Quick Answer for Big Data in Sharing Economy: Innovative Computer Architecture Design Facilitating Optimal Service-Demand Matching

Lei Guo, *Member, IEEE*, Zhaolong Ning, *Member, IEEE*, Weigang Hou, *Member, IEEE*, Bin Hu, *Senior Member, IEEE*, and Pengxing Guo

***Abstract*— In sharing economy, people offer idle social resources to others in a sharing manner. Through community-based online platforms, the people offering services can earn commission while others can enjoy a better life via renting social resources. Consequently, the value-in-use of services is expectedly strengthened within the unit time, although the total amount of social resources remains constant. Influenced by sharing economy, some famous companies have developed intelligent systems to analyze the most appropriate coincidence between citizens' idle supply and renting demand from numerous data sets. However, the big data analysis of the optimal service-demand matching usually runs on the traditional multiprocessors equipped in intelligent systems, so-called "system-on-chip." In this paper, we design a novel computer architecture—the accelerator based on optical network-on-chip (ONoC)—to further speed up the matching between citizens' offer and demand in sharing economy. Our ONoC-based accelerator is able to quickly calculate the optimal service-demand matching by processing computation tasks on parallel cores, i.e., task-core mapping. In addition, to improve the accelerator reliability, the assorted task-core mapping algorithm is also designed. The extensive simulation results based on real trace file demonstrate the effectiveness of our system and algorithm.**

***Note to Practitioners*—Sharing economy is of great importance for realizing green consumption and sustainable development in our human society. Sharing economy enterprise calls for intelligent system design for service-demand matching in the current big data era. In this paper, we design the accelerator based on ONoC to further speed up the matching between citizens' offer and demand in sharing economy. By processing computation tasks on parallel cores using our algorithm, the task-core mapping can be performed with high speed and reliability. The simulation results—based on the trace file of Amazon Mechanical Turk—can well guide the practitioners to design a more clever and reliable product by quickly calculating the optimal service-demand matching.**

***Index Terms*— Computer architecture design, optical network-on-chip (ONoC), service-demand matching, sharing economy.**

Manuscript received October 27, 2017; revised April 15, 2018; accepted May 8, 2018. Date of publication June 5, 2018; date of current version October 4, 2018. This paper was recommended for publication by Associate Editor X. Li and Editor M. P. Fanti upon evaluation of the reviewers' comments. This work was supported in part by the General Armament Department and Ministry of Education United Fund under Grant 6141A0224-003, in part by the National Nature Science Foundation of China under Grant 61502075, Grant 61471109, Grant 61501104, Grant 61775033, and Grant 61771120, and in part by the Fundamental Research Funds for the Central Universities under Grant N150406001, Grant N150401002, Grant N161604004, Grant N161608001, and Grant DUT17RC(4)49. *(Corresponding authors: Zhaolong Ning; Weigang Hou; Bin Hu.)*

L. Guo, W. Hou, and P. Guo are with the Smart Systems Lab, School of Computer Science and Engineering, Northeastern University, Shenyang 110819, China (e-mail: houweigang@cse.neu.edu.cn).

Z. Ning is with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116023, China (e-mail: zhaolongning@dlut.edu.cn).

B. Hu is with the School of Information Science and Engineering, Lanzhou University, Lanzhou 730000, China (e-mail: bh@lzu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2018.2838340

I. Introduction

SHARING economy refers to peer-based activities of sharing the access to goods and services, and it is coordinated through community-based online services. Sharing economy is also known as collaborative consumption that people sharing of resources rather than having individual ownership [1]–[3]. By leveraging idle resources to produce more goods and services, sharing economy significantly drives green consumption and sustainable development in our human society [4]–[6]. For example, someone earns commission through renting personal idle house/car/surplus labor to others, and someone gets the short-term access to social resources at a lower price instead of buying it [7], [8]. Thus, the social resources' use-in-value per unit time expectedly rises once the total amount remains constant. Currently, sharing economy has potentially resulted in great impact on citizens' everyday life and generated huge economic benefits, e.g., Amazon Mechanical Turk [9].

As an emerging economic-technological paradigm, sharing economy is driven by developments in information technologies, such as simplify sharing of social resources through intelligent systems. The Amazon Mechanical Turk model is a classic sharing economy confronted with the challenging issue of the optimal service-demand matching achieved by intelligent systems [10]–[12]. In Fig. 1, based on the Amazon Mechanical Turk model, the people—who have surplus labor—can find hits (their interested tasks) to work for money, while the people—who upload working tasks—can get good results. Two different kinds of data sets become large with the increasing number of online users. Among which,

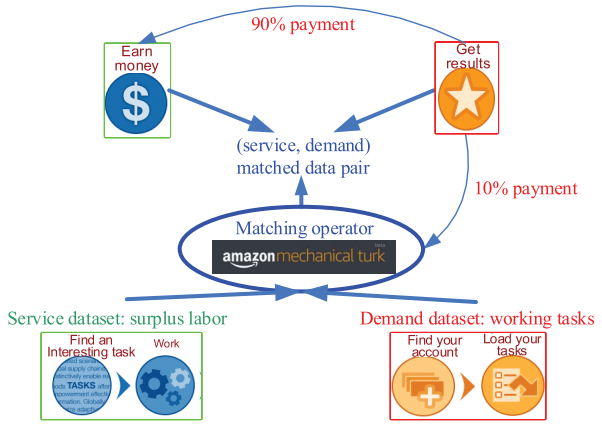


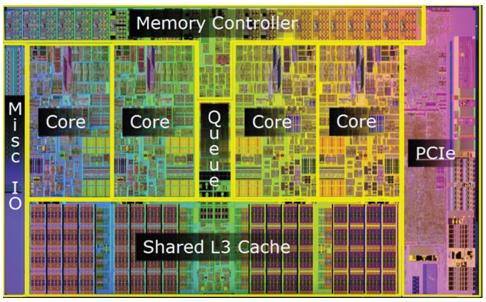
Fig. 1. Sharing economy model: Amazon Mechanical Turk.

the service data set includes a large volume of data describing the information about people's working place, working hours, and expected payment of doing good work. Similarly, a series of work descriptions is recorded in the demand data set. The big data analysis runs on the matching operator, i.e., Amazon artificial intelligence, to seek out an optimal service-demand matching solution, from two large data sets. Once a pair of service and demand is well matched, the corresponding deal—90% payment for the worker and residual 10% for the Amazon Mechanical Turk model—can be made.

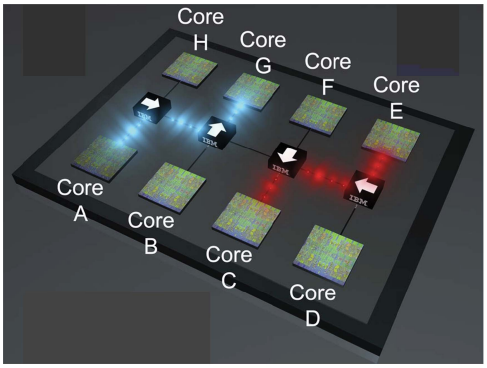
A. Motivation

Intuitively, the intelligent system's quick big data analysis plays a very important role in facilitating the optimal service-demand matching for sharing economy. As a famous Chinese quora knowledge website, "Zhihu" designed intelligent robots "Chinese monkey Wukong system" and "Zhihu Wall-E," deeply cutting useless services (e.g., Internet rumors) mismatched with the demand of knowledge acquisition [13]. To perform the optimal service-demand matching in a reasonable amount of time, these intelligent robots, together with the Amazon artificial intelligence, are equipped by high-performance intellectual property (IP) cores that work alongside a central processing unit (CPU) die, the so-called system-on-chip (SoC) [14] shown in Fig. 2(a).

To further reduce the memory contention among cores, the intelligent system supporting sharing economy urgently needs innovative computer architectures as accelerators. Decoupled from the bus architecture, discrete cores must be connected to the CPU via some kind of interconnect. Network-on-chip (NoC) [15], [16] is thus proposed. In Fig. 2(b), each core locally has a small size memory without contention, and cores are electrically interconnected via on-chip routers. However, with the increasing number of cores required to solve the optimal service-demand matching, the NoC's limited data switching rate becomes the bottleneck. The optical NoC (ONoC)—where high-speed waveguides replace metal wires to form the optical interconnection of cores—thus emerges [17]–[21].



(a)



(b)

Fig. 2. Traditional accelerators equipped in intelligent systems supporting sharing economy. (a) SoC instance. (b) NoC instance.

