound to PE i, indicating u as a source, the right side of the equation equals 1. If switch u is bound to PE j, indicating u as a drain, the right side becomes -1. For all other cases, it remains 0. This corresponds to the right-hand side of (2) in the previous subsection. The constraint (13) enforces that each PE must be bound to exactly one switch. The c´onstraint (14) restricts that each switch can be bound to maximum of one PE. Notably, these constraints are specifically valid for mesh-based topologies. For other topologies, such as tree-based, where a switch has multiple ports for connecting to the NIs of PEs, the constraints need to be modified accordingly. In the case of mesh-based topology, if the number of switches is greater than the number of PEs, not all switches may be bound to a PE.

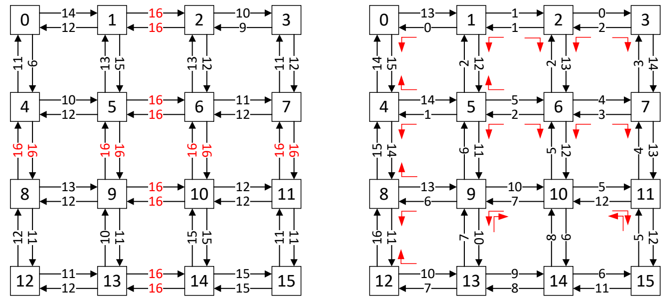
IV. Evaluation

The proposed ILP has been implemented using Python and Gurobi solver v10.0 [28] on 64-bit Linux workstations equipped with an Intel Core™ i9 3.2GHz CPU with 24 physical cores and 128GB of RAM.

A. Topology Analysis

To systematically analyze the deadlock-free routing generated by the ILP, we have constructed a G_C with uniform random traffic, where each PE sends packets to every other PE. For simplicity, the required bandwidth of each communication flow is set to 1. Three switch graphs G_S are created for mesh-based regular topologies: a 3×3 mesh, a 4×4 mesh and a 4×4 torus. The ILP is solved using a multi-objective approach that first minimizes the maximum required bandwidth and then minimizes the latency.

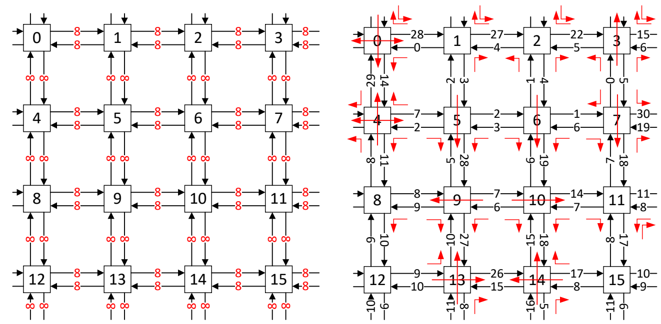
In Fig. 1, the results of the 3×3 mesh is displayed. On the left side, each link is labeled with its required bandwidth, while on the right side, the numbering of each link is displayed along the edge. Routing is only allowed in paths with descending order to prevent deadlocks. The forbidden turns are displayed



(a) Bandwidth requirement

(b) Numbering of links and forbidden turns

Fig. 2. Deadlock-free routing under uniform random traffic on a 4×4 mesh



(a) Bandwidth requirement

(b) Numbering of links and forbidden turns

Fig. 3. Deadlock-free routing under uniform random traffic on a 4×4 torus

with red arrows. The results of the ILP shows an identical routing as in the XY turn model. The routing restrictions are only a subset of those in XY turn model, because only those turns are forbidden that could actually used by communication flows otherwise. This confirms that the XY turn model produces optimal solutions under unified random traffic for the objectives.

The results of 4×4 mesh and torus are presented in Fig. 2 and Fig. 3. The red numbers in the left side serve as indicators of the system's bottleneck. It is important to note that some straight-through connections are considered mathematically as turns and therefore marked as forbidden, such as the turn from switch 1 through switch 5 to switch 9 on the 4×4 torus network. In torus, unlike mesh, straight-through transmissions alone can cause a deadlock, so this must be avoided. The results reveal that, while the center of the mesh NoC requires more bindwidth, routing is evenly distributed on the torus NoC.

