TABLE I. VARIABLES

Item	explanation
m_{ik}	task t_i is mapped to p_k
n_{ij}	task t_j is executed next to t_i on the same processor
s_i	the start time of executing task t_i
f_i	the end time of executing task t_i
se_{ij}	the start time of data transfer from t_i to t_j
fe_{ij}	the finish time of data transfer from t_i to t_j
d_{ij}	task t_j depends on the output of t_i
o_{ij}	task t_i needs to send data to its successor
P_{ij}	the contention probability of data transfer from two tasks

TABLE II. CONSTANTS

Item	explanation
τ	the time of transferring a unit with unit distance
τ'	the time of transferring a unit through a node
ϵ	the energy of transferring a unit with unit distance
ϵ'	the energy of transferring a unit through a node
a_i	the amount of computation for task t_i
c_i	the amount of data to be transferred from task t_i
q_i	the required size for data storage of t_i
dl_{ij}	maximal time allowed between executing task t_i and t_j
ρ_k	processing speed of p_k
ω_k	the memory limit for p_k
η	the cost of communication setup
ξ	the time bound for simultaneous communication
N	the number of iterations for all the tasks

The contribution of the paper is as follows. First, we evaluate the allocation strategy in executing the tasks for a certain number, instead of once execution of every task. This allows a more accurate evaluation. Second, as a task can be periodically executed, the deadline is not valid any more. We put forward the concept of latest response time between two tasks, to meet the requirements of time-critical systems. Third, we consider both time and energy optimization in the resource allocation process, where the latter details the processor consumption in various status. Additionally, we also consider the potential contention caused by communication from different resources in a communication network, which is evaluated by the trigger time and the overlapped area between two communication paths.

The hardware architecture considered here is a general framework, e.g., MPSoCs or NoCs. We assume that communication does not occupy computation resources, except for setup efforts. The channel for communication may consist of multiple links and nodes. Multiple data transfer may happen at the same time, or cannot overlapped, which is decided by the architecture.

First, we introduce the variables and constants in Table I and Table II, respectively, to present the constraints in allocating and scheduling issues.

A. Constraints

The constraints on static mapping are as follows.

- one task can only be mapped to one processor

$$\sum_{k=1}^{|P|} m_{ik} = 1 \tag{1}$$

- the capacity of a processor cannot be violated

$$\sum_{i=1}^{|T|} m_{ik} \cdot q_i \leq \omega_k \tag{2}$$

- the next-door execution relation is not transitive and reflexive

$$n_{ij}+n_{ji} \leq 1 \wedge (n_{ij} \wedge n_{jl} \rightarrow \neg n_{il}), \;\; n_{ij} \leq m_{ik} \cdot m_{jk'} \tag{3}$$

- the reasoning of o_{ij}

$$d_{ij} \wedge m_{ik} \wedge m_{jk'} \wedge (k \neq k') \rightarrow o_{ij}, \quad o_{ij} \rightarrow d_{ij} \tag{4}$$

For the precedence of computation and computation, we have the following constraints.

- the start of executing t_i should wait for data being transferred, except for the one without any predecessor

$$\sum_{j=1}^{|T|} d_{ji} > 0 \rightarrow fe_{ji}^u \leq s_i^u \tag{5}$$

- the start of data transfer has to wait for the finish of task execution

$$f_i^u \leq se_{ij}^u \tag{6}$$

- for two tasks scheduled next to each other in the same processor, the start of the latter should wait for the finish of the former, where $u \leq v$

$$n_{ij} \rightarrow f_i^u \leq s_j^v \tag{7}$$

- two tasks executed on the same processor cannot be overlapped

$$m_{ik} \wedge m_{jk} \rightarrow s_j^u \geq f_i^u \vee s_i^u \geq f_j^u \tag{8}$$

- the execution of task in later iteration v should not be earlier than any other tasks executed on the same processor in an earlier iteration u, where $u < v$

$$n_{ij} \rightarrow f_j^u \leq s_i^v \tag{9}$$

- the response time between certain tasks should less than the limitation

$$s_j^v - s_i^u \leq dl_{ij} \tag{10}$$

- two communications from the same source cannot overlap

$$m_{ik} \wedge m_{lk} \rightarrow se_{ij}^u \geq fe_{lr}^u \vee se_{lr}^u \geq fe_{ij}^u \tag{11}$$

Next we present quantitative relation between the variables on computation and communication.

