INTEGRATION the VLSI journal xxx (xxxx) xxx–xxx

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Lagrangian relaxation-based routing path allocation for application-specific network-on-chips☆

Jinglei Huang, Wei Zhong, Zhigang Li, Song Chen*

Department of Electronic Science and Technology, University of Science and Technology of China, Hefei 230027, China

ARTICLE INFO

Keywords:
Application-specific network-on-chip (ASNoC)
Routing path allocation
Lagrangian relaxation
Deadlock-free

ABSTRACT

The application-specific network-on-chip (ASNoC) has been proposed as a promising solution to address the global communication challenges in terms of nanoscale system-on-chips. However, as the number of cores increases on chips, the power consumption and communication latency present major challenges for routing path allocation in ASNoCs. In this study, given the floorplan results and clustering of cores, we propose an efficient routing path allocation algorithm based on the Lagrangian relaxation for routing the traffic flows while minimizing the power consumption under constraints, such as latency constraints, physical link capacity constraints, and switch port constraints. In addition, we propose a modified shortest path algorithm based on the backtracking method to prevent deadlocks, which is critical for the correct operation of NoCs. Our experimental results demonstrate the effectiveness of the proposed method; the method has power overheads of less than 1% compared with designs that do not support a deadlock removal method.

1. Introduction

The number of cores on a chip is increasing owing to technological advances, and thus, the traditional architectures for on-chip communication, such as bus-based and point-to-point connections, cannot satisfy the performance required for system-on-chips (SoCs). In recent decades, network-on-chips (NoCs) [2,3] have been proposed as a solution to address the global communication challenges in terms of SoC because of their better predictability, lower power consumption, and greater scalability compared with the traditional communication architectures.

NoCs can be designed as regular or custom topologies. Regular topologies, such as a mesh or torus, offer lower power consumption and shorter latency when implemented in general-purpose homogenous multi-core SoCs. Unfortunately, most SoCs comprise heterogeneous cores with large variations in size. Moreover, in [4], application-specific network-on-chips (ASNoCs) were described, which can greatly improve the power, area, and performance of on-chip interconnection networks compared with the regular NoCs.

The synthesis of topologies for ASNoCs is NP-hard [5], so it requires elaborate design processing, which generally involves solving three sub-problems: clustering the cores to determine the switch number, floorplanning for the cores and network components, and allocating the routing paths. In this study, we focus on the problem of routing path allocation for ASNoCs.

Many previous studies have addressed the problem of routing path allocation for ASNoCs. In [6], a two-phase framework was proposed for synthesizing ASNoC topologies, as well as a method based on dynamic programming for routing paths, but they only considered the energy consumption as the cost metric, and not the latency bounds and deadlock-free operation. In [7,8], methods were proposed for synthesizing ASNoCs and for routing path allocation, where they applied Dijkstra's shortest path algorithm to the complete graph, and the edges were weighted by a linear combination of the power and latency to route the paths for traffic flows that minimized the power consumption under latency constraints. However, the constrained shortest path problem is NP-hard and it is not possible to guarantee that the latency constraints in all the traffic flows are met. In addition, these methods do not explicitly avoid deadlocks, which can block communication between cores and even lead to complete network failure. In [9], mixed integer linear programming formulations were presented for the synthesis of ASNoCs and virtual channels were introduced to remove deadlocks. In [10,11], methods were presented based on the turn prohibition algorithm to remove deadlocks. However, the prohibited

☆ This study was extended from the paper published at ASICON 2015 [1], and it was supported by the National Natural Science Foundation of China (NSFC) under grant No. 61674133, 61732020 and 61404123, and Anhui Provincial Natural Science Foundation (1508085MF134, China). The authors would like to thank the Information Science Laboratory Center at USTC for hardware and software services.

* Corresponding author.
E-mail addresses: huangjl@mail.ustc.edu.cn (J. Huang), songch@ustc.edu.cn (S. Chen).

https://doi.org/10.1016/j.vlsi.2017.10.011
Received 20 June 2017; Received in revised form 30 September 2017; Accepted 24 October 2017

turns are pre-built before the routing paths phase and the application-specific communication patterns are not considered, which may increase the routing distance between heavily communicated cores, thereby increasing the link power consumption. In [12], a method was proposed that splits some routers to break cycles when routing deadlock occurs. Amit et al. [13] presented a binary search cheapest bounded path algorithm for routing traffic flows, which guarantees high performance of processing elements satisfy the communication latency constraints. However, the physical link capacity constraints and switch ports constraints are not considered.

In the present study, given the floorplan results and clustering of cores, we propose an efficient routing path allocation algorithm based on the Lagrangian relaxation for routing the traffic flows while minimizing the power consumption under constraints, such as latency constraints, physical link capacity constraints, and switch port constraints. A modified shortest path algorithm is also integrated in the proposed method to find deadlock-free routing paths using the backtracking method.

In the remainder of this paper, some important definitions and the problem formulation are presented in Section 2. Section 3 introduces the Lagrangian relaxation-based routing path allocation method. The experimental results and conclusions are presented in Sections 4 and 5, respectively.

2. Problem formulation

The ASNoC architectures that we generate are assumed to support packet-switched communications, with source routing and wormhole flow control. We introduce an efficient routing path allocation algorithm based on Lagrangian relaxation to route traffic flows.

Fig. 1 shows the design flow for routing path allocation in ASNoCs. The input for our routing path allocation problem is a $G_{CC}(C, E)$ (core communication graph, which represents the traffic characteristics of the application), where each vertex $c_i \in C$ represents a core and each directed edge $(c_i, c_j) \in E$ represents the communication from core c_i to c_j, and the corresponding communication requirement and latency constraint (the maximum number of switches on the routing path) are given by w_{c_i,c_j} and l_{c_i,c_j}, respectively. In addition, the floorplan results and clustering of cores are also given as inputs, which we obtained from [8,14].