It is worth to emphasize that the solver finds only one optimal solution for the ILP, and the results presented in the figures represent just one possible optimal routing under uniform random traffic. The forbidden turns, as depicted, are not the only way to achieve the minimum required bandwidth shown on the left side of the graphs and the smallest average latency. While well-known turn models with evenly distributed

TABLE I
COMPARISON OF COMMUNICATION COST (IN MB/S) FOR BENCHMARK APPLICATIONS USING DIFFERENT ALGORITHMS

Application	Number of PEs	Mesh sizes	DF ILP	NMAP [23]		PSMAP [25]		Kernighan-Lin [26]	
			Comm. cost	Comm. cost	Gap	Comm. cost	Gap	Comm. cost	Gap
PIP	8	4×2	640	640	0%	640	0%	640	0%
PIP	8	3×3	640	–	–	–	–	–	–
MP3	13	4×4	17.021	18.171	6.32%	17.021	0%	17.091	0.41%
MWD	12	4×4	1120	1184	5.40%	1120	0%	1248	10.25%
VOPD	16	4×4	4119	4265	3.43%	4119	0%	4189	1.67%
DVOPD	32	8×4	9570	10253	6.66%	9752	1.87%	9972	4.03%

TABLE II
COMPARISON OF CPU TIME AND COMMUNICATION COST (IN MB/S) FOR BENCHMARK APPLICATIONS USING DF ILP ON DIFFERENT TOPOLOGIES

Application	Number of PEs	Sizes	Mesh		Torus		Hexagonal	
			CPU time	Comm. cost	CPU time	Comm. cost	CPU time	Comm. cost
PIP	8	4×2	0.23s	640	0.05s	640	0.06s	640
PIP	8	3×3	0.09s	640	0.24s	576	0.26s	576
MP3	13	4×4	1.67s	17.021	0.84s	17.021	3.32s	16.521
MWD	12	4×4	0.39s	1120	0.37s	1120	0.57s	1120
VOPD	16	4×4	5.99s	4119	21.52s	4103	8.07s	3731
DVOPD	32	8×4	15381.37s	9570	24297.01s	9506	12267.38s	8826

TABLE III
COMMUNICATION COST (IN MB/S) FOR APPLICATION DVOPD USING DF ILP WITHIN LIMITED CPU TIME

Time limit	Mesh		Torus		Hexagonal	
	Comm. cost	Gap	Comm. cost	Gap	Comm. cost	Gap
600s	10167	5.87%	9888	3.86%	9034	2.30%
1200s	9682	1.16%	9704	2.04%	8842	0.18%
2400s	9650	0.83%	9522	0.17%	8826	0%
3600s	9624	0.56%	9522	0.17%	8826	0%
7200s	9570	0%	9506	0%	8826	0%
Optimum	9570	–	9506	–	8826	–

forbidden turns may also achieve optimal results as confirmed above, the distribution of forbidden turns in the ILP results is uneven.

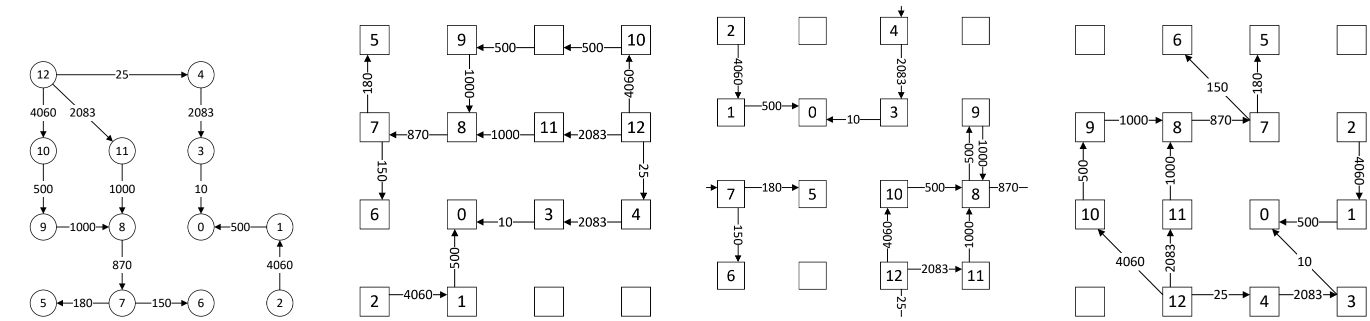
The ILP introduced in this work is not limited to a specific topology, making it versatile for generating and analyzing forbidden turns in NoCs with various topologies. One major advantage is that, unlike turn models where the forbidden turns are defined without considering the specifics of communication flows, the ILP incorporates parameters from the communication flows. This approach considers factors such as varying bandwidth requirements in real-world NoC communications. In cases where bandwidth requirements differ, turn models may not always deliver optimal results because they lack awareness of these variations. Additionally, the ILP can accommodate complex energy models aimed at reducing long-term running costs, a level of flexibility that turn models may not provide. Consequently, the ILP ensures optimal results even in scenarios with varying communication parameters and complex energy considerations.

