Microprocessors and Microsystems 000 (2016) 1–11

Contents lists available at ScienceDirect

Microprocessors and Microsystems

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

Energy-Aware on-chip virtual machine placement for cloud-supported cyber-physical systems

Xuanzhang Liu[a,c], Huaxi Gu[a,*], Haibo Zhang[b], Feiyang Liu[b], Yawen Chen[b], Xiaoshan Yu[a]

[a] State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an, China
[b] Department of Computer Science, University of Otago, Dunedin, New Zealand
[c] Science and Technology on Information Transmission and Dissemination in Communication Networks Laboratory, Shijiazhuang 050081, China

ARTICLE INFO

Article history:
Received 6 February 2016
Revised 15 June 2016
Accepted 15 July 2016
Available online xxx

Keywords:
Cyber-physical systems
Virtual machine placement
Network-on-Chip
Ant colony optimization

ABSTRACT

Recent trends in the design of cyber-physical systems (CPS) are moving towards heterogeneous multi-core architectures with cloud support. In this paper, we propose an energy-aware scheme for virtual machine placement in cloud-supported CPS with Network-on-Chip (NoC) architecture. We formulate the energy-aware on-chip virtual machine placement problem as an optimization problem, and design a heuristic scheme based on ant-colony optimization. We address problems of slow convergence speed and easily falling into stagnation in ant-colony algorithm by employing pheromone diffusion model that makes the proposed scheme more efficient. Simulation results show that our scheme achieves much higher energy efficiency compared with previous schemes with different network sizes and traffic models.

1. Introduction

Cyber-Physical Systems (CPS) are composed of services and applications deployed across a range of communication topologies, computing platforms, and sensing and actuation devices. Since numerous applications can be deployed in CPS platforms, cloud computing has become a promising computing paradigm which can provide a good support for CPS platforms in terms of cost-efficiency, scalability, and safety [1]. In cloud-supported CPS, the sensors and actuators communicate with the cloud through the cybernet, whereas information processing, control decision making, and system virtualization are all done at the cloud using the high-performance multi-core servers. This ensures quick responses to physical dynamics, and effective and stable control of the physical environment. In addition, applications in the CPS are inherently heterogeneous, real time, reactive and networked with hard deadlines [2]. Every virtual machine in an application is interdependent and has its own execution, arrival and time periods [3]. A key challenge for cloud-based CPS is to optimally execute different applications in the cloud. Server virtualization is an effective method in cloud computing that allows various applications being deployed more flexibly and feasibly in different locations around the world. Through virtualization technology, the physical machine (server or computer) can be logically divided into multiple virtual execution environments, with each acting as a full-function system. This isolated execution environment is so called virtual machine (VM). Therefore, one single server can simultaneously run multiple applications on separated virtual machines that leverage physical resources of the server [4,5].

With the increase number of cores integrated in a single server, the interconnection among these cores can significantly affect the system performance. The ever-advancing integration technology with billions of transistors integrated on a single chip enables on-chip multiple microprocessors based commercial servers, e.g., 80 cores in Intel Teraflops, 188 cores in Cisco/IBM SPP. Moreover, recent developments in Network-on-Chip (NoC), a technology of on-chip interconnection network, provide some efficient communication schemes for these multi-core servers. Compared with traditional bus architecture, NoC leverages the principle of interconnection network and packet switching to achieve low latency, high performance, and low power consumption [6]. For a high-performance server based on the multi-core processor, each virtual machine can be implemented on one single processor core, and the NoC architecture provides efficient communications for these virtual machines, e.g., data flows, cache coherence protocols. At present, the inter-core communication power consumption has already taken an important part of the total power budget owing to the long distance global on-chip communication and ultra-high bandwidth requirement (tens to hundreds of terabits per second) [7]. The cost of managing the power consumption and the associated cooling devices drives the need to design some application-level energy-efficient schemes and algorithms. Careless

* Corresponding author.
E-mail address: hxgu@xidian.edu.cn (H. Gu).

http://dx.doi.org/10.1016/j.micpro.2016.07.013
0141-9331/

Please cite this article as: X. Liu et al., Energy-Aware on-chip virtual machine placement for cloud-supported cyber-physical systems, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.07.013

on chip virtual machine placement, which leads to unreasonable traffic distribution and hotspots, may cause high communication energy consumption and deteriorate the communication performance. For example, placing two virtual machines with large volume communication requirement too far will lead to high power consumption and long communication delay. In this paper, we propose an energy-efficient on chip virtual machine placement algorithm in one single server to solve this challenging problem.

The objective of this work is to design an energy-aware on chip virtual machine placement scheme that is capable of implementing multiple virtual machines on a multi-core server with high power efficiency and desirable performance. The main contributions of this paper include:

- We propose an energy-aware on chip virtual machine placement scheme for cloud computing which sustains the strength of ant colony algorithm in accuracy and efficiency but offsets its weakness in slow convergence speed. By placing virtual machines running the same application to closer cores on the multi-core system according to their traffic rates, the energy consumption and communication delay can be reduced significantly.
- We formulate the on chip virtual machine placement problem as a binary integer programming (BIP) problem that aims to minimize the power consumption for inter-virtual-machine communications. To address this problem, we propose an improved ant colony algorithm that employs the pheromone diffusion model to solve these intrinsic problems of slow convergence and search stagnation. We customize the ant colony algorithm according to the application property and the character of Network on Chip communication architecture to achieve global energy efficiency.
- We carry out extensive simulations to evaluate the energy consumption of our proposed algorithm with both synthetic traffics and realistic data traces. Simulation results show that our algorithm can achieve better energy efficiency compared with some existing algorithms.

The remainder of the paper is organized as follows. Section 2 reviews some related works. In Section 3, we give a motivating example for the on chip virtual machine placement problem. Then the statement of the problem is described in Section 4. We present the energy efficient virtual machine placement algorithm in Section 5. Simulation results are shown in Sections 6, and Section 7 concludes this paper.

2. Related works

2.1. Virtual machine placement

Virtual machine placement has significant influence on cloud computing systems. For example, if two virtual machines with application dependency are non-optimally placed on two servers that locate on different racks or even different cities, the communication delay and energy consumption may be unacceptable. Generally, previous researches have focused on dynamic inter-server virtual machine migration and static server-level virtual machine placement [4,8]. Using these schemes, all the active virtual machines are migrated or placed onto a small number of servers with respect to the performance requirements and resource constraints, whereas the unused servers which have no active virtual machine can be shut down to save power. However, with the rapid development of high performance multi-core servers, tens of hundreds of virtual machines can be implemented on a single server. The on chip virtual machine placement which deals with the intra-server communications between the virtual machines has become an open issue that needs to be well addressed. Grot et al. [9] proposed the Kilo-NoC architecture which guarantees the service requirements of data flows by placing the virtual machines on a shared CMP (chip-level multiprocessor). Wang et al. [13] proposed a virtual machine placement algorithm for the heterogeneous multi-core system that exploits the different properties of each core to optimize the overall system performance and energy efficiency. Hu et al. [10] presented a virtual machine scheduling model to solve the I/O performance bottleneck based on the multi-core dynamic partitioning. Especially, for multicore platforms using CPS, Kanduri et al. explore the impact of application mapping on network contention and predictability [11]. All the current researches of on chip virtual machine placement for multi-core systems mainly target on some specific architectures and applications. It still lacks some general mathematical formulations and optimal solutions for the on-chip virtual machine placement problem, which is extremely important for designing a high performance and scalable multi-core system for cloud-based CPS.

