

# A Mapping Algorithm for Real-Time Applications on Network-on-Chips

Wenjia Wang

School of Economics, Eastern Liaoning University, Dandong, Liaoning, China.

**Abstract**—Networks-on-chips connect multiple cores together and have been widely used in electronic devices. This paper focuses on the mapping problem of networks-on-chips. We propose an algorithm MART to improve the real-time performance of applications and minimize the communication cost. Experimental results show that RTMAP can significantly improve the acceptance rate of real-time applications which only introduces small communication cost overhead.

**Keywords**—networks-on-chip; multicore; real-time performance; mapping

## I. INTRODUCTION

Multicores have been widely used in electronic devices. These multiple cores coordinate with each other to implement an entire function. For example, in multimedia applications [1], audio, images and video are received, decompressed and played in order on the special cores. We call the process as a task and each process as a subtask.

Networks-on-Chip (NoC), which connects cores together by networks and offers packet switched communication among cores, provides very high on-chip communication bandwidth [2]. It has been widely accepted that the NoC paradigm will be the default design choice for future multi-core processors.

In NoCs, due to the network-like architecture feature, a critical step in the system design is the application mapping, i.e., to decide how to allocate each task of the application on the NoC processing elements. When tasks are allocated on different locations, the sources and destinations of the communications would be changed accordingly. Since the mappings between tasks and processing elements can interfere the communications, we can use some mapping algorithms to optimize the relationship among different communications, and further improve the real-time performance of the applications. However, previous mapping algorithms [3-6] do not consider how to improve real-time performance of applications. Therefore, in this paper, we propose a mapping algorithm MART that can improve the real-time performance of applications and minimize the total communication cost. Experimental results show that comparing with the classical algorithms, the algorithm MART significantly improves the real-time performance of applications, which only introduces small communication cost overhead.

## II. PROBLEM MODEL

We consider a  $n \times n$  2D mesh network  $A = \langle N, L \rangle$ .  $N$  is the set of nodes, where each  $n_i \in N$  is a node in the 2D-mesh NoC.

$L: N \times N$  denotes the set of links, where each  $lij \in L$  is a directed edge from  $n_i$  to  $n_j$ . The bandwidths of all links are equal, denoted by  $BW$ .

We use  $P = (L)^*$  to denote the set of paths. The path  $pab$  from  $n_a$  to  $n_b$  is an ordered sequence of links  $\{l_{ai}, l_{ij}, \dots\}$  such that: (1) each pair of conjunctive links in the path share a node; (2) any node cannot appear more than once in a path, which means any path does not contain a circle. We use  $qab$  to denote the set of nodes which  $pab$  passes.

Note that, this paper focuses on 2D mesh network, but our algorithm does not impose any restriction on network topology: mesh, torus, ring, tree and irregular topologies can all be supported.

We consider an application consisting of several tasks. Each task exclusively execute on an unique node, i.e., we aim at one-to-one mapping between tasks on nodes. The communication between two tasks transfers data from one task to another. A real-time application is characterized by a directed acyclic graph  $G = \langle T, C \rangle$ : (1)  $T = \{t_0, t_1, \dots\}$  is the set of tasks. Each  $t_i$  is characterized by  $t_i = \langle e_{ti}, p_{ti} \rangle$ , where the two symbols denote execution time and period. (2)  $C: T \times T$  describes the direct communication. Each element  $c_{ij}$  in  $C$  represents the on-chip communication between task  $t_i$  and  $t_j$ . Each  $c_{ij}$  is characterized by  $c_{ij} = \langle w_{cij}, p_{cij} \rangle$ . They denote the workload and period of the communication. The execution time of  $c_{ij}$  is calculated as:  $e_{ijc} = w_{cij} / BW$ .

Each task and communication must complete within its period, i.e. relative deadlines are equal to periods. The network utilization  $u_{cij}$  of  $c_{ij}$  is calculated as  $e_{ijc} / p_{cij}$ . Similarly, the processing element utilization is  $u_{tij} = e_{tij} / p_{tij}$ . Then according to the sufficient conditions for scheduling communications [7], we find a mapping function  $M: T \rightarrow N$  that enables the communication set to be minimized the communication cost. In the following, we present the optimization objective and constraints of this mapping problem. The optimization objective is to minimize the total communication cost:

$$\text{Min : Cost} = \sum_{\forall c_{ij} \in C} w_{ij}^c \times \alpha(M(t_i), M(t_j)) \quad (1)$$

Where  $\alpha(M(t_i), M(t_j))$  represents the Manhattan Distance between the two nodes where  $t_i$  and  $t_j$  is mapped to respectively. The optimization problem must respect the following constraints:

(1) Task Mapping Constraint: We regulate that each node can only host at most one task.