B. Application Mapping

For application mapping, we have selected several common benchmark applications, including PIP, MP3, MWD, VOPD, and DVOPD [26]. The objective of the ILP is to minimize the communication cost. To evaluate the performance, the results are compared with values provided in [26], using various heuristic algorithms such as NMAP [23], PSMAP [25], and Kernighan-Lin [26]. Please note that these heuristic approaches do not account for deadlock avoidance, potentially having deadlock risks in their results.

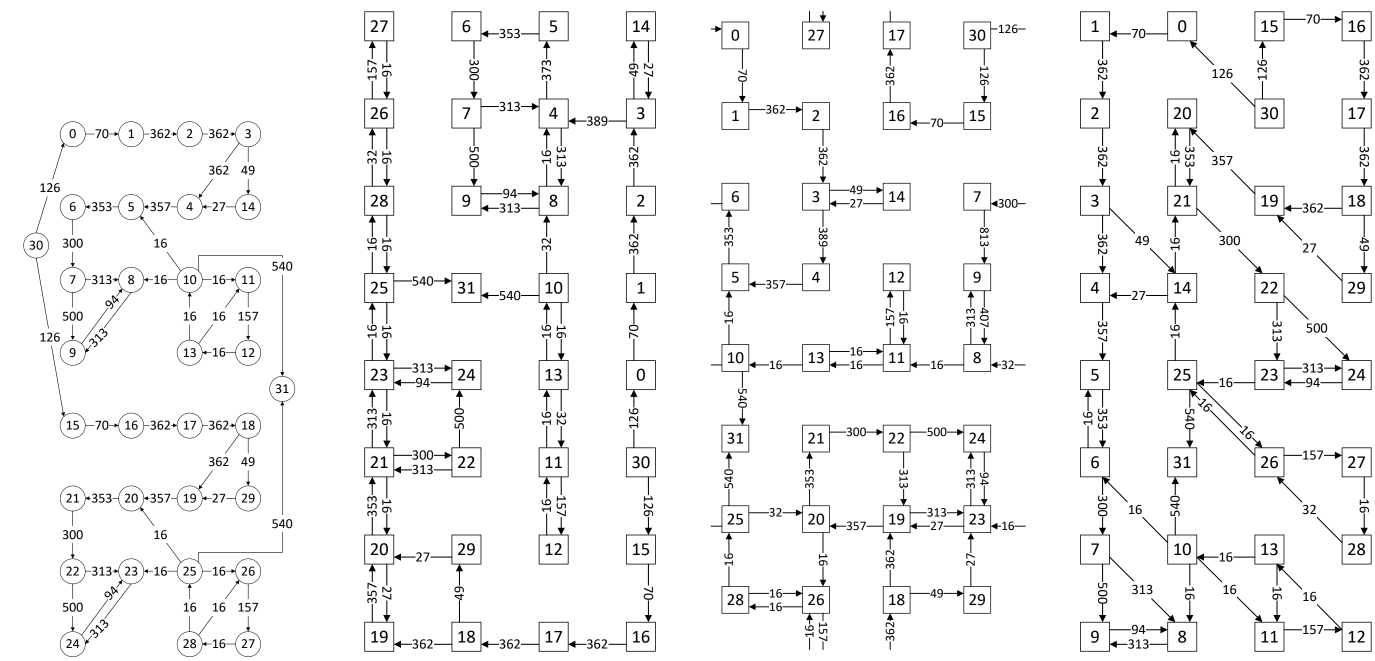
The results are presented in Table I. In the fourth column, the results obtained using our deadlock-free ILP approach are listed. For PIP, we have also created a variant which maps the application on a 3×3 mesh NoC. Notably, the ILP consistently delivers the best results among all the algorithms. The table also shows the gap between the results. It is observed that all algorithms can achieve optimal solutions for PIP, which consists of eight PEs. However, as the number of PEs and communication flows increase, heuristic algorithms start providing sub-optimal solutions. For instance, with DVOPD involving 32 PEs, none of the heuristic algorithms could produce the optimal solution. Due to variations in the computing power of the workstations running these algorithms, we could not directly compare CPU times with the heuristic methods.