B. Contribution

In this paper, we design a novel ONoC-based computer architecture as the accelerator for the intelligent system supporting sharing economy, and the whole procedure of the service-demand matching is illustrated in Fig. 3. First of all, the intelligent system's controller divides the global service-demand matching task into subgroups, each of which corresponds to a pair of smaller services and demand data sets. For example, in Fig. 3, the service data set of Amazon Mechanical Turk model is decomposed into smaller sets by considering the length of working time. According to the required working competency, the demand data set also has the corresponding decomposition process. Next, the controller flexibly offers the IP core so that the intelligent system can access computing resources from the ONoC-based accelerator, searching the optimal matching solution for each subgroup. In Fig. 3, three subgroups' matching tasks are mapped onto different cores. Through the parallel computing in the form of the communication among mapped cores—termed as task-core mapping—the global service-demand matching can be quickly performed. Although our preliminary work [22] has discussed the task-core mapping based on ONoCs, it is designed to process industrial applications instead of the service-demand matching in sharing economy.

A careless behavior of placing subgroups' matching tasks on an impropriate combination of cores could cause serious power loss and crosstalk noise on the ONoC-based accelerator. Partial power could be coupled with the other signals, resulting in the crosstalk noise due to the existence of bend/crossing waveguides [23]. In addition, the accelerator's

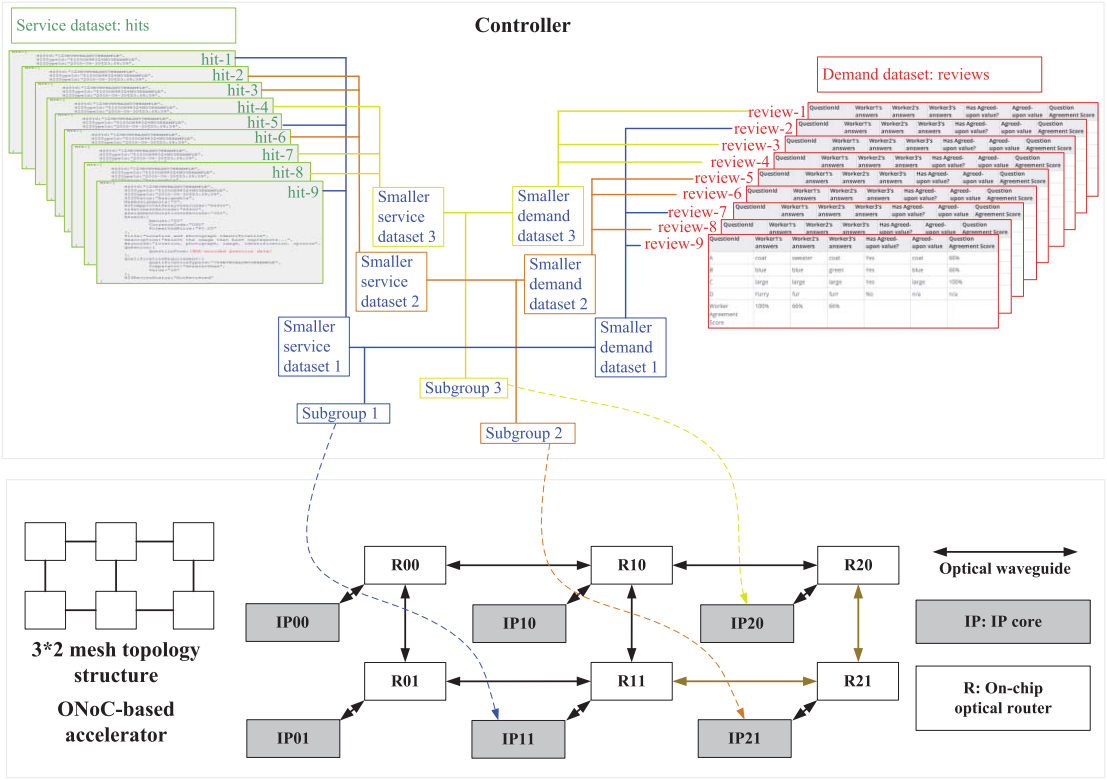


Fig. 3. Task-core mapping in the intelligent system supporting sharing economy.

thermal-sensitive device located at the hot spot may lose efficiency [24]. The aforementioned phenomenons drive the need to design reliability-aware task-core mapping schemes for the intelligent system supporting sharing economy. To facilitate the optimal service-demand matching while guaranteeing the high reliability of the ONoC-based accelerator, we propose a novel task-core mapping strategy termed as hybrid ant colony and genetic algorithm (ACGA). By using ACGA: the communication path along mapped cores can traverse the minimal number of bend/crossing waveguides. A good thermal balancing without hot spots can be ensured through achieving the appropriate distribution of the task loads separately tackled by cores. Simulation results show that our ACGA achieves the higher reliability over previous benchmarks. Furthermore, in Fig. 3, the mesh ONoC is utilized to support the interconnection of cores on the accelerator, but in fact, the other architectures—such as torus, ring, and butterfly—can also be taken into account. Different kinds of accelerator architectures may result in various levels of mapping efficiency. Thus, followed by ACGA, the optimal accelerator architecture can be determined from candidates for the service-demand matching.

C. Organization

The rest of this paper is organized as follows. Some concepts and definitions are described in Section II. The problem formulation is given in Section III. To solve the problem in a short span of time, we develop the heuristic algorithm in Section IV. We analyze numerical results in Section V before concluding this paper in Section VI.

II. Concepts and Definitions

We design the ONoC-based computer architecture as the accelerator of intelligent systems, determining the optimal service-demand matching for sharing economy. By using the controller embedded in the intelligent system, the global service-demand matching is divided into several subgroups' small tasks. We then place subgroups' tasks—that can be composed into a task graph—on the appropriate cores of candidate accelerator architectures, i.e., task-core mapping. Thus, the concepts of task graph, accelerator architecture, and task-core mapping are defined in the following.

A. Task Graph

In addition to the requirement of calculating the service-demand matching, a task graph also represents the communication mode and the bandwidth requirement among virtual nodes. Here, the virtual node corresponds to the subgroups small task after division. Thus, the task graph has the following definition.

Definition 1: The global service-demand matching—divided into $|C|$ subgroups—can be denoted as the directed task graph $G(C, E)$. Among which, C and E are the sets of virtual nodes and communication links, respectively. The link $e_{ij} \in E$ has the weight of the bandwidth requirement supporting the communication from $c_i \in C$ to $c_j \in C$. An example is shown in the left part of Fig. 4(a). The global service-demand matching task is divided into $|C| = 9$ subgroups, i.e., $\{c_1, c_2, \ldots c_9\}$. The number besides the link denotes the bandwidth requirement given in megabyte per second.

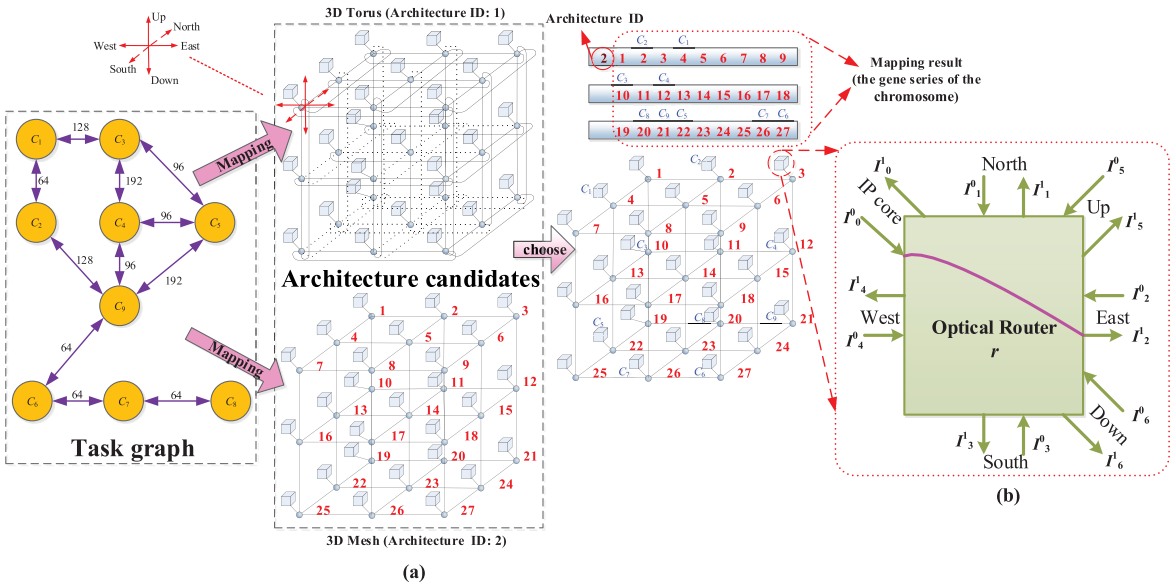


Fig. 4. Our task-core mapping process and simplified router model. (a) Illustration of our task-core mapping process. (b) Seven-port simplified model of ORs.