- the finish execution of a task considers the occupation of processor for data transfer

$$f_i^u = s_i^u + a_i/(\sum_{k=1}^{|P|} m_{ik} \cdot \rho_k) + \eta \cdot (\bigvee_{j=1}^{|T|} o_{ij}) \quad (12)$$

- the finish of a data transfer considers the distance of the destination

$$fe_{ij}^u = se_{ij}^u + c_i \cdot o_{ij} \cdot (\tau \cdot \delta_{ij} + \tau' \cdot (\delta_{ij} + 1)) \quad (13)$$

where δ_{ij} is the distance from the located processor of t_i to the processor that its successor locates. For an MPSoC, it can be zero or a constant. However, for an NoC, it cannot be ignored.

Discussion. The constraints list above are flexible. For example, Equation 9 can be relaxed such that a task can be executed several times before the next scheduled.

As these constraints are for a general framework, we can adjust it according to specific architectures, especially for communication issues. For example, if the architecture allows multiple communication simultaneously, Equation 11 can be relaxed.

The communication link for different architectures may be quite different. For an MPSoC with buses connected to processors, there is not intermediate node or link between two processors. However, for an NoC, the situation is quite different, where the communication between two processors is fulfilled through links for transmission and switches for router.

We consider 2D-mesh NoC, where the effort for communication can be computed according to the distance between two processors p_i and p_j, where (x_i, y_i) is the location of p_i[1]:

$$D_{ij} = abs(x_i - x_j) + abs(y_i - y_j) \quad (14)$$

B. Optimization objectives

With the defined variables, the makespan of executing an application within N iterations is

$$\mathcal{M} = max\{f_i^N\} \quad (15)$$

However, energy consumption includes cost for computation E_p and communication E_c. For computation, we distinguish different status in a processor and accumulate the total cost. For communication, we consider only the cost for data transfer. There are three modes for a processor: dynamic, idle and sleep. Let Rd_k, Ri_k and Rs_k be the rate of dynamic, static (idle) and sleep power consumption of processor p_k, respectively. The cost for dynamic mode is straightforward.

$$E_{pd} = N \cdot \sum_{k=1}^{|P|} Rd_k \cdot \sum_{i=1}^{|T|} m_{ik} \cdot a_i/\rho_k \quad (16)$$

If the duration of idle is longer than some threshold t_o, a processor will turn to sleep mode, with a time penalty t_Δ for switching into and out of sleep is considered, and the energy penalty ϵ_Δ.

The duration of every iteration may be different, thus the cost for every processor should differ. Let t_f be the first executed task in a processor p_k in every iteration, where the constraint $(\sum_{j=1}^{|T|} n_{jf} = 0) \wedge m_{fk}$ holds. The duration $\mathcal{D}_k^u$ for processor p_k in iteration u is

$$\mathcal{D}_k^u = \begin{cases} s_f^{u+1} - s_f^u & u < N \\ \mathcal{M} - s_f^N & u = N \end{cases} \quad (17)$$

Then the spare time of after executing t_i at p_k in iteration u is

$$\mathcal{S}_{ki}^u = \begin{cases} \mathcal{D}_k^u - f_i^u & m_{ik} \wedge \sum_{j=1}^{|T|} n_{ij} = 0 \\ n_{ij} \cdot (s_j^u - f_i^u) & otherwise \end{cases} \quad (18)$$