In addition to mesh topology, we also applied the deadlock-free ILP to map benchmark applications onto torus and hexagonal NoCs. The results, along with the CPU time using the ILP method, are presented in Table II. For 4×2 PIP and MWD,



(a) Communication flow of MP3 (b) Mapping of MP3 onto mesh (c) Mapping of MP3 onto torus (d) Mapping of MP3 onto hexagonal

Fig. 4. Application mapping and deadlock-free routing of application MP3



(a) Communication flow of DVOPD (b) Mapping of DVOPD onto mesh (c) Mapping of DVOPD onto torus (d) Mapping of DVOPD onto hexagonal

Fig. 5. Application mapping and deadlock-free routing of application DVOPD

all three topology NoCs yield the same communication cost. However, the torus topology NoC demonstrates advantages for scenarios involving 3×3 PIP, VOPD, and DVOPD. While the hexagonal NoC delivers further improved results for MP3, VOPD, and DVOPD.

Figure 4 displays the mapping of the MP3 application with bandwidth specified in kB/s, while Figure 5 shows the mapping of DVOPD with bandwidth specified in MB/s. To enhance clarity, the number labeled on the PEs in G_C shown in (a) are directly labeled on the switches in G_S shown in (b) to (d).

Solving the ILP is a rapid process for PIP, MP3, MWD, and VOPD, typically taking only seconds. However, for DVOPD, the ILP solver takes significantly longer time period, namely 15,381 seconds on mesh, 24,297 seconds on torus, and 12,267 seconds on hexagonal NoCs.

One should note that the ILP approach guarantees optimal results, whereas none of the aforementioned heuristic methods can achieve the same level of optimality. Therefore, employing the ILP is a valuable option, particularly in mass production scenarios where even slight improvements can yield substantial long-term benefits.

Spending a long time to find the optimal solution does not imply that the ILP cannot deliver results comparable to heuristic methods within a limited time frame. Modern ILP solvers often produce good results within a short time, well before reaching the optimal solution. To illustrate this, we applied the same ILP for benchmark application DVOPD with time constraints, and the results are presented in Table III. As shown, the results after 10 minutes are already superior

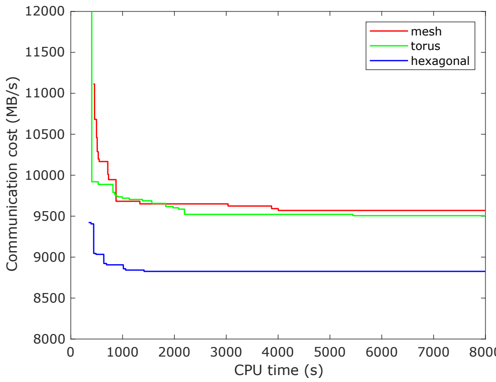


Fig. 6. Results of application DVOPD obtained during the solving process

to NMAP, and within 20 minutes, the ILP surpasses all other methods. After 40 minutes, the gaps to the optimal solution are less than 1%, and after 2 hours, the ILP achieves optimal solutions on all topologies. Fig. 6 illustrates the results obtained during the solving process. In summary, the ILP is capable of delivering significantly better results within an acceptable time period.