B. Accelerator Architecture

A kind of ONoC-based accelerator architecture is defined as a particular mode of the optical interconnection among cores. Thus, the following definition is given.

Definition 2: An ONoC-based accelerator architecture can be represented as a directed graph $G(R, L)$, where R and L are the sets of cores and intercore communication links, respectively. The virtual node—coming from the task graph—can be placed on only one core r $(r = 1, 2, 3 \ldots, |R|)$. Every core is locally connected to an optical router (OR). Two candidate architectures, 3-D torus and 3-D mesh, are shown in the middle part of Fig. 4(a).

C. Task-Core Mapping Process

There are several alternatives of accelerator architectures, and one of which is superior to the others based on a certain objective criteria. Thus, on every architecture candidate, we place the virtual nodes coming from the task graph onto cores, and a possible mapping solution can be recorded as the chromosome. In each chromosome, the first number is the ID of the selected architecture while the following gene series is the mapping result. As an example of the chromosome in the right part of Fig. 4(a): 1) the first number "2" means that 3-D mesh architecture is selected for the corresponding possible mapping solution and 2) in the gene series, the second item "2" is the ID of the core in 3-D mesh architecture, while the notation c_2 on the top of the second gene item "2" is the virtual node, i.e., c_2 is mapped onto the core r_2 in 3-D mesh architecture. There are numerous possible solutions because the virtual node can be mapped on any core, then the optimal mapping solution that has the best objective function value should be determined.

III. Problem Formulation

What is the objective criteria during the process of task-core mapping? As described in Section I-B, the answer is power loss, crosstalk noise, and thermal balancing. Among which, power loss and crosstalk noise can be together evaluated by signal-to-noise-ratio (SNR). Thus, SNR and thermal balancing are both optimized to guarantee the high accelerator reliability of the intelligent system supporting sharing economy.

In this section, the mathematical representations of SNR and thermal balancing are given in prior. Next, we formulate a multiobjective optimization function with the joint consideration of these mathematical models.

A. SNR Model

Proposition 1: The waveguides and microresonators (MRs)—extensively used in ONoC ORs—have undesirable crosstalk brought by the coupling among optical signals. On the basis of this, the SNR in (1) is the ratio of the received signal power to the total power of the crosstalk

$$\mathrm{SNR} = 10 \cdot \log\left(\frac{P_S}{P_C}\right). \tag{1}$$

Proposition 2: By virtue of the dimension-order routing, the macroscopic path P_a should be determined from the source OR to the destination OR on the ONoC-based accelerator architecture, once two virtual nodes have been mapped on a pair of cores/ORs. Next, given the initial signal power P_{ij} injected into the core of the source OR, the final signal power P_S—received by the core of the destination OR—can be obtained in the following:

$$P_S = P_{\mathrm{ij}} \cdot \prod_{r \in P_a} L^r_{I_i^0, I_j^1}. \tag{2}$$

Proof for Proposition 2: We first use $L^r_{I_i^0, I_j^1}$ to define the power loss of the OR r—which has the useful signal from the input port I_i^0 $(I_i^0 \in r)$ to the output port I_j^1 $(I_j^1 \in r)$. Usually, $L^r_{I_i^0, I_j^1}$ includes the power loss from bend waveguides, crossing waveguides, and MRs along the microscopic path.

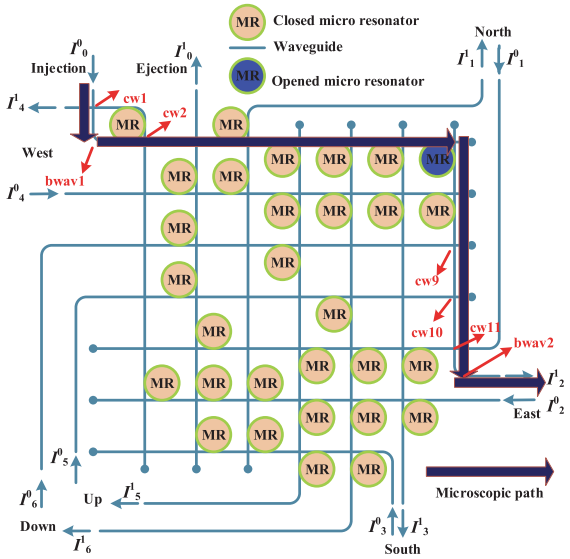


Fig. 5. Microscopic path within OR r.

As an example of Fig. 4(b), the signal traverses the OR r from the core (i.e., $I_0^0 \in r$) to the East (i.e., $I_2^1 \in r$), then the microscopic path will include two bend waveguides (bwav1 and bwav2), five closed MRs, one opened MR, and 11 crossing waveguides (from cw1 to cw11) in Fig. 5. So, the corresponding power loss $L^r_{I_0^0,I_2^1}$ is given in the following:

$$L^r_{I_0^0,I_2^1} = (L_b)^2 \cdot (L_{\text{mic,off}})^5 \cdot (L_{\text{mic,on}}) \cdot (L_c)^{11} \quad \forall r \in [1, |R|] \tag{3}$$

where L_b, $L_{\text{mic,off}}$, $L_{\text{mic,on}}$, and L_c are the values of the power loss of a bend waveguide, a closed MR, an opened MR, and a crossing waveguide, respectively.

As a result, the final power loss is the product of power loss values each of which has been determined for the corresponding OR traversed by the macroscopic path P_a according to (3).

Proposition 3: The power of the crosstalk noise is

$$P_C = \sum_{r \in P_a} \left(C^r \cdot \frac{P_S}{P_{\text{ij}} \cdot L^r} \right). \tag{4}$$

Proof for Proposition 3: The crosstalk noise of OR r in (5) comes from adjacent ORs

$$C^r = \sum_{r_h} (L^{r_h} \cdot c_h) \quad \forall r \in [1, |R|] \tag{5}$$

where r_h is the adjacent OR; c_h is the crosstalk noise of r_h; L^{r_h} is the power loss of r_h when the crosstalk is transmitted from r_h to r.

As an example of the OR r designated in Fig. 6(a), the crosstalk noise generates from adjacent ORs r_1, r_2, r_3, and r_4. In Fig. 6(b), the crosstalk noise c_1 of r_1 is injected into r through the input Up port I_5^0 ($I_5^0 \in r$), the crosstalk noise c_2 of r_2 is injected into r through the input West port I_4^0 ($I_4^0 \in r$), the crosstalk noise c_3 of r_3 is injected into r through the input East port I_2^0 ($I_2^0 \in r$), and the crosstalk noise c_4 of r_4 is injected into r through the input North port I_1^0 ($I_1^0 \in r$). Correspondingly, in Fig. 6(c), the worst case crosstalk noise c_1 traverses r from I_5^0 to the output port I_0^1 ($I_0^1 \in r$), the worst case crosstalk noise c_2 traverses r from I_4^0 to the output port I_2^1 ($I_2^1 \in r$), the worst case crosstalk noise c_3 traverses r from I_2^0 to the output port I_4^1 ($I_4^1 \in r$), and the worst case crosstalk noise c_4 traverses r from I_1^0 to the output port I_0^1 ($I_0^1 \in r$).

Then, $c_1 = L^r_{I_5^0,I_0^1} \cdot K_c$ because the microscopic path of c_1 traverses the crossing waveguide cw that results in the crosstalk noise K_c in Fig. 6(c). Note that K_c is generated along with the power loss $L^r_{I_5^0,I_0^1}$ determined by the computing method mentioned in Proposition 2. The power loss of L^{r_1} is $L^{r_1}_{I_0^0,I_6^1}$, when c_1 is transmitted from the local core I_0^0 ($I_0^0 \in r_1$) to the output Down port I_6^1 ($I_6^1 \in r_1$). Using the computing method mentioned in Proposition 2, we can obtain $L^{r_1}_{I_0^0,I_6^1}$, i.e., L^{r_1}. Similarly, we compute the other values of crosstalk noise and power loss so that the crosstalk noise C^r can be found.

As a result, the final power of the crosstalk is the sum of noise values each of which has been determined for the corresponding OR traversed by the macroscopic path P_a according to (5).

Proposition 4: The total SNR $\text{SNR}_{\text{total}}$ is

$$\text{SNR}_{\text{total}} = \sum_{P} \text{SNR}. \tag{6}$$

Proof for Proposition 4: Given P that is the set of macroscopic paths established to support the communication among virtual nodes, the total SNR $\text{SNR}_{\text{total}}$ is the sum of SNR values each of which has been obtained according to (1). Intuitively, a high $\text{SNR}_{\text{total}}$ denotes a good accelerator reliability.