$$\forall t_i \neq t_j \in T : M(t_i) \neq M(t_j) \quad (2)$$

(2) Task Utilization Constraint: The utilization of each task must be smaller or equal to 1. Otherwise the task cannot be executed on one node.

$$\forall t_i \in T : u_i^t \leq 1 \quad (3)$$

(3) Link Utilization Constraint: We regulate that the utilization of each link must satisfy the sufficient scheduling condition [7]. Otherwise, the communication cannot be scheduled.

$$\forall D_j \subset C : u_j^D \leq \frac{L-1}{L} \quad (4)$$

### III. MAPPING ALGORITHM

Due to each task exclusively executes on a unique node,  $t_i$  can be executed under real-time constraint if the utilization satisfies Equation (3). So the major consideration of MART is how to satisfy the real-time performance of communications while minimizing total communication cost.

The application mapping problem on NoCs is in general intractable (NP-hard). Therefore we propose a heuristic algorithm MART to solve the problem. Algorithm 1 describes the algorithm MART. In line 1, the communication utilization of each task is calculated. We define the reachable distance of a node to be the sum of the distance between this node and all other nodes in the NoC. In line 2, we calculate the reachable distance  $\xi_i$  of each node. MART maps the task with the largest communication utilization to the location with the longest reachable distance (line 3). Then the task is removed from  $T$  to  $\Gamma$  which is the set of tasks that have been mapped. At the beginning of MART,  $\Gamma$  is empty. The node which has been allocated is deleted from  $N$  (line 4). In line 5-14, each task is mapped to one node. First the task that communication utilizations most with the already mapped tasks is selected (line 6 and 7).  $h_i$  denotes the heuristic value when task  $t_{max}$  is mapped to  $n_i$ . Then the node with the smallest heuristic value is allocated to  $t_{max}$  (line 11). In line 13, the utilization of all links is updated since the link utilization is used in function  $HeuristicValue()$ . After all tasks are mapped, we determine whether the mapping can satisfy the schedulability conditions (line 15-17). If the conditions cannot be satisfied, MART finishes. The selection of  $h_i$  makes the utilizations of all links are as balanced as possible, so in general the initial mapping (line 1-14) can satisfy these condition well. In line 18-32, the major task is to optimize the total communication cost. We find the mapping with the smallest communication cost by pair-wise swapping of tasks.  $GetCost(M)$  calculate the total communication cost of the mapping  $M$  according to the definition of communication cost (Equality (1)).

$HeuristicValue()$  returns the biggest utilization of links along the paths between  $t_{max}$  and mapped tasks, when  $t_{max}$  is mapped to  $n_i$ . In line 1-9, the communications whose source is  $t_{max}$  are calculated. And in line 10-12, it is the communications with the destination  $t_{max}$ . In line 3, the communications from  $t_{max}$  to other mapped tasks is routed on the NoC and the routing algorithm  $Dijkstra()$  is similar to the Dijkstra's shortest path routing. The different between these two routing algorithms is the selection of link weights. In  $Dijkstra()$ , the weight is the link utilization since MART focuses on the schedulability of real-time applications.  $ulgk$  denotes the utilization of  $lgk$ . If the communication  $c_{max;j}$  passes  $lgk$ ,  $sgk$  is the updated utilization of  $lgk$ .

The function  $HeuristicValue()$  finds the biggest link utilization as the heuristic value  $h_i$  of  $n_i$  when  $t_{max}$  is mapped to  $n_i$ . Algorithm 1 invokes this function to calculate  $h_i$  for each  $n_i$  and searches the smallest one among all  $h_i$ . This means that MART minimizes the maximum link utilization, and thus the utilizations of all links are as balanced as possible.

### IV. EXPERIMENTAL RESULTS

We compare MART with NMAP [7]. NMAP is a fast heuristic mapping algorithm that optimizes bandwidth and packet latency. It does not consider the real-time performance of applications. Two metrics are used in the comparison: acceptance rate and communication cost. We generate 1000 random applications at each measurement point. The number of tasks is equal to the number of nodes in the NoC. The communications between tasks are randomly generated and the workload of each communication is a random number. We use the maximal communication utilization ( $maxu$ ) as a controlled variable. The utilization of each communication is randomly generated in the range from 0 to  $maxu$ . In the following experiments  $L$  is equal to 10.

Figure 1 shows the acceptance rates of NMAP and MART with various  $maxu$ . The size of NoC is  $6 \times 6$ . We can see that with our mapping algorithm MART the acceptance rate improvement over NMAP is significant, since the real-time performance of applications is a key consideration in MART. When  $maxu > 0.2$ , the acceptance rates of these two methods begin to decrease. And at  $maxu = 0.5$ , the acceptance rate of NMAP is equal to 0, however our algorithm MART is 0.38.