V. Fault Tolerance

The growing integration density makes the NoC increasingly vulnerably and many efforts have been made for designing fault tolerant NoCs [31]. There are ILP-based approaches available [17], [20], [21]. However, none of them take deadlocks into consideration. In [20], a fault-tolerant design method considering link faults is proposed. The work [21] extends the approach to also consider switch faults. The principles of these ILP methods can be incorporated with our ILP approach, allowing us to address application mapping, optimal deadlock-free routing and fault tolerance within a single ILP.

Following the approach in [20] with the consideration of link faults, an additional parameter K is introduced, which indicates the degree of fault tolerance. For each communication flow $(i,j) \in G_C$, the goal is to find K disjoint routing paths in G_S.

By introducing an array, denoted as A_K, ranging from 0 to K (e.g. $A_K = [0,1]$ for 1-fault tolerance), the ILP variables are modified to the following:

$R_{(i,j),k}(u,v)$: a binary variable that indicates whether the communication (i,j) in G_C routes through link (u,v) in G_S for the kth disjoint routing path.

$T_{(i,j),k}(u,v,w)$: a binary variable that indicates whether the turn from (u,v) to (v,w) in G_S is in the kth disjoint routing path of (i,j) in G_C.

While the same objectives as in section III are employed for the ILP, it is important to note that under normal circumstances, only the 0th routing paths are utilized. Consequently, optimization efforts can be focused solely on the 0th route to reduce the CPU time for the solving process. Other up-to-Kth routes are reserved for fault cases, hence if no optimization is required, the primary goal is to ensure that they remain deadlock-free.

The ILP is formulated as follows:
Subject to:

$$\sum_{v:(u,v)\in E_S} (R_{(i,j),k}(u,v) - R_{(i,j),k}(v,u)) \\ = Bd(i,u) - Bd(j,u),\ \forall u \in V_S,\ \forall (i,j) \in E_C,\ \forall k \in A_K \tag{15}$$

$$R_{(i,j),k}(u,v) + R_{(i,j),k}(v,u) \le 1, \\ \forall (u,v) \in E_S,\ \forall (i,j) \in E_C,\ \forall k \in A_K \tag{16}$$

$$\sum_{k\in A_K} R_{(i,j),k}(u,v) \le 1,\ \forall (u,v) \in E_S,\ \forall (i,j) \in E_C \tag{17}$$

$$BW(u,v) = \sum_{(i,j)\in E_C} \sum_{k\in A_K} (R_{(i,j),k}(u,v) \times w(i,j)), \\ \forall (u,v) \in E_S \tag{18}$$

$$T_{(i,j),k}(u,v,w) = \min(R_{(i,j),k}(u,v), R_{(i,j),k}(v,w)), \\ \forall v : (u,v),(v,w) \in E_S,\ u \ne w,\ \forall (i,j) \in E_C,\ \forall k \in A_K \tag{19}$$

$$N(u,v) - N(v,w) \ge 1 \text{ if } T_{(i,j),k}(u,v,w) = 1, \\ \forall v : (u,v),(v,w) \in E_S,\ u \ne w,\ \forall (i,j) \in E_C,\ \forall k \in A_K \tag{20}$$

and (13), (14), (5)

Constrains (15) to (16) and (18) to (20) have been adapted and modified from section III to determine K distinct routing paths for each communication flow (i,j) in G_C. A new constraint (17) is introduced to ensure that the K different routing paths are link-disjoint.