B. Thermal Balancing Model

After task-core mapping, the core—which has the task load far beyond the average value—becomes the hot spot whose thermal becomes much higher than the others, leading to the thermal shift of relative MRs.

To mitigate the wavelength mismatch between the input signal and the resonator brought by the MRs' thermal shift, a good load-thermal balancing (LTB) defined in the following is very necessary:

$$\text{LTB} = \frac{1}{|R|} \cdot \sum_{r=1}^{|R|} (\Delta tf_r \times \delta_r) \tag{7}$$

where Δtf_r is the absolute difference between the task load tf_r of the OR r and the average value, that is

$$\Delta tf_r = \left| tf_r - \frac{1}{|R|} \cdot \sum_{r=1}^{|R|} tf_r \right|. \tag{8}$$

In addition, δ_r is the impact factor of thermal balancing, and it is relative to the core's geographical location in the accelerator. Then, we have

$$\delta_r = e^{-\beta \cdot \text{dis}(r,r^*)} \tag{9}$$

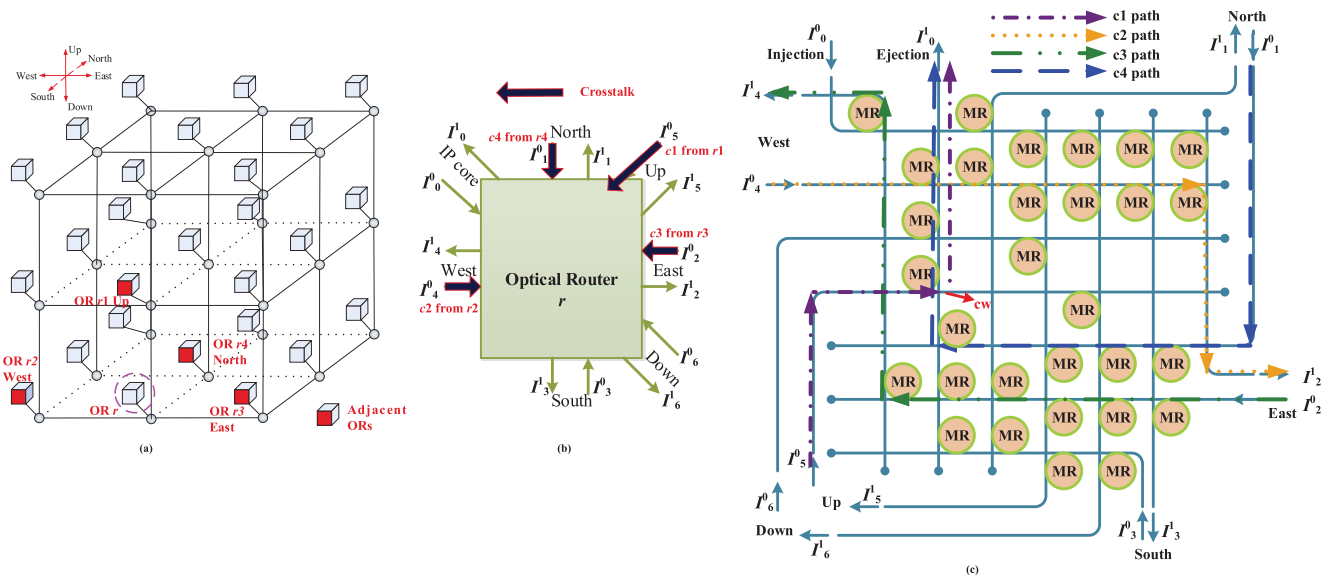


Fig. 6. Crosstalk analysis of OR r.

where $\mathrm{dis}(r, r^*)$ is the physical distance between r and the center core r^*; β is a distance impact factor.

For the 3-D accelerator architecture, we utilize a 3-D coordinate (x, y, z) to represent the geographical location of cores. Hence

$$\mathrm{dis}(r, r^*) = \sqrt{(x_r - x_{r^*})^2 + (y_r - y_{r^*})^2 + (z_r - z_{r^*})^2}. \quad (10)$$

In summary, a good LTB will be achieved if there is a small LTB, thus ensuring a high accelerator reliability.

C. Optimization Objective Function

On the basis of SNR and thermal balancing models, we have the following optimization objective function:

$$\min\left\{\frac{(\mathrm{LTB})^{\alpha_B}}{(\mathrm{SNR}_{\mathrm{total}})^{\alpha_S}}\right\} \quad (11)$$

where α_B and α_S are the weight factors of thermal balancing and SNR, respectively.

In fact, (11) has two main parts: the denominator SNR defined in (1), and the molecules LTB defined in (8). Different mapping results may cause various objective function values. However, the optimal mapping solution should have the lowest function value (i.e., the smallest LTB and the biggest SNR).

In addition, the aforementioned optimization objective function should comply with the following constraints:

$$\forall c_i \in C, \quad \exists r \in R, \ r = \mathrm{mapping}(c_i) \quad (12)$$

and

$$\forall c_i \neq c_j \in C, \quad \mathrm{mapping}(c_i) \neq \mathrm{mapping}(c_j) \in R \quad (13)$$

and

$$P_S \geq P_{\min}. \quad (14)$$

Here, (12) and (13) are the constraints that ensure the core and the virtual node must have one-to-one mapping relationship. Equation (14) is the constraint of the received signal power, and $P_{\min}$ is the receiver sensitivity.

IV. Heuristic Algorithm

In this section, we propose a novel heuristic scheme for the task-core mapping based on hybrid ACGA, in order to optimize the value of the objective function (i.e., minimizing the mapping cost) defined in (11). In our ACGA, the ant colony algorithm (ACA) (see Section IV-A) is used to globally search the local optimal solution m'_{best} according to the probability of mapping the virtual node onto the core. It is worth to noticing that the mapping probability can be varied through updating the ants' pheromone concentration (pls. see Sections IV-B1 and IV-B5). The mapping solution m'_{best} may be replaced by the global optimal one m_{best} generated via executing a series of genetic operations that includes 3OPT (see Section IV-B2), encoding-based hybrid (see Section IV-B3), and roulette selection (see Section IV-B4).

Fig. 7 illustrates the procedure of our ACGA during one iteration. The task graph is shown in Part 1 while two candidate accelerator architectures, 3-D mesh and 3-D torus are listed in Part 2. In Part 3, each mapping solution is represented by the chromosome that includes the first number recording the selected architecture's ID and the following gene series corresponding to a possible mapping result. The length of the gene series is equal to $|R|$. Specifically, the nonzero item in the gene series denotes one virtual node's ID while the item's location in the chromosome is the ID of the mapped core. The item "0" in the gene series means that the corresponding core has not been mapped by any virtual node yet. As an example of the first chromosome represented as 1, {0, 0, 1, 3, 0, 7, 0, 0, 0, 2, 0, 0, 4, 0, 0, 5, 0, 6, 0, 0, 0, 0, 8, 0, 0, 9, 0} in Part 3 of Fig. 7: 1) the first number "1" is the ID of the selected architecture and 2) the part {•} denotes the gene series, and for example, the first item "0" of the gene series means that the core with the ID "1," i.e., r_1 has not been mapped by any virtual node yet, the fourth item "3" of the gene series means that the core r_4 has been mapped by the virtual node c_3.