According to Equation 18, we can calculate the energy cost for both sleep and idle mode,

$$E_{is} = \sum_{u=1}^{N} \sum_{k=1}^{|P|} \sum_{i=1}^{|T|} \begin{cases} Rs_k \cdot (\mathcal{S}_{ki}^u - t_\Delta) + \epsilon_\Delta & \mathcal{S}_{ki}^u \geq t_o \\ Ri_k \cdot \mathcal{S}_{ki}^u & 0 < \mathcal{S}_{ki}^u < t_o \end{cases} \quad (19)$$

The energy cost for communication is evaluated according to the amount of data and the distance of the transmission.

$$E_c = N \cdot \sum_{k,k'=1}^{|P|} \sum_{i,j=1}^{|T|} o_{ij} \cdot c_{ij} \cdot m_{ik} \cdot m_{jk'} \cdot (\epsilon \cdot D_{kk'} + \epsilon' \cdot (D_{kk'} + 1)) \quad (20)$$

Addition to the energy and makespan optimization, we also expect that the probability of communication contention is reduced. As data transfer occurs in the end of task execution, we consider the potential contention when two tasks at various processors finish execution almost simultaneously (their difference is less than a limited time bound. The probability is evaluated according to the overlapped area between two communication paths in a communication network. Suppose two paths are from processor i_1 to j_1, and i_2 to j_2 respectively. Let oX and oY be the overlapped width and length of the two paths. That is

$$oX = (x_{j1} - x_{i_1}) + (x_{j2} - x_{i_2}) - (max\{x_{j_1}, x_{j_2}\} - min\{x_{i_1}, x_{i_2}\})$$

$$oY = (y_{j1} - y_{i_1}) + (y_{j2} - y_{i_2}) - (max\{y_{j_1}, y_{j_2}\} - min\{y_{i_1}, y_{i_2}\}).$$

If $oX \leq 0$ or $oY \leq 0$, there is no contention. Therefore, we have $oX = max\{oX, 0\}$, and $oY = max\{oY, 0\}$. Let the areas of the two paths be $S_1 = abs((x_{i_1} - x_{j_1}) \cdot (y_{i_1} - y_{j_1}))$ and $S_2 = abs((x_{i_2} - x_{j_2}) \cdot (y_{i_2} - y_{j_2}))$ respectively. The contention probability of two paths is

$$p_c(i_1, j_1, i_2, j_2) = oX \cdot oY/(S_1 + S_2) \quad (21)$$

For any two tasks t_{i_1} and t_{i_2}, the contention probability of data transmission in an NoC is

$$P_{i_1 i_2 =} (abs(f_{i_1}^u - f_{i_2}^v) \leq \xi) \cdot o_{i_1 j_1} \cdot o_{i_2 j_2} \cdot p_c(k_{i_1}, k_{j_1}, k_{i_2}, k_{j_2}) \quad (22)$$

However, data transmission is fulfilled by bus in an MPSoC, we have

$$P_{i_1 i_2 =} (abs(f_{i_1}^u - f_{i_2}^v) \leq \xi) \quad (23)$$

Therefore, the sum is

$$P_c = \sum_{i,j=1}^{|T|} P_{ij} \quad (24)$$

We apply the average contention probability to evaluate the

quality of the allocation strategy, which is denoted by

$$\bar{P_c} = \sum_{i,j=1}^{|T|} abs(p_{ij} - P_c/(|T| \cdot N)) \tag{25}$$

And there are three objectives to be optimized: contention, makespan, total amount of energy cost:

$$minimize(\bar{P_c}) \tag{26}$$

$$minimize(\mathcal{M}) \tag{27}$$

$$minimize(E_{pd} + E_{is} + E_c) \tag{28}$$

Discussion. For an NoC, workload balance is not the main concern, for the number of available processors may be greater than that of tasks.

REFERENCES

[1] J. Huang, C. Buckl, A. Raabe, and A. Knoll. Energy-aware task allocation for network-on-chip based heterogeneous multiprocessor systems. In *Parallel, Distributed and Network-Based Processing (PDP), 2011 19th Euromicro International Conference on*, pages 447–454. IEEE, 2011.