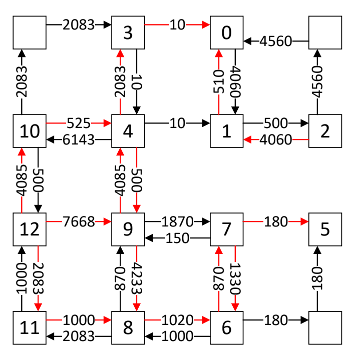


Fig. 7. Application mapping and 1-link-fault-tolerant deadlock-free routing of MP3 application

Fig. 7 presents the result of the ILP for application mapping and 1-link-fault-tolerant deadlock-free routing of MP3 application. Links used by the 0th paths, which corresponds to routing paths in normal situations, are displayed in red. The required bandwidth for each edge takes into consider the routing in both normal and fault scenarios. For any communication flow, there are two link-disjoint routing paths available, with the 0th path being the optimal one. For example, communication from PE 3 to PE 0 is routed through switches 3, 4, 1, and 0 in the fault case. Similarly, communication from PE 12 to PE 4 is routed through switches 12, 9, and 4 in the normal case and through switches 12, 10, and 4 in the fault case.

VI. Conclusion

In this work, we have proposed an ILP method for deadlock-free routing that can be applied on any NoC topology. We have utilized this ILP to systematically construct routing methods for mesh and torus topologies under uniform random traffic. Although we confirm that well-known turn models may achieve optimal results under certain circumstances, the ILP provides solutions alternative to the turn models and ensures optimal results even in scenarios with varying communication parameters and complex energy considerations.

One notable feature of the ILP method is its ability to simultaneously solve application mapping while constructing the deadlock-free routing. We have applied this method to several benchmark applications and compared the results with various heuristic methods, demonstrating that the ILP consistently delivers optimal solutions. It is important to note that while larger NoCs may require significant CPU time to achieve an optimal solution, modern solvers can still provide close-to-optimal results within acceptable time.

Furthermore, the incorporation of fault tolerance techniques allows us to achieve application mapping, optimal deadlock-free, and fault tolerance within a single ILP. As an example, a 1-link-fault-tolerant deadlock-free routing and application mapping for the MP3 application on a mesh network has been demonstrated.

References

[1] W. J. Dally and B. Towles, "Route packets, not wires: on-chip inteconnection networks," *38th Annual Design Automation Conference (DAC)*, pp. 684–689, Las Vegas, NV, 2001.

[2] L. Benini and G. de Micheli, "Networks on chips: a new SoC paradigm," *Computer*, vol. 35, no. 1, pp. 70–78, 2002.

[3] L. Benini and D. Bertozzi, "Network-op-chip architectures and design methods," *IEE Proceedings - Computers and Digital Techniques*, vol. 152, no. 2, pp. 261–272, 2005.

[4] J. Flich, S. Rodrigo, and J. Duato, "An efficient implementation of distributed routing algorithms for NoCs," *2nd ACM/IEEE International Symposium on Networks-on-Chip (NOCS)*, pp. 87–96, Newcastle upon Tyne, UK, 2008.

[5] W. J. Dally and C. L. Seitz, "Deadlock-free message routing in multiprocessor interconnection networks," *IEEE Transactions on Computers*, vol. C-36, no. 5, pp. 547–553, 1987.

[6] W. J. Dally and H. Aoki, "Deadlock-free adaptive routing in multicomputer networks using virtual channels," *IEEE Transactions on Parallel and Distributed Systems*, vol. 4, no. 4, pp. 466–475, 1993.

[7] C. J. Glass and L. M. Ni, "The turn model for adaptive routing," *19th Annual International Symposium on Computer Architecture (ISCA)*, pp. 278–287, New York, NY, 1992.

[8] G. M. Chiu, "The odd-even turn model for adaptive routing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 7, pp. 729–738, 2000.

[9] M. Tang, X. Lin, and M. Palesi, "The repetitive turn model for adaptive routing," *IEEE Transactions on Computers*, vol. 66, no. 1, pp. 138–146, 2017.

[10] C. J. Glass and L. M. Ni, "Fault-tolerant wormhole routing in meshes," *23rd International Symposium on Fault-Tolerant Computing (FTCS)*, pp. 240–249, Toulouse, France, 1993.

[11] J. Wu, "A deterministic fault-tolerant and deadlock-free routing protocol in 2-D meshes based on odd-even turn model," *16th International Conference on Supercomputing (ICS)*, pp. 67–76, New York, NY, 2002.