2.2. Placement algorithms

Virtual machine placement can be formulated as optimization problems with objectives to minimize communication delay or maximize throughput or energy efficiency. Some current existing inter-server virtual machine placement algorithms in data centers use Linear Programming model [12,14], Bin Packing algorithm [15] and Artificial Intelligent algorithm [4,16,17]. The basic ideas of these algorithms are described in the followings.

The Linear Programming (LP) schemes assume that the performance goal is linearly related to the placement of virtual machines. For example, Chaisiri et al. [14] proposed an algorithm which places virtual machines on different physical servers with the assumption that the minimal number of servers required and the resources in each server subject to a linear function. In [12], the authors also designed some extension constraints for the linear programming model, such as restricting the number of virtual machines in a single physical server, limiting the number of virtual machine migrations, etc. The main advantage of the Linear Programming based schemes is its simplicity.

The Bin Packing schemes assume that the virtual machine placement can be formulated as a variant of the vector bin-packing problem and various heuristic solutions have been used to solve this problem. Zhang et al. [15] designed several heterogeneity-aware heuristic algorithms for virtual machines placement, which explores the heterogeneity of the requirements of virtual machines for different resources and utilizes the dominant resources (e.g. CPU, memory) as constraints to assist inter-server virtual machine placement. However, this kind of schemes has higher complexity because of the heuristic algorithms.

Artificial Intelligent algorithm is derived from some natural activities and can be used to achieve an optimal virtual machine placement. Xu and Fortes [16] proposed a two-level control system that adopts a modified genetic algorithm with fuzzy multi-objective evaluation to deal with the problem of allocating workloads to virtual machines and virtual machines to physical servers. Ant colony algorithm has been applied to the field of multi-objective optimization for virtual machine placement. Gao et al. [4] formulated the problem of virtual machine placement as a multi-objective combinatorial optimization problem aiming to simultaneously optimize total resource wastage and power consumption. A modified version of the ant colony system algorithm is proposed to effectively deal with the potential large solution space for large-scale data centers. Similarly, Liu et al. [17] proposed an approach based on the ant colony optimization to solve the virtual machine placement problem so as to effectively use the physical resources and to reduce the number of running physical servers.

Please cite this article as: X. Liu et al., Energy-Aware on-chip virtual machine placement for cloud-supported cyber-physical systems, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.07.013

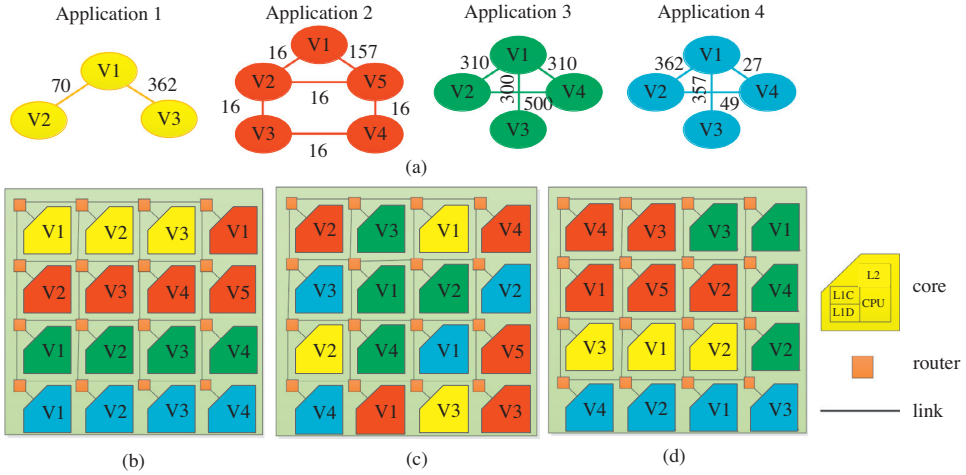

Fig. 1. An example of virtual machine placement on a multi-core server with 16 processor cores. (a) Four applications and traffic between virtual machines; (b) Order Placement; (c) FFD Placement; (d) Ant colony Placement.

These schemes are accurate and efficient for inter-server virtual machine placement. However, as the application properties of intra-server communications between virtual machines are quite different (e.g. cache coherence protocols), and limited by chip hardware resources, they cannot simply be applied to on chip virtual machine placement. The main objective of this paper is to design an efficient virtual machine placement algorithm in terms of high energy efficiency.

3. An motivating example

As shown in Fig. 1, we have a multi-core server with 16 processer cores using a mesh architecture. Suppose each core can only host one virtual machine. Therefore, up to 16 virtual machines can be placed on this multi-core server. We assume that the system is currently hosting four parallel applications with each having several different tasks running on the virtual machines. The applications are labeled from Application 1 to Application 4 using different colors and the virtual machines are labeled as V_i in each application. It is assumed that there is no communication between different applications and the traffic demand between two virtual machines (MB/s) is shown in Fig. 1(a). We can place one virtual machine on a processer core in any position of the chip. However, different placement methods may lead to different energy consumption for data communication. We assume that the energy consumption of transmitting 1-bit data through each hop of router and link is k (pJ). In Fig. 1(b), the virtual machines are placed obliviously in order. In Fig. 1(c), the virtual machines are placed by using a classical bin-packing algorithm that is so-called the First Fit Decreasing (FFD) scheme [15]. In Fig. 1(d), the virtual machines are placed using our scheme which is based on ant-colony optimization. The energy consumptions of three placements are 210.78k (mW), 249.51k (mW) 148.55k (mW), respectively. It can be seen that our scheme produces the lowest energy consumption compared with random scheme and FFD scheme. That is because our scheme is able to search the solution space more efficiently and globally.

4. System model and problem formulation

In this section, we first describe the models, and then give a detailed formulation of the virtual machine placement problem.

4.1. Traffic model

The traffic model defines the communication requirement between any two virtual machines in the system. It is the input of the virtual machine placement problem. We define the traffic matrix W. The traffic demand between two virtual machines V_i and V_j is $W(V_i, V_j)$, which is the average communication volume between two virtual machines in a time period. There are several ways to setup the traffic matrix. The most ordinary method is to use some theoretic models to approximate the traffic pattern, such Poisson distribution [18], Normal distribution [19], Self-similar distribution [20], etc. We adopt this approach in simulations with synthetic traffic. Profiling is another approach to estimate the traffic among cores. In Section 6, we use a trace-based network traffic model to test our algorithm.

