Towards a Unified Framework for the Evaluation and Optimization of NoC Application Mapping Algorithms

Ciprian Radu*, Lucian Vinţan*

* *Advanced Computer Architecture & Processing Systems Research Lab - http://acaps.ulbsibiu.ro/research.php, "Lucian Blaga" University of Sibiu, Romania*

ABSTRACT

This work proposes a unified framework for Network-on-Chip (NoC) application mapping algorithms' evaluation and optimization. This tool allows multiple application mapping algorithms to be tested on the same NoC design, which leads to a more thorough evaluation of their impact on the network's performance. The framework is also intended to be flexible so that different NoC designs can be used when comparing the performance of different algorithms. This could contribute to the optimization of certain algorithms, for specific NoC architectures.

KEYWORDS: Network-on-Chip (NoC); parallel application NoC mapping; simulation, evaluation and optimization; software framework

1 Introduction

The application mapping problem is defined as the topological placement of the Intellectual Properties (IPs) onto the tiles of a Network-on-Chip. Application mapping algorithms were proposed for determining the best placement of IP cores onto network nodes. Network performance metrics like bandwidth, latency and energy consumption are globally considered when searching for the optimal mapping. Before the IPs can be mapped onto network nodes, the assignment of tasks to heterogeneous IP cores must be determined. This is why the application mapping problem is related to the scheduling problem. Also, better mapping solutions can be found if network routes are considered, too.

[1]E-mail: {ciprian.radu,lucian.vintan}@ulbsibiu.ro

[2]This work was partially supported by CNCSIS no. 485/2008 research grant offered by the Romanian National Council for Academic Research.

In this paper we propose using a unified framework for the evaluation and optimization of different application mapping algorithms, on multiple NoC designs (e.g.: application mapping algorithms are mainly evaluated on 2D meshes).

2 The Framework Design

The major components of our unified framework are presented in [RV10]. However, Figure 1 provides a more detailed description of the developed framework. With a Communication

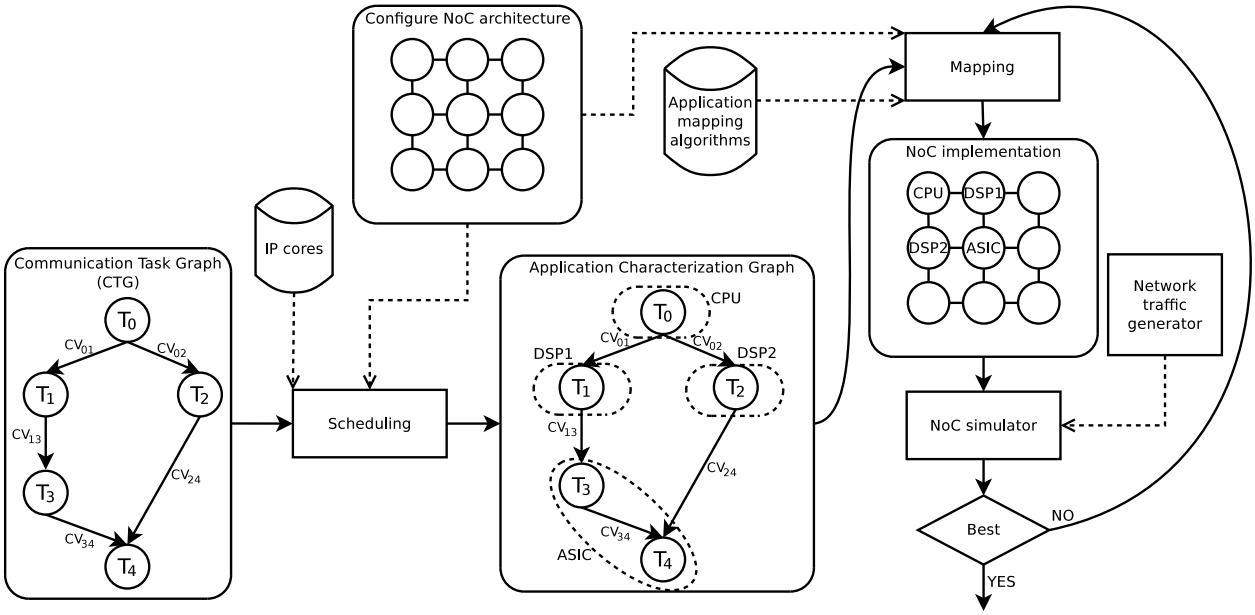

Figure 1: The framework design flow

Task Graph (CTG), an application is partitioned into a set of concurrent threads. A CTG also specifies the data dependencies among these threads and how much volume of information is communicated from a thread to another (e.g.: CV_{01} denotes the communication volume from thread T_0 to thread T_1). We propose obtaining CTGs in three distinct ways: (1) randomly, by using the TGFF (`http://ziyang.eecs.umich.edu/~dickrp/tgff/`) tool, (2) from realistic embedded applications, using the E3S benchmark suite (`http://ziyang.eecs.umich.edu/~dickrp/e3s/`) and (3) from real-world multithreaded applications, using the CETA (`http://ziyang.eecs.umich.edu/projects/ceta/`) tool.