[12] Y. Cai, D. Xiang, and X. Ji, "Deadlock-free adaptive 3D network-on-chips routing algorithm with repetitive turn concept," *IET Communications*, vol. 14, no. 11, pp. 1783–1792, 2020.

[13] S. Moriam and G. P. Fettweis, "Fault tolerant deadlock-free adaptive routing algorithms for hexagonal networks-on-chip," *Euromicro Conference on Digital System Design (DSD)*, pp. 131–137, Limassol, Cyprus, 2016.

[14] J. Duato, "A necessary and sufficient condition for deadlock-free adaptive routing in wormhole networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 6, no. 10, pp. 1055–1067, 1995.

[15] F. Verbeek and J. Schmaltz, "On necessary and sufficient conditions for deadlock-free routing in wormhole networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 12, pp. 2022–2032, 2011.

[16] S. Tosun, O. Ozturk, and M. Ozen, "An ILP formulation for application mapping onto network-on-chips," *International Conference on Application of Information and Communication Technologies (AICT)*, Baku, Azerbaijan, 2009.

[17] P. V. Bhanu, P. V. Kulkarni, and S. J, "Fault-tolerant network-on-chip design with flexible spare core placement," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 15, no. 1, art. 54, 2019.

[18] P. V. Bhanu and S. J, "Fault-tolerant application mapping on mesh-of-tree based network-on-chip," *Journal of Systems Architecture*, vol. 116, 102026, 2021.

[19] 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, 2006.

[20] Z. Li, J. Huang, Q. Xu, and S. Chen, "Integer linear programming based fault-tolerant topology synthesis for application-specific NoC," *IEEE 12th International Conference on ASIC (ASICON)*, pp. 96–99, Guiyang, China, 2017.

[21] S. Chen, M. Ge, Z. Li, J. Huang, Q. Xu, and F. Wu, "Generalized fault-tolerance topology generation for application-specific network-on-chips," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 6, pp. 1191–1204, 2020.

[22] P. Ghosh, A. Sen, and A. Hall, "Energy efficient application mapping to NoC processing elements operating at multiple voltage levels," *3rd ACM/IEEE International Symposium on Networks-on-Chip (NOCS)*, pp. 80–85, La Jolla, CA, 2009.

[23] S. Murali and G. de Micheli, "Bandwidth-constrained mapping of cores onto NoC architectures," *Design, Automation and Test in Europe Conference and Exhibition (DATE)*, pp. 896–901, Paris, France, 2004.

[24] C. Marcon, A. Borin, A. Susin, L. Carro, and F. Wagner, "Time and energy efficient mapping of embedded applications onto NoCs," *Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 33–38, New York, NY, 2005.

[25] P. K. Sahu, P. Venkatesh, S. Gollapalli, and S. Chattopadhyay, "Application mapping onto mesh structured network-on-chip using particle swarm optimization," *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 335–336, Chennai, India, 2011.

[26] P. K. Sahu, K. Manna, N. Shah, and S. Chattopadhyay, "Extending Kernighan-Lin partitioning heuristic for application mapping onto network-on-chip," *Journal of Systems Architecture*, vol. 60, no. 7, pp. 562–578, 2014.

[27] C. Xu, Y. Liu, P. Li, and Y. Yang, "Unified multi-objective mapping for network-on-chip using genetic-based hyper-heuristic algorithms," *IET Computers & Digital Techniques*, vol. 12, no. 4, pp. 158–166, 2018.

[28] Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, ver. 10.0, 2023.

[29] S. Tosun, V. B. Ajabshir, O. Mercanoglu, and O. Ozturk, "Fault-tolerant topology generation method for application-specific network-on-chips," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 9, pp. 1495–1508, 2015.

[30] W. Zhong, S. Chen, B. Huang, T. Yoshimura, and S. Goto, "Floorplanning and topology synthesis for application-specific network-on-chips," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E96-A, no. 6, pp. 1174–1184, 2013.

[31] M. Radetzki, C. Feng, X. Zhao, and A. Jantsch, "Methods for fault tolerance in networks-on-chip," *ACM Computing Surveys*, vol. 46, no. 1, art. 8, 2013.