4.2. Energy model

Energy consumption is directly related to the traffic in the system. For example, a packet transmit through a router may consume some energy for routing, buffering, and switching, etc. In our energy model, it includes the energy consumption of transferring data between routers and the energy consumption consumed by the transmission links between the routers and the routers. According to [21], the energy consumption of transmitting 1-bit data is calculated as follows:

$$E_{bit} = E_{rbit} + E_{lbit} \tag{1}$$

where E_{rbit} and E_{lbit} are the energy consumption of transferring data through one hop of router and the inter-router link, respectively.

4.3. Problem statement

We assume the physical server is a multi-core system with N processor cores connected using a NoC communication architecture. In the multi-core system, each processor core is composed of a micro-CPU and its dedicated L1/L2 cache, which has full capacity for a virtual machine. We assume that the cores are connected with a 2D mesh topology because of its simplicity in terms of floorplan and scalability.

The set of virtual machines to be placed is represented by $V = \{V_1, V_2, V_3, \ldots V_m\}$ and the set of processer cores is denoted by

Please cite this article as: X. Liu et al., Energy-Aware on-chip virtual machine placement for cloud-supported cyber-physical systems, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.07.013

$C = \{C_1, C_2, C_3, \ldots C_n\}$. Assume that a virtual machine network is a dependency graph denoted by $G(V, W)$, where V is the set of virtual machines. Both V_i and V_j are dependent with each other if any communication takes place between them. So we define a binary decision variable X_{ik} for placing virtual machine V_i on the processer core C_k , $C_k \in C$, as follows:

$$X_{ik} = \begin{cases} 1 & if\ V_i\ is\ assigned\ to\ C_k \\ 0 & otherwise \end{cases} \tag{2}$$

We assume that each virtual machine can be only placed on one processor core and each processor core can only run one virtual machine each time. Therefore, the additional constraint condition must be satisfied:

$$\sum_{k}^{|C|} X_{ik} = 1, \forall i, V_i \in V \tag{3}$$

$$\sum_{i}^{|V|} X_{ik} \le 1, \forall k, C_k \in C \tag{4}$$

Eq. 3 guarantees that each virtual machine can be located on at most one processer core. Eq. 4 means each core can host at most one virtual machine. Based on these definitions, the energy consumed by transmitting 1-bit data between virtual machine V_i and V_j placed on C_k and C_l is:

$$E_{bit}^{k,l} = (Distance(C_k, C_l) + 1) \times E_{rbit} + Distance(C_k, C_l) \times E_{lbit} \tag{5}$$

where $Distance(C_k, C_l)$ is the number of hops between two processer cores C_k and C_l. Therefore, the virtual machine placement problem can be formulated as the following optimization problem:

$$\begin{aligned} \min \quad & E = \sum_{i=1,j=1}^{i=m,j=m} \sum_{k=1,l=1}^{k=n,l=n} E_{bit}^{k,l} \times W(V_i, V_j) \times X_{ik} \times X_{jl} \\ Subject\ to: \quad & \sum_{k}^{|C|} X_{ik} = 1, \forall i, V_i \in V \\ & \sum_{i}^{|V|} X_{ik} \le 1, \forall k, C_k \in C \end{aligned} \tag{6}$$

Since the number of processer cores is $|C|$, the number of placement schemes will be the factorial $|C|!$ when $|V| = |C|$, which is an NP-hard problem, we propose an intelligent optimization algorithm to solve this problem according to the above optimization model.

5. Heuristic solution based on ant colony algorithm

Ant colony algorithm is one of the most efficient meta-heuristic algorithms, which is inspired by the observation of real ant colonies based on their collective foraging behavior [22]. Ants are social insects and live in colonies. Their behaviors are controlled by the goal of finding food. First, ants search for food surrounding their nest in a random manner. A special substance called pheromone that is used to exchange path information among individual ants is laid when ants are moving. Once an ant finds a food source, it will carry the food and leave certain quantity of pheromones on the ground during the return trip. The pheromone trails will guide other ants to the food source. Ants will return faster on the shortest path to the food. Thereby, this path will have a stronger pheromone concentration, thus being more attractive for subsequent ants to follow it. Through this positive feedback mechanism, the probability for these ants choosing the shortest path could be much higher.

Pheromone trails start to evaporate after a certain period of time. And like the chemical leakage in the air, the pheromone diffusion [23] approximately subjects to Gaussian plume model [24]. However, in traditional ant colony algorithm, the pheromone exchange among ants is insufficient and not in time. It causes slow convergence speed and easily falls into stagnation of the solution [23]. In this section, we propose an improved ant colony algorithm based on pheromone diffusion to solve the placement problem. The pseudo code of the proposed ant colony algorithm is given in Algorithm 1. This algorithm works as follows. In the initialization phase, the parameters are initialized and all the pheromone trails are set to τ_0. In each iteration we use the roulette wheel rule to choose a particular processer core as the next one to hold current virtual machine. This rule is based on the information of the current pheromone concentration on processer cores and the heuristic factor which guides the ants towards choosing the most promising processer cores. After all ants have constructed their solutions, a global update is performed with each solution of the current objective value.

Algorithm 1 VM placement

Input:
Set of virtual machine V and set of processer core C, traffic matrix W

Output:
VM placement solution

```
/*Initialization*/
Set parameters value α, β, ρ, Q
Calculate the heuristic information according to Eq. 8
Initialize all the pheromone values be tween virtual machines to τ0
/*Iteration*/
for each ant=1 to N(number of ants) do
    for each i=1 to |V|(number of virtual machines) do
        for each j=1 to |C|(number of processer cores) do
            Calculate the probability pij according to Eq. 7
        end for
        Generate a random number q
        if q < pij then
            Select the processer core to place the virtual machine
        end if
    end for
end for
Calculate the value of objective for every ant
if S(local_solution)>S(global_solution) then
/*local_solution means the best solution of this iteration*/
/*global_solution means the best solution until this iteration*/
    Update the S(globalsolution)
end if
Update pheromone information τij according to Eq. 9
/*Iteration end*/
Return global best solution
```

5.1. Constructing a solution

Each ant represents a feasible solution of virtual machine placement. When virtual machine V_i is placed on processer core C_j, the state of ant k has been changed. That is so call the move of the ant, and every unavailable move is retained in a set $Tabu_k$. Moreover, the colony of ants will be re-constructed in the same way at each iteration. For each virtual machine V_i, we select C_j by using roulette rule according to the probability p_{ij}. In this work, we define the probability p_{ij} that ant k chooses to assign V_i to C_j as follows:

$$p_{ij} = \frac{(\tau_{ij}(t))^{\alpha} \times (\eta_{ij}(t))^{\beta}}{\sum_{s \notin Tabu_k} (\tau_{sj}(t))^{\alpha} \times (\eta_{sj}(t))^{\beta}} \forall i, V_i \in V \tag{7}$$

Please cite this article as: X. Liu et al., Energy-Aware on-chip virtual machine placement for cloud-supported cyber-physical systems, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.07.013

whereby, τ_{ij} denotes the pheromone concentrated on the processer cores which is defined in Eq. 9 below, and η_{ij} is the heuristic factor defined in Eq. 8. Eq. 7 shows that a candidate is chosen relatively to the transition probability which depends on two factors: a heuristic factor and a pheromone factor. Moreover, two parameters α and β are used in order to respectively determine the relative importance of the pheromone trail and the heuristic information.