The output of routing path allocation is a custom NoC topology with pre-determined deadlock-free routing paths for traffic flows under the constraints, such as latency constraints, physical link capacity constraints, and switch size (number of ports) constraints.

Given the floorplan and clustering of cores, where the cores, C, are partitioned into n_{sw} subsets, $clu_i \subseteq C$, $1 \le i \le n_{sw}$, such that

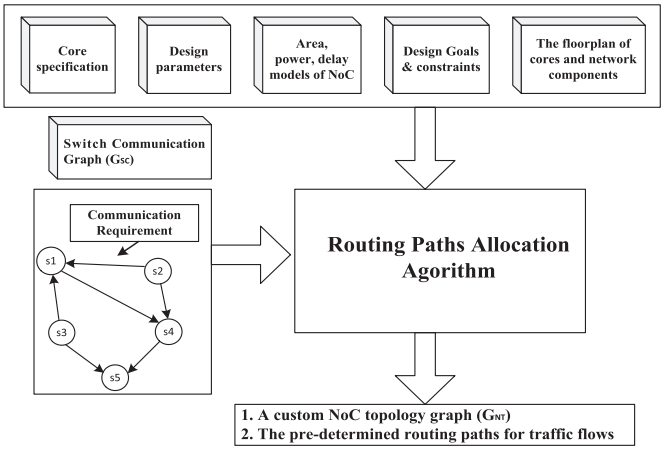

Fig. 1. Routing path allocation for ASNoCs.

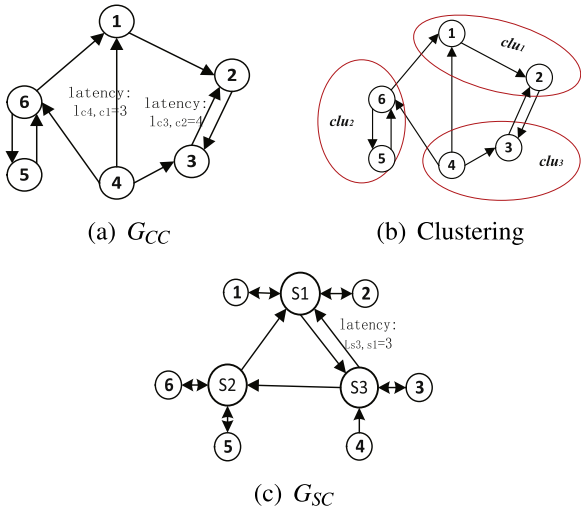

Fig. 2. G_{CC}, clustering, and the corresponding G_{SC}.

$clu_i \cap clu_j = \varnothing$ and $\bigcup_{i=1}^{n_{sw}} clu_i = C$, we can construct a switch communication graph (G_{SC}) to represent the communication requirements between the switches by assigning one switch to each cluster for global communication.

Definition 1. Switch communication graph G_{SC}: a directed graph $G_{SC} = (S, F)$, where $S = \{s_i | 1 \le i \le n_{sw}, s_i$ represents the switch shared by all the cores in the cluster $clu_i\}$, and $F = \{(s_i, s_j) | s_i, s_j \in S$, and there is a traffic flow from switch s_i to $s_j\}$.

The communication requirement for traffic flow (s_i, s_j) is calculated as follows.

$$ws_{s_i,s_j} = \sum_{\forall c_k \in clu_i} \sum_{\forall c_l \in clu_j} w_{c_k,c_l} \tag{1}$$

Fig. 2 shows a simple example to illustrate the procedure for constructing G_{SC}. A simple G_{CC} with six cores is shown in Fig. 2(a). Fig. 2(b) and Fig. 2(c) show the clustering of cores with $n_{sw} = 3$ and the corresponding G_{SC}, respectively, where the edges (s_1, s_3), (s_2, s_1), (s_3, s_1), and (s_3, s_2) represent four traffic flows.

The traffic flows in the same cluster can easily be routed and communicate with each other via the corresponding switch, so the routing path allocation process focuses on finding the routing paths for the traffic flows F in G_{SC}.

The latency constraint on a traffic flow $(s_i, s_j) \in F$, denoted as L_{s_i,s_j}, can be calculated as follows:

$$L_{s_i,s_j} = \min\{l_{c_m,c_n} | (c_m, c_n \in C) \wedge ((c_m, c_n) \in E) \wedge (c_m \in clu_i) \wedge (c_n \in clu_j)\} \tag{2}$$

For example, in Fig. 2, we assume that the latency constraint l_{c_4,c_1} of $(c_4, c_1) \in E(G_{CC})$ is 3, and the latency constraint l_{c_3,c_2} of $(c_3, c_2) \in E(G_{CC})$ is 4. Therefore, the latency constraint $L_{s_3,s_1} = 3$ for traffic flow (s_3, s_1) between switch s_3 and s_1 in G_{SC} as shown in Fig. 2(c) can be obtained.

Note that the edges in G_{SC} do not denote the actual physical connections between the switches, but instead they represent the communication requirements between the switches. The actual physical connectivity between the switches is established during the procedure of routing path allocation for traffic flows.

Our routing path allocation algorithm is performed on a switch routing graph G_{SR}, which is a complete graph and the edges represent the possible routing channels for traffic flows, and is defined as follows:

Definition 2. Switch routing graph G_{SR}: a complete graph $G_{SR} = (S, A)$, where each vertex $s_i \in S$, $1 \le i \le n_{sw}$, represents a switch and each directed edge $ch_{s_i,s_j} = (s_i, s_j) \in A$ represents a possible routing channel from switch s_i to s_j, which has an initial bandwidth capacity $f_{cap}(s_i, s_j)$