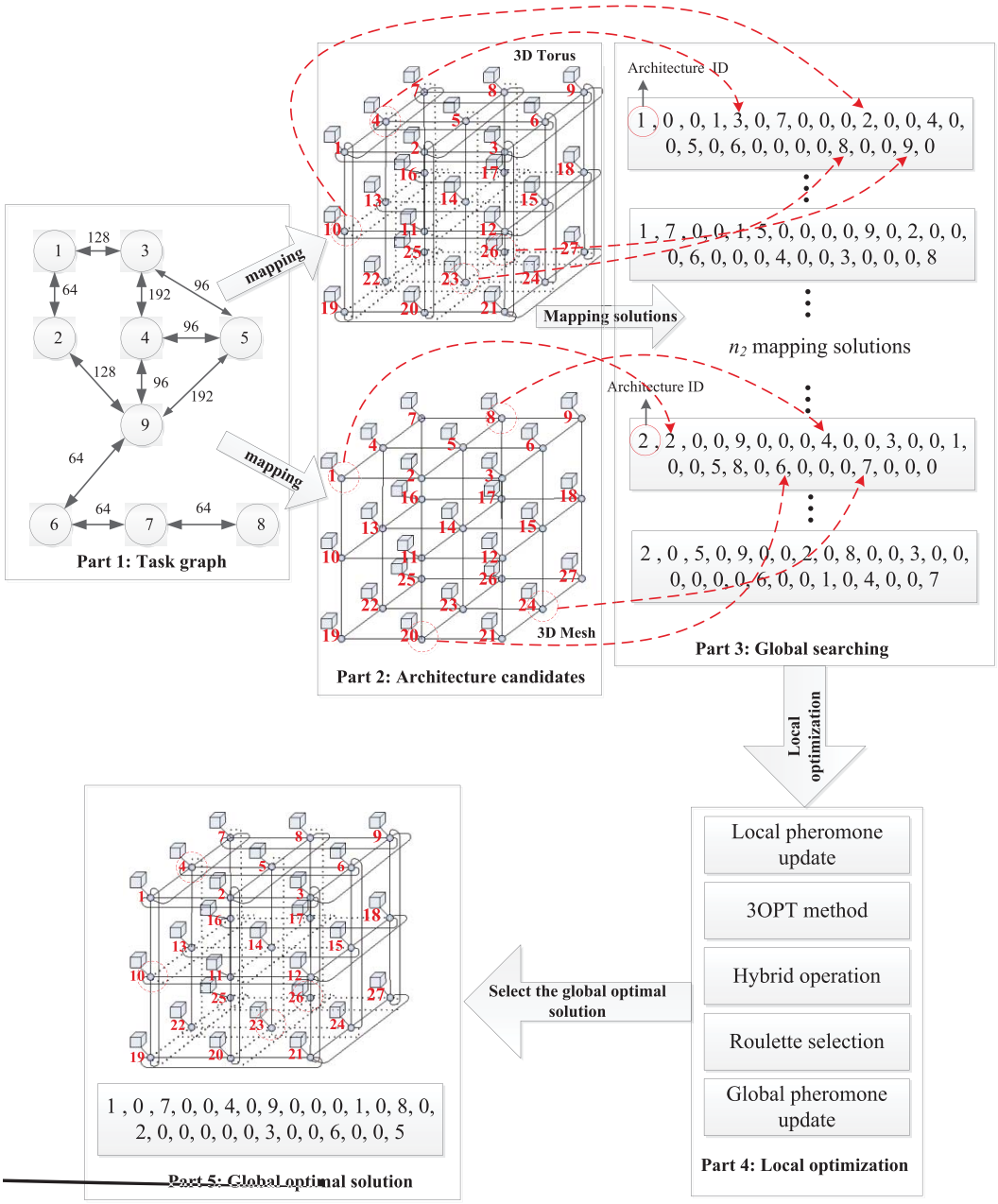


Fig. 7. Illustration of ACGA during one iteration.

The detail procedure of ACGA is described as follows.

Step 1: Initialize the task graph (e.g., Part 1 of Fig. 7) and candidate architectures (e.g., Part 2 in Fig. 7) of the ONoC-based accelerator.

Step 2: On each candidate architecture, the ant tries to place the virtual node c_i on the idle core r_j along the allocated foraging trajectory, according to the probability P_{ij} codetermined by heuristic information η_{ij} and pheromone τ_{ij} in (15). As a result, there are n_2 mapping solutions since we totally have n_2 ants.

Step 3: Eliminate the solutions that cannot satisfy the constraints in (12)–(14), compute the mapping cost $f()$ for each solution according to (11), and finally determine the local optimal solution m'_{best} that has the minimal mapping cost at the current iteration. An example shown in Part 3 of Fig. 7.

Step 4: Execute the following local optimization operations, as illustrated in Part 4 of Fig. 7. Specifically, update the local pheromone matrix according to (16). Search the new solution at the neighborhood of m'_{best} by using 3OPT method, and compute the corresponding mapping cost. If the mapping cost is lower than the current $f(m'_{best})$, update the global optimal solution as m_{best}. Continuingly, the new solution is further generated through the hybrid of two solutions obtained by using roulette selection, and we compute the corresponding mapping cost. If the mapping cost is lower than the current $f(m_{best})$,

update the global optimal solution m_{best}; finally, we update the global pheromone matrix according to (19).

Step 5: If the current iteration is the maximal generation, output m_{best} and $f(m_{\text{best}})$; otherwise, go back to **step 2**.

A. Global Searching

As mentioned above, the ant k places the virtual node c_i on the idle core r_j according to the unique probability P_{ij} codetermined by heuristic information η_{ij} and pheromone τ_{ij}, thus, we have

$$P_{i,j} = \frac{[(\tau_{\text{ij}}(t))^{\alpha}] \cdot (\eta_{\text{ij}})^{\iota}}{\sum_{u \in \text{allowed}_k} [(\tau_{iu}(t))^{\alpha}] \cdot (\eta_{iu})^{\iota}}, \quad u \in \text{allowed}_k. \tag{15}$$

Here, allowed_k is the set of idle cores and it thus initially has $(|R|-1)$ elements except from the first core along the foraging trajectory allocated for the ant k. Then, the size of allowed_k gradually shrinks over time. α and ι are the importance factors of heuristic information and pheromone, respectively, which simulates the ant's memory ability. In addition, η_{ij} remains constant once it is determined by the given information about task graph and relative architecture, while the pheromone $\tau_{\text{ij}}(t)$ should be dynamically updated over time, i.e., $\tau_{\text{ij}}(t) = (f_i/d_j)$. It means that the virtual node—which has large task loads between itself and the others—tends to be placed on the center core. Here, d_j denotes the total distance between r_j and the other cores, and it is determined according to the candidate architecture's information. The core r_j that has a smaller d_j is more likely to be located in the architecture center; f_i denotes the total task load between c_i and the other virtual nodes, and it is determined by the information of a given task graph.

B. Local Optimization

We orderly execute the following operations to perform the local optimization.

1) Local Pheromone Update: In our ACGA, we have two approaches of updating the pheromone matrix: local pheromone update and global pheromone update. First of all, once one iteration of global searching is completed, i.e., the local optimal solution m'_{best} is determined, we immediately execute the following local pheromone update:

$$\tau_{\text{ij}}(t+1) = \rho \cdot \tau_{\text{ij}}(t) + \frac{Q}{f(m_k)}. \tag{16}$$

Here, $f(m_k)$ is the mapping cost of the solution m_k found by the kth ant; ρ is the weight factor of the local pheromone; Q is the experiential parameter.

2) 3OPT Method: After finding the local optimal mapping solution m'_{best} through the global searching, the re-encoding is conducted for the chromosome of m'_{best}. The gene item of the re-encoded chromosome still denotes the ID of one virtual node, while the location of the gene item still denotes the core's ID. Based on 3OPT, the detail re-encoding procedure can be described as follows: 1) randomly select three genes from the chromosome, exchange their locations to generate five new chromosomes, and compute their mapping cost and 2) compare the mapping cost between five new chromosomes and their parent m'_{best}, and then make the hybrid operation presented later on the best two chromosomes m'_0 and m'_1, so as to form the new chromosome, i.e., the new solution.

3) Hybrid Operation: The main idea is to exchange the partial set of gene items between chromosomes m'_0 and m'_1 obtained by using 3OPT method, thus generating the new chromosome/mapping solution. Take the following operation as an example, where the chromosome of m'_0 is represented as

$$2, \{1, 0, 0, 7, |9, 0, 0, 0, 3, 0, 0, 5, 0, 2, 0, 0, 0, 8, 0, | \\ 0, 0, 6, 0, 0, 0, 0, 4\}$$

and the chromosome of m'_1 is represented as

$$1, \{3, 8, 0, 0, |0, 0, 2, 0, 1, 0, 0, 7, 0, 0, 5, 0, 0, 0, 6, | \\ 0, 4, 0, 0, 0, 0, 9, 0\}.$$

Here, the first number of two chromosomes is the architecture ID, $\{\bullet\}$ denotes the gene series, while $\{|\bullet|\}$ denotes the hybrid domain. Exchange their hybrid domains, and for each new chromosome, add the exchanged hybrid domain to the beginning of the original gene series. Thus, we have

$$m''_0 = 2, \{|0, 0, 2, 0, 1, 0, 0, 7, 0, 0, 5, 0, 0, 0, 6, | \\ 1, 0, 0, 7, 9, 0, 0, 0, 3, 0, 0, 5, 0, 2, 0, 0, 0, 8, 0, \\ 0, 0, 6, 0, 0, 0, 0, 4\}$$

and

$$m''_1 = 1, \{|9, 0, 0, 0, 3, 0, 0, 5, 0, 2, 0, 0, 0, 8, 0, | \\ 3, 8, 0, 0, 0, 0, 2, 0, 1, 0, 0, 7, 0, 0, 5, 0, 0, 0, 6, \\ 0, 4, 0, 0, 0, 0, 9, 0\}.$$