5.2. Definition of heuristic information

Different from real ant, the artificial ant can make use of the heuristic information when they search for the optimal solution. This heuristic information η_{ij} indicates the desirability of assigning V_i to C_j. Hence, the appropriate calculation method of the heuristic information may significantly affects the efficiency of the algorithm. Based on characteristics of NoC architecture, in this paper, we calculate heuristic information fixedly with the algorithm running because this method is faster. For each processer core we define the heuristic information as:

$$\eta_{ij} = \frac{1}{d(C_j)}, \forall i, V_i \in V \tag{8}$$

where $d(C_j)$ is the average distance of C_j to any other processer cores. Apparently, $d(C_j)$ has the inverse ratio with η_{ij}. According to Eq. 7, the larger η_{ij} is, the more probability that C_j can be chosen. Hence, the processer core with lower average distance is more likely to be selected. Thus, with the iteration performed, the traffic intensive virtual machines will have higher probability to place on these processer cores.

5.3. Pheromone trail update

Another crucial part of ant colony algorithm is the update of pheromone trails. In the initialization phase, initial pheromone level is calculated by $\tau_0 = 1/|C|$. After all ants have constructed the solutions, pheromone trails on all pairs need to be updated in order to help guiding the algorithm towards the optimal solution. The pheromone trail value could either increase, as ants deposit pheromone, or decrease, due to pheromone evaporation. When updating pheromone trails, one has to decide on which of the constructed solutions to lay pheromones. In traditional ant colony algorithm, there are usually two strategies to update the pheromone trails. The first strategy is that each ant contributes to the trail update by using the global information or the local information. The global information is the total objective function value of this ant in current iteration. Meanwhile, the local information is the partial objective function value of this ant in current iteration. For instance, Dorigo M [25] proposed three classical update model called Ant-Cycle model, Ant-Quantity model and Ant-Density model. The difference amongst three models is that only Ant-Cycle model could use the global information.

The second strategy is only to use the information contained in the iteration-best or the best-so-far solutions to update the pheromone. The iteration-best solution is done after each solution has been constructed, and its purpose is to decay the pheromone intensity of the components of the solution just constructed. Thus, other component choices in the subsequent solutions can also be explored. Best-so-far solution has been finished after all solutions in a colony (an iteration) have been constructed and improved by local search. It aims at reflecting the discoveries of this iteration.

Different from traditional methods, our algorithm adopts a comprehensive method of two strategies. The key reason to achieve this method is that the pheromone can not only evaporate but also diffuse in the real world. And the diffusion model approximately subjects to Gaussian plume model [24]. To truly reflect the real state, our algorithm enhances the pheromone of the processer core whose total energy consumption is relatively small. Moreover, with the pheromone diffusion, the other processer cores that are not selected at this iteration will also obtain pheromone. Early stagnation of the search is most likely to be avoided by introducing this pheromone diffusion model. The pheromone update rule is defined as:

$$\tau_{ij}(t) = \rho \times \tau_{ij}(t-1) + \sum_{k}^{N} \Delta\tau_{ij}^{k} \tag{9}$$

where the constant $\rho \in [0, 1]$ is the parameter that controls the pheromone persistence. $\Delta\tau_{ij}^{k}$ represents the pheromone increment that the ant k leaves. To improve the convergence speed of the algorithm, we introduce pheromone diffusion model to simulate the behavior of ant colony more realistically. Thereby, $\Delta\tau_{ij}^{k}$ is defined as:

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{\sqrt{2\pi}E}, & X_{ij} = 1 \\ \dfrac{Q}{\sqrt{2\pi}E} \exp\left\{-D^2(C_i, C_j)/2E^2\right\}, & \\ \qquad X_{ij} \neq 1 \text{ and } D(C_i, C_j) \leq |C|/2, \forall i \in m & \\ 0, \qquad \text{otherwise} & \end{cases} \tag{10}$$

where Q means the impact factor of the pheromone. It is a constant which can influence the coverage speed in a certain degree. When the $X_{ij} = 1$, which means V_i is placed to C_j, the pheromone increment is the maximum. And then due to the pheromone diffusion, the pheromone concentration in the area of $|C|/2$ radius from the center of C_j will increase by following the Gaussian plume model. Two purposes of this operation have been achieved, 1) to decay the pheromone intensity of the components of the solution just constructed and 2) to encourage exploration of other component choices in the subsequent solutions to be constructed.

6. Performance evaluation

With the increase of the scale of NoC architecture, it is necessary to gain a first insight into the performance of the algorithm on large-scale before implementing it in a real environment. For the limitation of facilities, in this section, we use some simulations to evaluate the proposed algorithm with respect to performance and computing efficiency. The performance of the proposed ant algorithm is compared with that of the random placement algorithm and FFD algorithm. All the algorithms are coded by using C language and run on an Intel Pentium Dual-Core processor with 2.94 GHz CPU and 4 GB RAM. All the settings for various parameters of improved ant colony algorithm have a direct effect on the algorithm performance. Appropriate parameter values are determined on the basis of preliminary experiments [4,17,23]. The parameters of the ant colony algorithm are set to the following values, $\alpha = 1$, $\beta = 2$, $\rho = 0.8$, $Q = 10000$, and 100 ants are used in each iteration. The max iteration is set as 100, 150, 200 respectively with the increase of the scale of NoC. Every scenario is repeated with 100 runs for each instance.

Mesh topology is employed in the simulation which is a widely used in NoC architecture [26,27]. Different communication traffic models are tested in the simulation. Another important question is how to calculate the sum of traffic demands. Although several works on routing algorithms [28,29] have been proposed, we adopt XY-routing in this paper for simplicity. In scenario one and scenario two, we evaluate the performance with different network sizes that include 16 processer cores (4 × 4 mesh), 36 processer cores (6 × 6 mesh), 64 processer cores (8 × 8 mesh) respectively, and with different traffic demands of virtual machines and the interdependencies among them. The traffic demands of virtual machines subject to the normal distribution $N(0.2, 0.1)$, $N(0.4, 0.1)$,

Please cite this article as: X. Liu et al., Energy-Aware on-chip virtual machine placement for cloud-supported cyber-physical systems, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.07.013

Table 1
Execution time of ant colony placement.