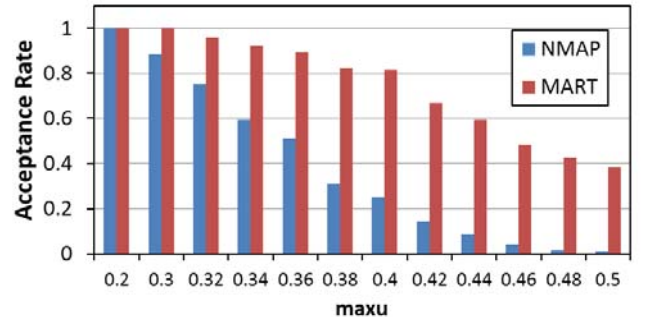


FIGURE I. ACCEPTANCE RATE WITH THE VARYING MAXU.

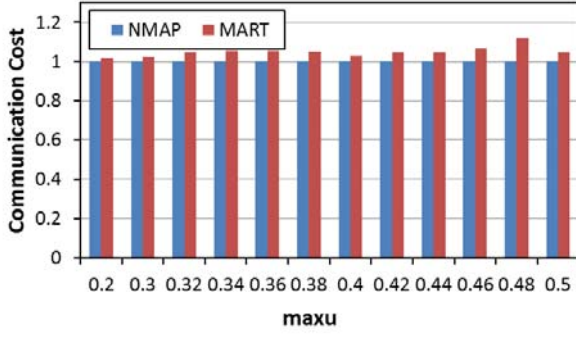


FIGURE II. COMMUNICATIONCOST WITH THE VARYING MAXU

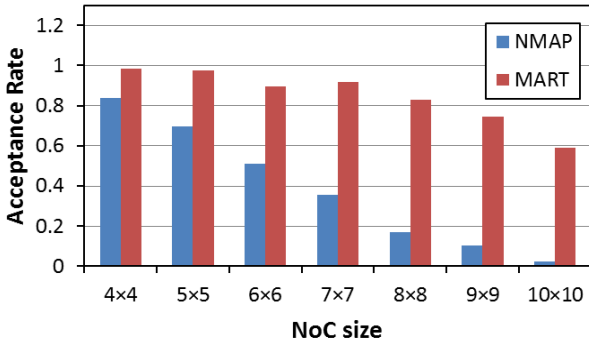


FIGURE III. ACCEPTANCE RATE WITH THE VARYING NETWORK SIZE

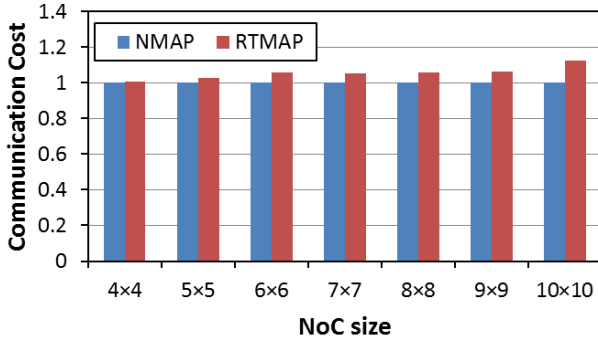


FIGURE IV. COMMUNICATION COST WITH THE VARYING NETWORK SIZE

---

#### Algorithm 1 MAST

---

**Require:**  $A, G$

**Ensure:**  $M : T \mapsto N$

```

1:  $\forall t_i \in T, \mu_i = \sum_{\forall c_{ij} \in C} u_j^t + \sum_{\forall c_{ki} \in C} u_k^t$ ;
2: calculate the reachable distance  $\xi_i$  of  $\forall n_i \in N$ ;
3:  $M(\arg(\max_{\forall t_i \in T}(\mu_i))) = \arg(\max_{\forall n_i \in N}(\xi_i))$ ;
4: remove  $\arg(\max_{\forall t_i \in T}(\mu_i))$  from  $T$  to  $\Gamma$ ,
   delete  $\arg(\max_{\forall n_i \in N}(\xi_i))$  from  $N$ ;
5: while  $T \neq \emptyset$  do
6:    $\forall t_i \in T, \sigma_i = \sum_{\forall c_{ij} \in C, t_j \in \Gamma} u_{ij}^c + \sum_{\forall c_{ji} \in C, t_j \in \Gamma} u_{ji}^c$ ;
7:    $t_{maxt} = \arg(\max_{\forall t_i \in T}(\sigma_i))$ ;
8:   for each  $n_i \in N$  do
9:      $h_i = \text{HeuristicValue}(t_{maxt}, n_i)$ ;
10:  end for
11:   $M(t_{maxt}) = \arg(\min_{\forall n_i \in N}(h_i))$ ;
12:  remove  $t_{maxt}$  from  $T$  to  $\Gamma$ , delete  $\arg(\min_{\forall n_i \in N}(h_i))$  from  $N$ ;
13:  update the utilization  $u_{gk}^l$  of all links;
14: end while
15: if  $M$  cannot satisfies Equation (4) then
16:   return FAIL;
17: end if
18:  $cost^{min} = \text{GetCost}(M)$ ;
19: for each  $i = 1$  to  $|T|$  do
20:   for each  $j = 1$  to  $|T|$  do
21:      $M' \leftarrow M$  swapping  $M(t_i)$  and  $M(t_j)$ ;
22:     if  $M'$  cannot satisfies Equation (4) then
23:       continue;
24:     end if
25:      $cost^{cur} = \text{GetCost}(M')$ ;
26:     if  $cost^{min} > cost^{cur}$  then
27:        $cost^{min} = cost^{cur}$ ;
28:        $M^{cur} \leftarrow M'$ ;
29:     end if
30:   end for
31:    $M \leftarrow M^{cur}$ ;
32: end for
```