For the short gene series behind the hybrid domain of each chromosome, delete the gene items included by the hybrid domain. For example, in the short gene series behind the hybrid domain of m''_0, i.e., $\{1, 0, 0, 7, 9, \ldots, 0, 4\}$, since gene items $\{1, 2, 5, 6, 7\}$ emerge in the hybrid domain $\{|0, 0, 2, \ldots 0, 6|\}$, delete $\{1, 2, 5, 6, 7\}$ from $\{1, 0, 0, 7, 9, \ldots, 0, 4\}$, then we have

$$m''_0 = 2, \{|0, 0, 2, 0, 1, 0, 0, 7, 0, 0, 5, 0, 0, 0, 6, | \\ 0, 0, 9, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 8, 0, \\ 0, 0, 0, 0, 0, 0, 4\}$$

similarly

$$m''_1 = 1, \{|9, 0, 0, 0, 3, 0, 0, 5, 0, 2, 0, 0, 0, 8, 0, | \\ 0, 0, 0, 0, 0, 1, 0, 0, 7, 0, 0, 0, 0, 0, 6, \\ 0, 4, 0, 0, 0, 0, 0\}.$$

Next, randomly delete the gene items labeled by "0" from two chromosomes, until they have the same length as original m'_0 and m'_1, for example

$$m''_0 = 2, \{|0, 2, 0, 1, 0, 0, 7, 0, 0, 5, 0, 0, 0, 6, | \\ 0, 0, 9, 0, 0, 0, 3, 0, 0, 0, 0, 8, 4\}$$

and

$$m''_1 = 1, \{|9, 0, 0, 0, 3, 0, 5, 0, 2, 0, 0, 0, 8, | \\ 0, 0, 0, 1, 0, 0, 7, 0, 6, 0, 4, 0, 0, 0\}.$$

Finally, the better chromosome that has a lower mapping cost is the new solution.

4) Roulette Selection: As previously discussed, the solution that has a lower mapping cost tends to be selected. On the basis of this principle, the roulette selection probability of the mapping solution m_k has the following definition:

$$\mathrm{RP}_k = \frac{1}{f(m_k) \cdot \sum_{k=1}^{n_2} \frac{1}{f(m_k)}}. \tag{17}$$

All possible mapping solutions compose one roulette because each of them occupies the fan part of the roulette. The fan area is relative to the solution's mapping cost. The solution m_k that has a lower mapping cost $f(m_k)$ occupies a bigger fan area, thus easier being selected when the roulette stops rotating. The aforementioned procedure can be simulated by the following operations: 1) randomly generate a number υ within the scope [0, 1] and 2) if $\upsilon \leq S_1$, m_1 is selected as the new mapping solution; if $(S_k - 1) < \upsilon \leq S_k$, $(2 \leq k \leq n_2)$, the corresponding mapping solution m_k $(k = 1, 2, \ldots, n_2)$ is selected. Here, S_k is the cumulative probability of m_k, and it is defined in the following:

$$S_k = \sum_{i=1}^{k} \mathrm{RP}_i. \tag{18}$$

5) Global Pheromone Update: After updating the global mapping solution m_{best} through executing aforementioned operations, we also need to update the pheromone along the foraging trajectory allocated for the corresponding ant as follows:

$$\tau'_{\mathrm{ij}}(t+1) = \tau_{\mathrm{ij}}(t+1) + \delta \cdot \frac{Q}{f(m_{\mathrm{best}})}. \tag{19}$$

Here, δ is the weight factor of the global pheromone.

C. Complexity Analysis

The pseudocode of our ACGA is shown in Algorithm 1. As shown in Algorithm 1, at the worst case, the maximal number of loop operations is $(n_1 \times n_2)$. Here, n_1 is the maximal number of iterations.

V. Simulation and Discussion

In this section, we first introduce simulation settings and benchmarks. We then discuss comparative results between our ACGA and benchmarks.

A. Simulation Setup and Benchmarks

ACGA is compared with GA, ACA, and the cataclysm genetic-based simulated annealing (CGSA) proposed by us in [25]. To ensure the fairness, the same objective function described in (11) is used. In (11), we let $\alpha_B = \alpha_S = 1$ since thermal balancing and SNR are equally important. We generate task graphs according to the computation requirement of instances in the Amazon Mechanical Turk [9]. Two candidate accelerator architectures, 3-D mesh and 3-D torus, are taken into account, and they have the same scale, i.e., the same number of cores $|R|$. In order to quantitatively calculate the SNR value, the configuration of insertion loss parameters is given in the following: $L_b = 0.005$ dB/90°, $L_c = 0.12$ dB, $L_{\mathrm{mic,off}} = 0.005$ dB, and $L_{\mathrm{mic,on}} = 0.5$ dB. The setting of other important parameters is given in Table I.

Algorithm 1 ACGA

Require: Task graph, candidate accelerator architectures and so forth.
Output: Optimal solution m_{best} that has the lowest mapping cost $f(m_{best})$.

```
Initialize the pheromone matrix Λ;
for i = 1, 2, ..., n1 do
  for ant k ∈ [1, n2] do
    Generate the local optimal solution m'_best based on global searching;
    Calculate the mapping cost f(m'_best) according to Eq. 11;
    Update the pheromone matrix Λ according to Eq. 16;
  end for
  Search the new mapping solution at the neighborhood of m'_best by using 3OPT method;
  Execute hybrid operation on two chromosomes obtained by using 3OPT method, generating the new mapping solution m_best;
  Execute hybrid operation on two chromosomes obtained by using roulette selection, further generating the new mapping solution m_best;
  if f(m'_best) > f(m_best) then
    Renew m'_best as m_best;
  end if
  Update the pheromone matrix Λ according to Eq. 19;
end for
Return m_best and f(m_best).
```

TABLE I
Parameter Settings for Simulations

Parameter	Physical meaning	Value
n_1	Maximal number of iterations	200
n_2	Total number of ants	50
α	Importance factor of heuristic information	1
ι	Importance factor of pheromone	5
Q	Experiential parameter	1
ρ	Weight factor of the local pheromone	0.9
δ	Weight factor of the global pheromone	0.3
$\tau_{ij}(0)$	Initial pheromone	1

B. Simulation Results

The setting of some parameters greatly affects the algorithms' efficiency. First of all, we determine the most appropriate value for some important parameters through simulations, so as to gain an expected mapping solution.

In Section IV-B2, we exchange only three genes' locations between two chromosomes. In fact, a greater number of exchanged genes can further expand the diversity of superior chromosomes/mapping solutions, but meanwhile, the algorithm complexity exponentially increases. To obtain an appropriate number of exchanged genes, through randomly generating the task graphs each of which has $|C| = \{25, 56, 81\}$, the objective function value of our ACGA is evaluated in $5 \times 5 \times 5$ candidate accelerator architectures

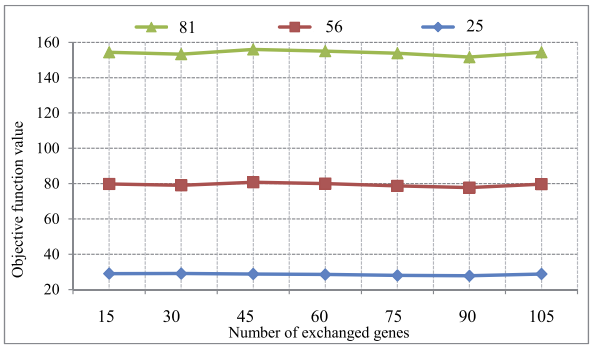


Fig. 8. Objective function value varying with the increasing number of exchanged genes.

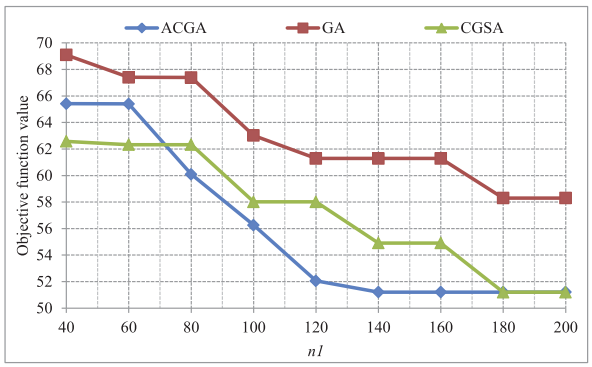


Fig. 9. Objective function value varying with the rising of n_1.