Traffic model	Topology	Variation	Execution time(ms)
Global	4 × 4	0.2	18.051
		0.4	18.055
		0.6	18.885
Global	6 × 6	0.2	115.43
		0.4	118.55
		0.6	119.53
Global	8 × 8	0.2	516.82
		0.4	525.53
		0.6	528.02
Partioned	4 × 4	0.2	15.59
		0.4	15.83
		0.6	17.12
Partitioned	6 × 6	0.2	118.05
		0.4	118.20
		0.6	118.56
Partitioned	8 × 8	0.2	564.48
		0.4	565.41
		0.6	566.43

$N(0.6, 0.1)$ with the unit of MB/S [30]. In scenario three, we introduce the trace-based network traffic models which have been collected from execution of applications to show that our algorithm still has better performance for on-chip virtual machine placement [31]. In addition, computation time is also an essential metric to evaluate. Since the similarity of network scales, we only test the computation time in Scenario one which can prove the efficiency of our algorithm.

6.1. Scenario one: performance comparisons with different traffic models

In this scenario, we compare 1) global traffic model in which each virtual machine communicates with each other at a constant rate, 2) partitioned traffic model in which the virtual machines are divided into isolated partitions, and only virtual machines within the same partition will communicate with each other. The energy consumption of transmitting a bit of data through a router and a link is 4.171nJ and 0.449nJ, respectively according to [21]. The objective value we calculated is used to show the differences, the smaller objective value indicates the better performance. Traffic demands of virtual machines meet the normal distribution function with different parameters (mean and variance value) that mentioned above. Since all of the virtual machines have connections with each other in global traffic model, the number of application can be seen as only one. Meanwhile, the number of applications is generated randomly in partitioned traffic model, which the minimum number of applications is set to 2. In each case, the number of virtual machines equals to the number of processer cores. The time simulation results are depicted in Table 1.

Fig. 2 shows the comparison results, where X axis stands for the mean traffic from each virtual machine. For example, 0.2 means that communication traffic between virtual machines obeys the normal distribution with 0.2 as mean value, and 0.1 represents variance value. In the figure, each bar indicates that the objective values calculated by three algorithms. According to Fig. 2(a) (c) (e), under the global traffic model, the objective function values provided by improved ant colony placement are about average 3.2% and 1.9% smaller than those of the other two algorithms. In other words, if a chip is devoted to just one application with homogeneous traffic rates among virtual machines, the objective values obtain few discrepancies among three algorithms. Since all virtual machines transfer traffic to each other, the location is not the main impact factor to objective values. In addition, with the network scale expansion, the improvements are less for the range of feasible solution increasing. That causes the improved ant colony algorithm needs more iterations to search for the approximate optimal solution. Fig. 2(b) (d) (f) compares the performance of three placement algorithms with different network scale under partitioned traffic model. We have three groups for each test. The results indicate the same trends as those under global traffic model, with the performance improvement potential being even more prominent. Ant colony algorithm can provide average 47% and 42% improvement than random placement and FFD placement. This can be attributed to that improved ant colony algorithm has much higher global searching ability.

As it can be observed, computation time is required to search for the placement. The computation complexity is closely related to the scale of NoC, i.e., the number of nodes in the network. As expected, the computation complexity represented by the run time in Table 1 increases as the scale of NoC increases. And the max computation time is less than 1 second. This confirms the efficiency of the proposed algorithm even for large scale NoCs.

6.2. Scenario two: performance comparisons with different number of the utilized cores

In this scenario, we compare the performance of three algorithms with different numbers of the utilized cores [30]. The number of the cores in the higher utilization is followed uniform distribution which the scale is from three quarters number of the cores to the whole number of the cores. And The number of the cores in the lower utilization will use half or one quarter number of the cores randomly. The traffic model is adopted partitioned traffic model. Similar to the scenario one, the objective value is used to show the differences. And The interdependencies among applications are also varying randomly which the minimum number of applications is set to 2.

Fig. 3(a) (c) (e) shows the performance of three algorithms where all the processer cores are higher utilized and Fig. 3(b) (d) (f) displays the performance of three algorithms are lower utilized. We can see that, 1) the performance of improved ant colony placement provides about average 66.27% and 47.53% improvement compared with the other two algorithms. Since our placement algorithm takes into account of the dependencies of applications or communication requiring frequency, it can greatly reduce the total energy consumption for the placements. In addition, the improved ant colony placement produces the lowest energy consumption because it is able to search the solution space more efficiently and globally. Thus, it can find solutions with lower energy consumption compared with random and FFD. 2) The energy consumption of FFD is between those of the other two. The reason is that FFD can find local optimal solutions in iteration which may cause global solution degradation.

6.3. Scenario three: performance comparisons with trace-based traffic model

In this scenario, we compare the performance of three placement algorithms with trace-based network traffic model. The set of on-chip network traffic traces is collected from the PARSEC v2.1 benchmark suite [32]. The PARSEC suite contains multiple input sets for each benchmark, and we collect traces for all the benchmarks that work with simulations up to 64 cores. The iteration in this scenario is 150.

Fig. 4 plots Objective values of three algorithms under different application traffic models on 8 × 8 mesh architecture. It can be seen that the improved ant colony algorithm provides better performance than those of the other two algorithms. However, under traffic models swaptions and bodytrack, the iteration is set to 250 in order to obtain the better performance of our algorithm than

Please cite this article as: X. Liu et al., Energy-Aware on-chip virtual machine placement for cloud-supported cyber-physical systems, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.07.013