The threads must be first assigned to the heterogeneous IP cores. This is typically done with a scheduling algorithm. Besides simply assigning threads to IPs, a scheduling algorithm also determines the threads' execution order. This is useful when dealing with real-time constraints. Since the main goal of this framework is to address the mapping problem, we propose using only a specific scheduling algorithm. The Energy-Aware Scheduling (EAS) [HM05] algorithm is able to perform scheduling under real-time restrictions, while trying to optimize the energy consumption of the NoC architecture. A library of IP cores will be available in the framework. For each IP, information like thread execution time and power consumption for a given thread are known. Such an IP library is provided by the E3S benchmark suite. However, when building CTGs with CETA, the information metrics associated to the processing core (thread execution time, power consumption) must be measured.

The output of the scheduling algorithm will be an Application Characterization Graph (ACG). Compared to the CTG, the ACG specifies the assignment of application threads to IP cores and it represents the input for the mapping phase. A main component of the framework will consist in a library containing known application mapping algorithms' implementations. The performance of every mapping algorithm can be evaluated on multiple NoC designs. The NoC simulator is another important part of the unified framework. An important aspect of the simulator consists in its flexibility. This will impact on the number of possible ways in which the simulated NoC can be configured. The simulator is also responsible with determining the network's performance represented through multiple objectives (performance, energy consumption, etc.). This allows a thorough comparison of the mapping algorithms, in a unified manner. For each selected network design (e.g.: the network topology can be varied), an application mapping algorithm will provide multiple mappings, until the best mapping is determined. The NoC simulator will run by generating network traffic which emulates the communicational behavior of the application.

2.1 Network Traffic Generation

Our unified framework will emulate the communication between the threads of the application by modeling each Processing Element (PE) from the Network-on-Chip as the Finite State Machine (FSM) described in Figure 2. Initially, all the Processing Elements are in state 1,

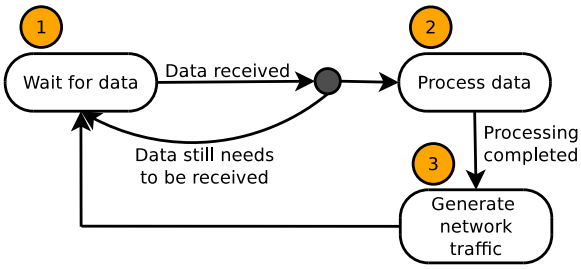

Figure 2: The FSM associated to a PE

waiting to receive data to be processed. The first PE which will enter in the processing phase (state 2) is the one that contains the root task of the CTG. Any PE enters in the processing state after it has received all the data from the other PEs it depends on (data dependencies are modeled by the CTG). Any PE will stay in this state for a period of time equal to the time needed by the IP to process the task. After this time is elapsed, it means that the task's processing is finished, and the PE enters in state 3. At this point, the processed data is injected into the network. The communication volume is specified by the CTG. All these data will be divided into packets, having a given size.

2.2 The ns-3 NoC Simulator

The ns-3 NoC is a flexible Network-on-Chip simulator developed for this framework. It is based on ns-3 (`http://www.nsnam.org`), a scalable simulator for Internet systems, one of the fastest and most memory efficient simulators currently available. Our NoC simulator currently allows the user to specify: the packet size, packet injection rate, buffer size, network size, switching mechanism (Store-and-Forward, Virtual Cut-Through and Wormhole),

routing protocol (XY, YX and two adaptive protocols that consider the network's load), network topology (currently, only variations of a 2D mesh) and traffic patterns. It can evaluate the simulated NoC in terms of network latency and throughput. Figure 3 presents the main

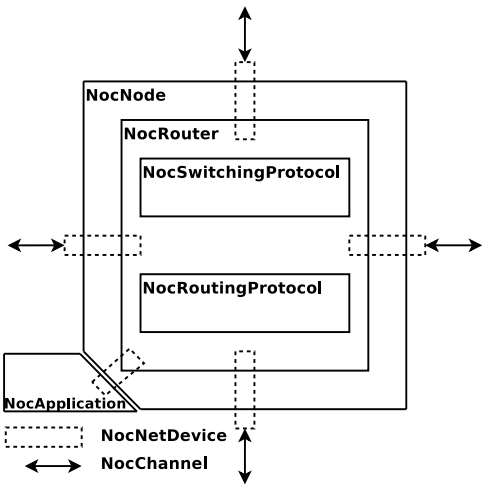

Figure 3: The basic architecture of the ns-3 NoC simulator

components of the ns-3 NoC simulator. The **NocApplication** models a Processing Element, being thus responsible with injecting packets into the network. Packets can be inserted into the NoC synchronously or asynchronously, with a specified injection probability. **NocNode** models the network node. It contains a router architecture. The **NocRouter** aggregates the switching mechanism (specified by **NocSwitchingProtocol**) and the routing protocol (represented by **NocRoutingProtocol**). The **NocNetDevice** represents a network interface, by connecting a network node to a channel. It provides input channel buffering. The **NocChannel** models a bidirectional communication channel. Further details about ns-3 NoC are available in [RV10].

3 Conclusions and Further Work

In this short paper we have presented a preliminary research towards creating a unified framework for the evaluation and optimization of NoC application mapping algorithms. Further work involves the implementation of some application mapping algorithms and a network traffic generator based on CTGs. Also the ns-3 NoC will be further developed so that it will allow more NoC architectures to be simulated.

References

[HM05] J. Hu and R. Marculescu. Communication and task scheduling of application-specific networks-on-chip. *IEE Proceedings - Computers and Digital Techniques*, 152(5):643, 2005.

[RV10] Ciprian Radu and Lucian Vinţan. Optimizing application mapping algorithms for NoCs through a unified framework. In *Proceedings of the 9-th IEEE RoEduNet International Conference*, Sibiu, Romania, June 2010. IEEE Computer Society.