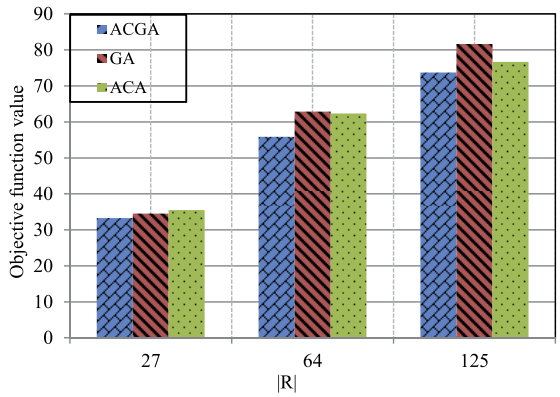


Fig. 10. Objective function value varying with the rising scale of accelerator architectures.

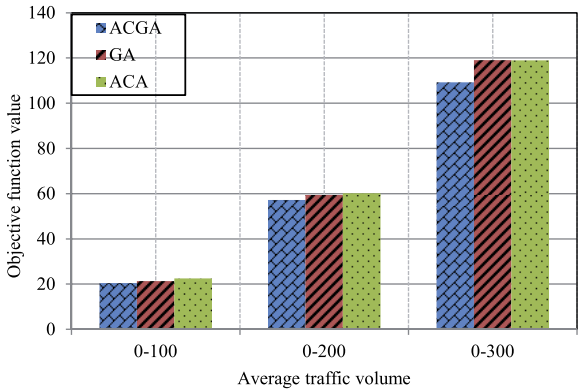


Fig. 11. Objective function value varying with the rising scale of traffic volume.

where $|R| = 125$. At each test point, we calculate the average of 100 values in Fig. 8. We can see that the optimal (lowest) objective function value can be obtained when we have 90 genes to exchange.

On the other hand, as the maximal number of iterations, n_1 also plays an important role in affecting the algorithms' efficiency. The algorithm using a small n_1 cannot get a fast convergence, while it also consumes the high time cost under a big n_1. To obtain the appropriate value of n_1, we evaluate the objective function value among GA, ACGA, and CGSA algorithms when $|R| = 125$ and $|C| = 56$ in Fig. 9. We can see that compared with the traditional GA, our ACGA and previously proposed CGSA own the better mapping solution. Especially when $n_1 = 140$, our ACGA is the first one to get convergence because the optimal solution remains unchanged, by virtue of strong local searching ability. Until $n_1 = 180$, the other algorithms get convergence. In other words, the time consumed for preforming 40 iterations can be saved by our ACGA to get the real optimal value (i.e., the gap between the performance of our algorithm and the real optimal value is 0) compared with other methods. Obviously, our ACGA has the best performance. Whatever the algorithm we use, when $n_1 = 200$, the convergence can be achieved. Thus, we let $n_1 = 200$ in the following simulations.

Next, when $|C| = \{27, 64, 125\}$ in Fig. 10, we compare the objective function value among our ACGA, GA, and ACA in different scaled accelerator architectures, i.e., $|R| = \{27, 64, 125\}$. We can see that since our ACGA considers local optimization strategies, the better mapping solution can be obtained. With the increased architecture scale, the performance advantage brought by our ACGA becomes gradually obvious. The corresponding improvement ratios over ACA and GA are 9.7% and 10.35%, respectively. In Fig. 11, we compare the same performance metric among these three algorithms when $|R| = 125$ and $|C| = 100$. The average traffic volume along the X-axis denotes the requirement of the communication bandwidth between two virtual nodes. The simulation results demonstrate that our ACGA also has the better mapping solution than that of the other two algorithms. In addition, with the increased traffic volume, the performance advantage follows a rising trend. The corresponding improvement ratios over ACA and GA are 8.26% and 8.19%, respectively.

Given that the average traffic volume is 300, we compare objective function value and SNR (higher SNR is better) among our ACGA, GA, and CGSA when $|R| = 125$, in Figs. 12 and 13, respectively. Under different numbers of virtual nodes in task graphs, GA performs the worst because it lacks the measures of overcoming its premature convergence. Under the case of the small task graphs that have less virtual nodes, CGSA gains the best mapping solution. However, our ACGA performs the best for big tasks because it has the strong ability of global searching. For the big data analysis of the large-scale service-demand matching in sharing

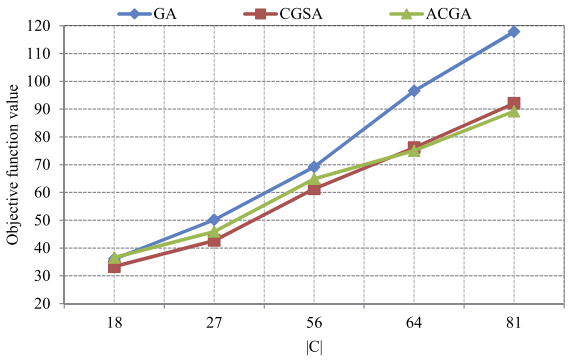


Fig. 12. Comparison of objective function value under different numbers of virtual nodes.

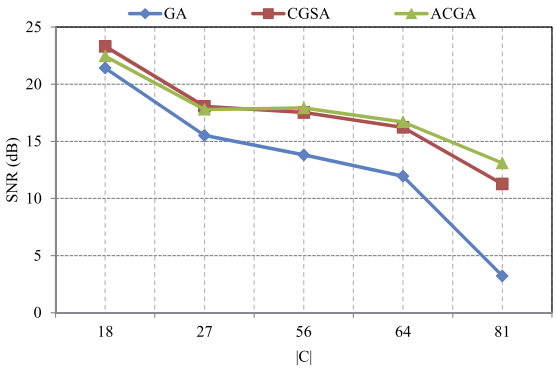


Fig. 13. Comparison of SNR under different numbers of virtual nodes.

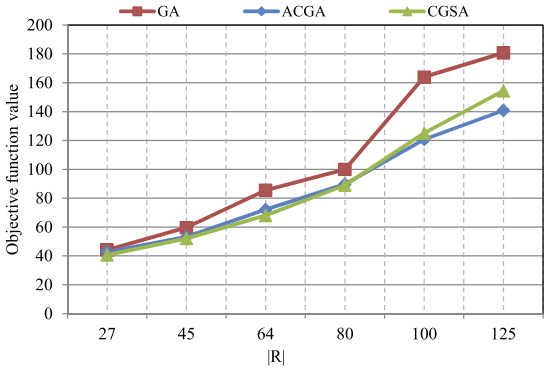


Fig. 14. Comparison of objective function value under different architecture scales.

economy, our ACGA undoubtedly becomes the mainstream. In addition, the improvement ratios of objective function value and SNR over GA are only 19.3% and 16.6% for CGSA, while 21.7% and 21.9% for our ACGA that also demonstrates the superiority of our ACGA.

Given that the average traffic volume is 300, we also compare the objective function value and SNR among our ACGA, GA, and CGSA under different scaled accelerator architectures when $|C| = 27$, in Figs. 14 and 15, respectively. We can see that when $|R| = 27$, both ACGA and CGSA obtain the global optimal mapping solution within the expected number of iterations, while GA cannot do that. With the increased architecture scale, the performance of ACGA and

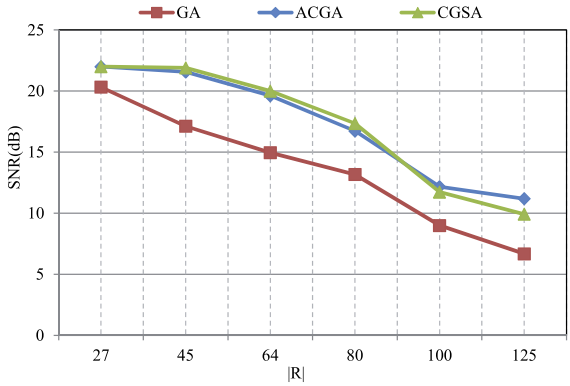


Fig. 15. Comparison of SNR under different architecture scales.

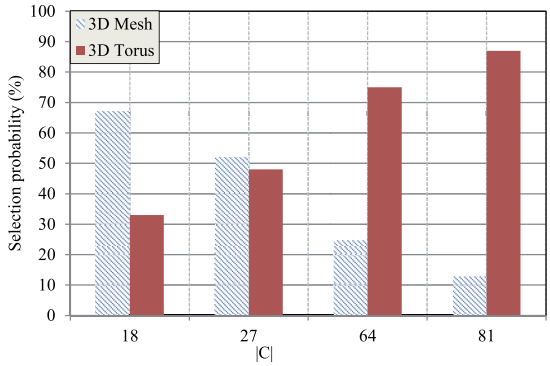


Fig. 16. Comparison of the selection probability of candidate accelerator architectures under different numbers of virtual nodes.

CGSA is still better than that of GA, and this performance advantage follows a rising trend. Similarly, when the architecture scale becomes large, our ACGA performs the best. In summary, for the big data analysis of the large-scale service-demand matching in sharing economy, our ACGA is the most appropriate selection.