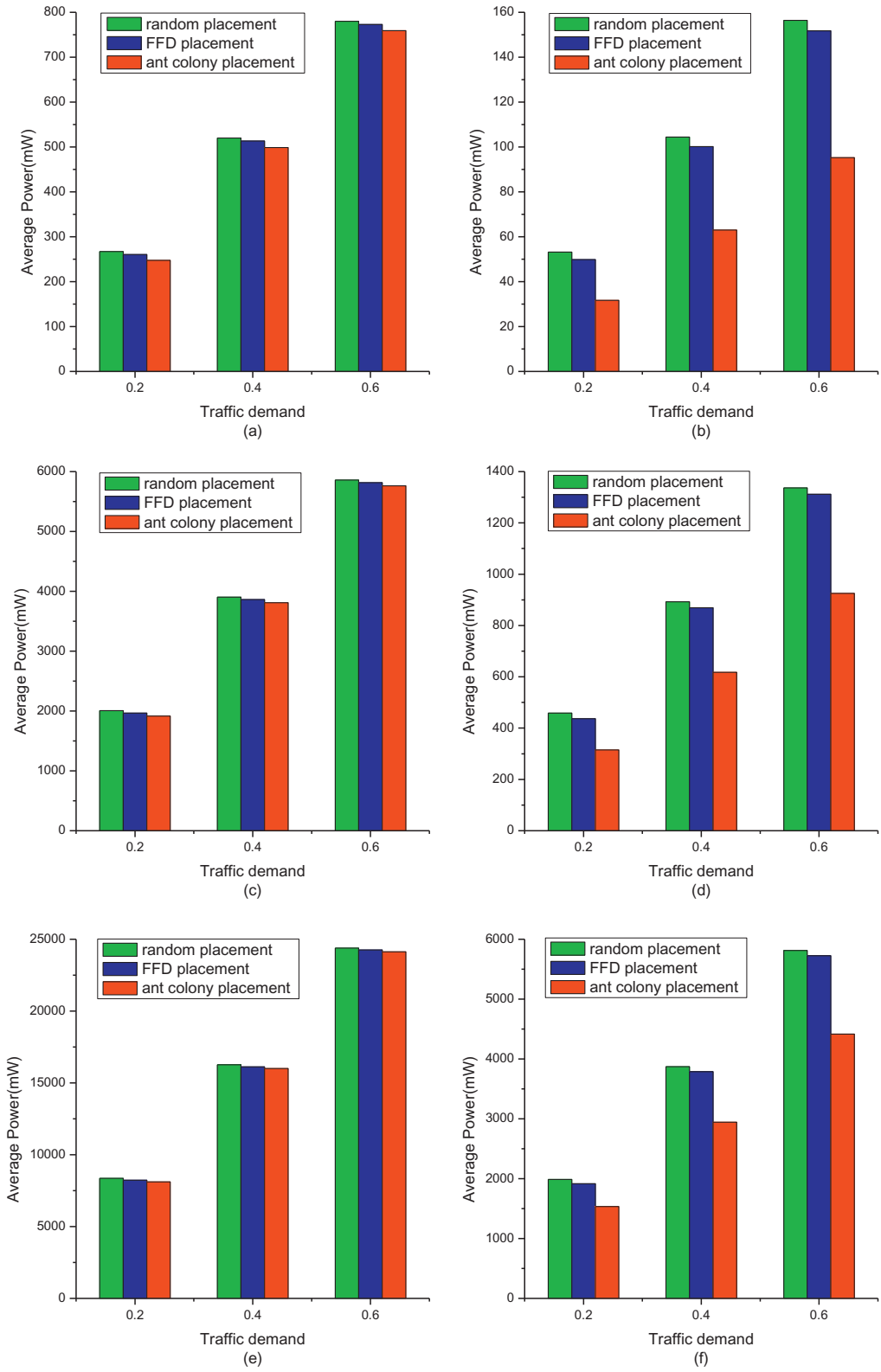

Fig. 2. Objective value of three algorithms (global traffic model and partitioned traffic model with different traffic variance) (a) 4 × 4 mesh, global traffic model; (b) 4 × 4 mesh, partitioned traffic model; (c) 6 × 6 mesh, global traffic model; (d) 6 × 6 mesh, partitioned traffic model; (e) 8 × 8 mesh, global traffic model; (f) 8 × 8 mesh, partitioned traffic model.

Please cite this article as: X. Liu et al., Energy-Aware on-chip virtual machine placement for cloud-supported cyber-physical systems, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.07.013

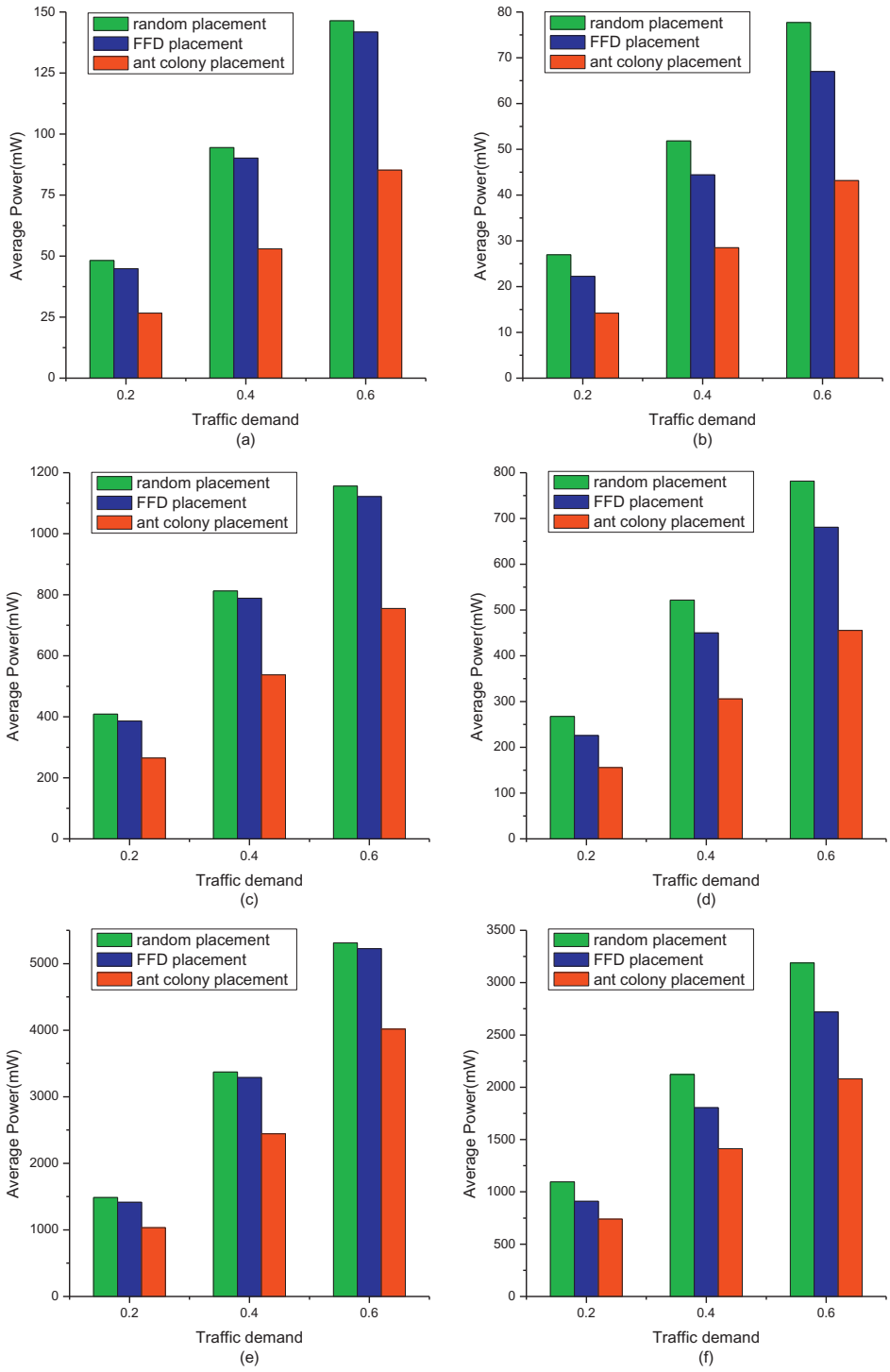

Fig. 3. Objective value of three algorithms (different number of the utilized cores under partitioned traffic model with different traffic variance) (a) 4 × 4 mesh, higher utilization; (b) 4 × 4 mesh, lower utilization; (c) 6 × 6 mesh, higher utilization; (d) 6 × 6 mesh, lower utilization; (e) 8 × 8 mesh, higher utilization; (f) 8 × 8 mesh, lower utilization.

that of FFD algorithm. Analyzing the traffic matrices of these two application traffic models, it can be seen that there are some traffic rates which are much higher than the others in traffic matrices. This means that, by using FFD algorithm, these virtual machines with higher traffic rates will be placed first which can reduce the total traffic on chip so as to the energy consumption. While, for our improved ant colony algorithm, these special points in traffic matrices will cause more iterations to search for the approximate optimal solution.