---



---

#### Algorithm 2 HeuristicValue

---

**Require:**  $t_{maxt}, n_i, \Gamma, \forall u_{gk}^l$

**Ensure:**  $h_i$

```

1: for each  $c_{maxt,j} \in C$  do
2:   if  $t_j \in \Gamma$  then
3:      $p_{n_i, \mathcal{M}(t_j)} = \text{Dijkstra}(n_i, \mathcal{M}(t_j))$ ;
4:     for each  $l_{gk} \in p_{n_i, \mathcal{M}(t_j)}$  do
5:        $s_{gk} = u_{gk}^l + u_{maxt,j}^c$ ;
6:     end for
7:      $r_{maxt,j} = \max_{\forall l_{gk} \in p_{n_i, \mathcal{M}(t_j)}}(s_{gk})$ ;
8:   end if
9: end for
10: for each  $c_{j,maxt} \in C$  do
11:   //same as the line 2-8;
12: end for
13: return  $\max_{\forall r_{ab}}(r_{ab})$ ;
```

---

Figure 2 shows the normalized communication cost of Figure 1. We can see that our mapping algorithm MART only leads to a slightly larger communication cost (The average loss of communication cost is 5.0% compared to NMAP). This is

the price we paid for considering the real-time performance of applications in the mapping.

Figure 3 shows the acceptance rates of NMAP and MART with various NoC size. The max is 0.36. We can see that the acceptance rate of MART is still higher than that of NMAP especially when NoC size is higher. When the acceptance rate of NMAP is almost equal to 0, our proposed algorithm MART is still 0.6. The reason is that in these situations, there are more routing path overlaps. Therefore, if mapping algorithm does not consider the utilizations of links, some of them may be very high.

Figure 4 shows the normalized communication cost of Figure 3. MART only leads to a slightly larger communication cost (on average 5.6%). The acceptance rate improvement of MART is significant, so the slightly cost overhead is acceptable.

## V. CONCLUSIONS

In this paper, we propose an algorithm MART to solve the mapping problem of real-time applications on NoCs. The mapping algorithm MART not only improves the real-time performance of applications but also minimizes the communication cost. Experimental results show that RTMAP can significantly improve the acceptance rate of real-time applications which only introduces small communication cost overhead.

## ACKNOWLEDGMENT

This work was supported by the Humanities and Social Science Foundation for youth of Ministry of Education of China (No. 15YJC630100); and the Humanities and Social Science Foundation of Liaoning Province (No. L14CGL039).

## REFERENCES

- [1] A. Pakarinen: Multi-core Platforms for Audio and Multimedia Coding Algorithms in Telecommunications, Master's Thesis (2012).
- [2] W. J. Dally and B. Towles, Route Packets: Not Wires: on-chip Interconnection Networks, in proceedings of Design Automation Conference (2002), pp. 648-649.
- [3] S. Murali and G. D. Micheli, Bandwidth-Constrained Mapping of Cores onto NoC Architectures, Design, Automation and Test in Europe, 2004.
- [4] J. Hu and R. Marculescu. Energy-aware mapping for tile-based noc architectures under performance constraints. In Proceedings of Asia and South Pacific Design Automation Conference, 2003.
- [5] J. Hu and R. Marculescu. Exploiting the routing flexibility for energy/performance aware mapping of regular noc architectures. In Design, Automation and Test in Europe Conference and Exhibition, 2003.
- [6] J. Hu and R. Marculescu. Energy-and performance-aware mapping for regular noc architectures. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 24(4):551–562, 2005.
- [7] B. Bui, M. Caccamo and R. Pellizzoni: A Slot-based Real-time Scheduling Algorithm for Concurrent Transactions in NoC, in proceedings of Embedded and Real-time Computing Systems and Applications (2011), pp. 329-338.