Please cite this article as: X. Liu et al., Energy-Aware on-chip virtual machine placement for cloud-supported cyber-physical systems, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.07.013

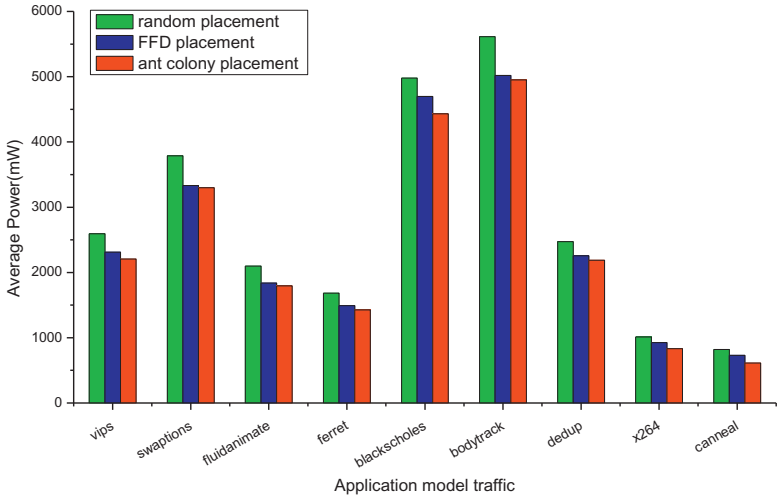

Fig. 4. Objective value of three algorithms under different application traffic models.

7. Conclusion

This paper proposes an energy-aware on chip virtual machine placement with NoC architecture for cloud-supported Cyber-Physical Systems. Our primary aim is to reduce the energy consumption that is generated by traffic communications among different virtual machines. We formulate the virtual machine placement algorithm as an optimization problem and derive the detailed energy model. We design an improved ant colony algorithm based on the characteristic of NoC architecture, which efficiently improves the energy efficiency of virtual machine communication. Compared with traditional random placement and FFD placement schemes, the simulation results show that our algorithm performs better with different traffic models and network sizes. For the future work, we are planning to consider the multi-cores virtual machines placement on chip. And it is more realistic and also very important to improve the algorithm after evaluating the system performance.

Acknowledgment

This work was supported by the National Science Foundation of China Grant No. 61472300, the Fundamental Research Funds for the Central Universities Grant No. JB150318, the 111 Project Grant No. B08038, and the Research Funds of the Science and Technology on Information Transmission and Dissemination in Communication Networks Laboratory.

References

[1] J. Wan, M. Chen, F. Xia, D. Li, K. Zhou, From machine-to-machine communications towards cyber-physical systems, Comput. Sci. Inform. Syst. 10 (2013) 1105C1128.
[2] L. Ordinez, O. Alimenti, E. Rinland, M. Gmez, J. Marchetti, Modeling and specifying requirements for cyber-physical systems, in: Proceedings of IEEE Latin America Transactions, 2013, pp. 625–632.
[3] A. Masrur, M. Kit, V. Matna, T. Bures, W. Hardt, Component-based design of cyber-physical applications with safety-critical requirements, Microprocess. Microsy. 42 (2016) 70–86.
[4] Y. Gao, H. Guan, Z. Qi, Y. Hou, L. Liu, A multi-objective ant colony system algorithm for virtual machine placement in cloud computing, J. Comput. Syst. Sci. 79 (2013) 1230–1242.
[5] K. Zheng, X. Wang, L. Li, X. Wang, Joint power optimization of data center network and servers with correlation analysis, in: Proceeding of the IEEE International Conference on Computer Communications (INFOCOM), IEEE, 2014, pp. 2598–2606.
[6] J.M. Ye, M. Cao, Z. Qu, T. Chen, Regional cache organization for noc based many-core processors, J. Comput. Syst. Sci. 79 (2013) 175–186.
[7] C. Batten, A. Joshi, V. Stojanovc, K. Asanovic, Designing chip-level nanophotonic interconnection networks, Integrated Optical Interconnect Architectures for Embedded Systems, Springer, 2013, pp. 81–135.
[8] Y. Feng, B. Li, B. Li, Bargaining towards maximized resource utilization in video streaming datacenters, in: Proceeding of the IEEE International Conference on Computer Communications (INFOCOM), IEEE, 2012, pp. 1134–1142.
[9] B. Grot, J. Hestness, S.W. Keckler, O. Mutlu, Kilo-NOC: a heterogeneous network-on-chip architecture for scalability and service guarantees, ACM SIGARCH Comput. Archit. News 39 (2011) 401–412.
[10] Y. Hu, X. Long, J. Zhang, J. He, L. Xia, I/O scheduling model of virtual machine based on multi-core dynamic partitioning, in: Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, ACM, 2010, pp. 142–154.
[11] A. Kanduri, A.M. Rahmani, P. Liljeberg, H. Tenhunen, Predictable application mapping for manycore real-time and cyber-physical systems, in: Proceedings of the 9th IEEE International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC), IEEE, 2015, pp. 135–142.
[12] B. Speitkamp, M. Bichler, A mathematical programming approach for server consolidation problems in virtualized data centers, IEEE Trans. Serv. Comput. 3 (2010) 266–278.
[13] Y. Wang, X. Wang, Y. Chen, Energy-efficient virtual machine scheduling in performance-asymmetric multi-core architectures, in: Proceedings of the 8th International Conference on Network and Service Management, International Federation for Information Processing, 2012, pp. 288–294.
[14] S. Chaisiri, B.-S. Lee, D. Niyato, Optimal virtual machine placement across multiple cloud providers, in: Proceedings of the IEEE Asia-Pacific Services Computing Conference, IEEE, 2009, pp. 103–110.
[15] Y. Zhang, N. Ansari, Heterogeneity aware dominant resource assistant heuristics for virtual machine consolidation, in: Proceedings of the IEEE Global Communications Conference (GLOBECOM), IEEE, 2013, pp. 1297–1302.
[16] J. Xu, J.A. Fortes, Multi-objective virtual machine placement in virtualized data center environments, in: Proceedings of the IEEE/ACM International Conference on Green Computing and Communications & IEEE/ACM International Conference on Cyber, Physical and Social Computing, IEEE, 2010, pp. 179–188.
[17] X. Liu, Z. Zhan, K. Du, W. Chen, Energy aware virtual machine placement scheduling in cloud computing based on ant colony optimization approach, in: Proceedings of the 2014 conference on Genetic and evolutionary computation, ACM, 2014, pp. 41–48.
[18] M. Arjomand, H. Sarbazi-Azad, Power-performance analysis of networks-on-chip with arbitrary buffer allocation schemes, IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. 29 (2010) 1558–1571.
[19] A.M. Rahmani, P. Liljeberg, K. Latif, J. Plosila, K.R. Vaddina, H. Tenhunen, Congestion aware, fault tolerant, and thermally efficient inter-layer communication scheme for hybrid noc-bus 3D architectures, in: Proceedings of 5th IEEE/ACM International Symposium on Networks on Chip (NoCS), IEEE, 2011, pp. 65–72.
[20] M. Geyong, M. Ould-Khaoua, A performance model for wormhole-switched interconnection networks under self-similar traffic, IEEE Trans. Comput. 53 (2004) 601–613.
[21] C. Wu, C. Deng, L. Liu, J. Han, J. Chen, S. Yin, S. Wei, An efficient application mapping approach for the co-optimization of reliability, energy, and performance in reconfigurable noc architectures, IEEE Trans. Comput. Aid. D. 34 (2015) 1264–1277.
[22] C. Lin, G. Wu, F. Xia, M. Li, L. Yao, Z. Pei, Energy efficient ant colony algorithms for data aggregation in wireless sensor networks, J. Comput. Syst. Sci. 78 (2012) 1686–1702.

Please cite this article as: X. Liu et al., Energy-Aware on-chip virtual machine placement for cloud-supported cyber-physical systems, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.07.013

[23] G. Huang, X. Cao, X. Wang, An ant colony optimization algorithm based on pheromone diffusion, ACTA ELECTRONICA SINICA. 32 (2004) 865–868.
[24] Y. Fares, P.J. Sharpe, C.E. Magnuson, Pheromone dispersion in forests, J. Theor. Biol. 84 (1980) 335–359.
[25] M. Dorigo, L.M. Gambardella, M. Middendorf, T. Stutzle, Guest editorial: special section on ant colony optimization, IEEE Trans. Evolut. Comput. 6 (2002) 317–319.
[26] D. Xu, J. Fan, X. Jia, S. Zhang, X. Wang, Hamiltonian properties of honeycomb meshes, Inform. Sciences 240 (2013) 184–190.
[27] X. Wang, J. Fan, X. Jia, S. Zhang, J. Yu, Embedding meshes into twisted-cubes, Inform. Sciences 181 (2011) 3085–3099.
[28] D. Xiang, K. Chakrabarty, H. Fujiwara, Multicast-based testing and thermal-aware test scheduling for 3D ICs with a stacked network-on-chip, IEEE Trans. Comput. (2015).
[29] Z. Chen, K. Chakrabarty, D. Xiang, MVP: Minimum-violations partitioning for reducing capture power in at-speed delay-fault testing, IEEE Trans. on Computer-Aided Design 30 (2011) 1762–1767.
[30] X. Meng, V. Pappas, L. Zhang, Improving the scalability of data center networks with traffic-aware virtual machine placement, in: Proceeding of the IEEE International Conference on Computer Communications (INFOCOM), IEEE, 2010, pp. 1–9.
[31] J. Hestness, B. Grot, S.W. Keckler, Netrace: dependency-driven trace-based network-on-chip simulation, in: Proceedings of the 3rd International Workshop on Network on Chip Architectures, ACM, 2010, pp. 31–36.
[32] M. Gebhart, J. Hestness, E. Fatehi, P. Gratz, S.W. Keckler, Running PARSEC 2.1 on M5. university of texas at austin, department of computer science,, Technical Report# TR-09-32, 2009.

Please cite this article as: X. Liu et al., Energy-Aware on-chip virtual machine placement for cloud-supported cyber-physical systems, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.07.013

Xuanzhang Liu is a postgraduate student in School of Telecommunications Engineering, Xidian University, China. In 2012, He received his B.S. from Yunnan University, China. The main research interests include: virtual machine placement in Data Center network and virtual machine placement optimization aimed at NoC-based platform.

Gu Huaxi received the B.E. degree, M.E. and PhD in Telecommunication Engineering and Telecommunication and Information Systems from Xidian University, Xidian in 2000, 2003 and 2005 respectively. He is full Professor in the State key lab of ISN, Telecommunication Department, Xidian University, Xidian, China. His current interests include interconnection networks, networks on chip and optical intrachip communication. He has more than 100 publications in refereed journals and conferences. He has been working as a reviewer of IEEE Transaction on Computer, IEEE Transactions on Dependable and Secure Computing, IEEE system Journal, IEEE Communication letters, Information Sciences, Journal of Supercomputing, Journal of System Architecture, Journal of Parallel and Distributed Computing, Microprocessors and Microsystems etc.

Haibo Zhang received the MSc degree in Computer Science from Shandong Normal University, China in 2005, and the PhD degree in Computer Science from the University of Adelaide, Australia in 2009. From 2009 to 2010, he was a postdoctoral research associate at Automatic Control Laboratory, KTH, Sweden. Currently he is a lecturer at Computer Science department of University of Otago, New Zealand. His research interests include real-time industrial wireless communications, wireless sensor/ad hoc networks, delay-tolerant networks, green computing, distributed algorithms and protocol design.

Feiyang Liu is a PhD candidate from Department of Computer Science, University of Otago, New Zealand. He obtained B.S. and M.S. degrees from Xidian University, China in 2009 and 2012, respectively. His research interests include Network on Chip (NoC), Optical Network on Chip (ONoC), Wireless Sensor Network (WSN), etc.

Yawen Chen obtained her PhD degree in Computer Science from The University of Adelaide in Australia in 2008. Before her PhD study, she received Bachelor degree in Computer Science in 2002 and Master degree in 2004 from Shandong Normal University in China, and then worked as a researcher in Japan Advanced Institute of Science and Technology (JAIST) in 2005. After her PhD study, she worked as postdoctoral researcher at Royal Institute of Technology (KTH) in 2009. She has been a Lecturer in the University of Otago in New Zealand since 2011. Her research interests include resource optimization and performance evaluation in computer networking and computer architecture (optical networks, interconnection networks, green computing, and etc).

Xiaoshang Yu received the M.E. degree in Electronics and Communications Engineering from Xidian University in 2013. Now he is doing the Ph.D. Programme in Telecommunication and information system in the State key lab of ISN, Xidian University. His main research interests are related to optical interconnected networks, data center networks.

Please cite this article as: X. Liu et al., Energy-Aware on-chip virtual machine placement for cloud-supported cyber-physical systems, Microprocessors and Microsystems (2016), http://dx.doi.org/10.1016/j.micpro.2016.07.013