In the following, we run our ACGA on two different accelerator architectures, and then we check which architecture is more frequently selected by mapping solutions when $|R| = 125$. Given that the average traffic volume is within the scope [0, 200] in Fig. 16, the simulation results show that: 1) most of mapping solutions select 3-D mesh when task graphs have less virtual nodes and 2) with the increasing number of virtual nodes in each task graph, 3-D torus is more frequently selected by mapping solutions, and it has an absolute advantage when $|C| = 81$. It means that 3-D torus is the more appropriate accelerator architecture used for the big data analysis of the large-scale service-demand matching in sharing economy.

Finally, when $|C| = 27$ in Fig. 17, we also compare the selection probability of candidate accelerator architectures under different values of the average traffic volume. We can see that when the average traffic volume is within the scope [0, 200], two architectures have the similar selection probability. However, when the traffic volume reaches up to [0, 300], 3-D torus is more frequently selected, which also demonstrates that 3-D torus is the more appropriate accelerator architecture

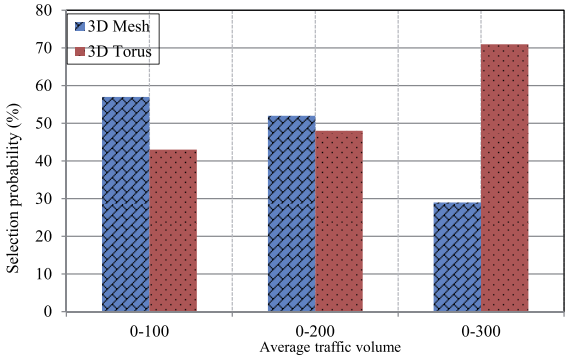


Fig. 17. Comparison of the selection probability of candidate accelerator architectures under different values of average traffic volume.

used for the big data analysis of the large-scale service-demand mapping in sharing economy.

VI. Conclusion

It is very necessary to perform the big data analysis of the service-demand matching through intelligent systems designed by the sharing economy enterprise. In this paper, we proposed a novel reliability-aware task-core mapping algorithm, based on the ONoC-based accelerator functioned as a new computer architecture for intelligent systems supporting sharing economy. Our primary aim was to optimize SNR and thermal balancing of the accelerator architecture while ensuring the optimal service-demand matching. We formulated this optimization problem and designed a novel heuristic scheme ACGA to determine the most appropriate accelerator architecture that had the optimal task-core mapping solution for the service-demand matching. Our ACGA achieved higher accelerator reliability in comparison to benchmarks, and 3-D torus was the more appropriate accelerator architecture used for the big data analysis of the large-scale service-demand matching in sharing economy. In our future work, we will obtain the trace file recording numerous data sets related to sharing economy. Next, we will develop the field-programmable gate array-based real platform to demonstrate the advantage of speedup and scalability using the real trace as input in optical data centers.

References

[1] A. Sundararajan, *The Sharing Economy—The End of Employment and the Rise of Crowd–Based Capitalism*. Cambridge, MA, USA: MIT Press, 2016.

[2] X. Hu, X. Li, E. Ngai, V. Leung, and P. Kruchten, "Multidimensional context-aware social network architecture for mobile crowdsensing," *IEEE Commun. Mag.*, vol. 52, no. 6, pp. 78–87, Jun. 2014.

[3] W. Hou, Z. Ning, and L. Guo, "Green survivable collaborative edge computing in smart cities," *IEEE Trans. Ind. Informat.,* vol. 14, no. 4, pp. 1594–1605, Apr. 2018.

[4] J. Bi *et al.*, "Application-aware dynamic fine-grained resource provisioning in a virtualized cloud data center," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 2, pp. 1172–1184, Apr. 2017.

[5] X. Hu *et al.*, "Emotion-aware cognitive system in multi-channel cognitive radio ad hoc networks," *IEEE Commun. Mag.*, vol. 56, no. 4, pp. 180–187, Apr. 2018.

[6] D. Helbing and E. Pournaras, "Society: Build digital democracy," *Nature*, vol. 527, no. 7576, pp. 33–35, 2015.

[7] Z. Ning, X. Wang, X. Kong, and W. Hou, "A social-aware group formation framework for information diffusion in narrow-band Internet of Things," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2017.2777480.

[8] X. Wang, Z. Ning, and L. Wang, "Offloading in Internet of vehicles: A fog-enabled real-time traffic management system," *IEEE Trans. Ind. Informat.*, to be published, doi: 10.1109/TII.2018.2816590.

[9] (2017). *Amazon Mechanical Turk*. [Online]. Available: https: www.mturk.com/mturk/welcome

[10] X. Hu *et al.*, "SAfeDJ: A crowd-cloud codesign approach to situation-aware music delivery for drivers," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 12, no. 1s, 2015, Art. no. 21.

[11] X. Li and Z. Cai, "Elastic resource provisioning for cloud workflow applications," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 2, pp. 1195–1210, Apr. 2017.

[12] X. Hu, T. H. S. Chu, V. C. M. Leung, E. C.-H. Ngai, P. Kruchten, and H. C. B. Chan, "A survey on mobile social networks: Applications, platforms, system architectures, and future research directions," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1557–1581, 3rd Quart., 2015.

[13] J. Jin, Y. Li, X. Zhong, and L. Zhai, "Why users contribute knowledge to online communities: An empirical study of an online social Q&A community," *Inf. Manage.*, vol. 52, no. 7, pp. 840–849, 2015.

[14] W. Hou, Z. Ning, L. Guo, Z. Chen, and M. S. Obaidat, "Novel framework of risk-aware virtual network embedding in optical data center networks," *IEEE Syst. J.*, to be published, doi: 10.1109/JSYST.2017.2673828.

[15] S. Sarkar, G. R. Kulkarni, P. P. Pande, and A. Kalyanaraman, "Network-on-chip hardware accelerators for biological sequence alignment," *IEEE Trans. Comput.*, vol. 59, no. 1, pp. 29–41, Jan. 2010.

[16] W. Hou, Z. Ning, L. Guo, and M. S. Obaidat, "Service degradability supported by forecasting system in optical data center networks," *IEEE Syst. J.*, to be published, doi: 10.1109/JSYST.2018.2821714.

[17] Y. Ye *et al.*, "System-level modeling and analysis of thermal effects in WDM-based optical networks-on-chip," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.,* vol. 33, no. 11, pp. 1718–1731, Nov. 2014.

[18] M. Nikdast *et al.*, "Crosstalk noise in WDM-based optical networks-on-chip: A formal study and comparison," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 11, pp. 2552–2565, Nov. 2014.

[19] M. Zeng, W. Fang, and Z. Zhu, "Orchestrating tree-type VNF forwarding graphs in inter-DC elastic optical networks," *J. Lightw. Technol.*, vol. 34, no. 14, pp. 3330–3341, Jul. 15, 2016.

[20] Q. Sun, X. Yang, K. Long, X. Yin, and Z. Li, "On vector linear solvability of multicast networks," *IEEE Trans. Commun.*, vol. 64, no. 12, pp. 5096–5107, Dec. 2016.

[21] W. Hou *et al.*, "On-chip hardware accelerator for automated diagnosis through human-machine interactions in healthcare delivery," *IEEE Trans. Autom. Sci. Eng.*, to be published, doi: 10.1109/TASE.2018.2832454.

[22] Z. Ning, W. Hou, X. Hu, and X. Gong, "A cloud-supported cps approach to control decision of process manufacturing: 3D ONoC," in *Proc. 13th IEEE Int. Conf. Autom. Sci. Eng.*, Aug. 2017, pp. 458–463.

[23] L. Bai, H. Gu, Y. Yang, and K. Wang, "A crosstalk aware routing algorithm for Benes ONoC," *IEICE Electron. Exp.*, vol. 9, no. 12, pp. 1069–1074, 2012.

[24] S. Koohi and S. Hessabi, "All-optical wavelength-routed architecture for a power-efficient network on chip," *IEEE Trans. Comput.*, vol. 63, no. 3, pp. 777–792, Mar. 2014.

[25] L. Guo, Y. Ge, W. Hou, P. Guo, Q. Cai, and J. Wu, "A novel IP-core mapping algorithm in reliable 3D optical network-on-chips," *Opt. Switching Netw.*, vol. 27, pp. 50–57, Jan. 2018.

Lei Guo (M'06) received the Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2006.

He is currently a Professor with Northeastern University, Shenyang, China. He has authored or co-authored more than 200 technical papers in international journals and conferences. His current research interests include communication networks, optical communications, and wireless communications.

Dr. Guo is currently serving as an Editor for several international journals.

Zhaolong Ning (M'14) received the M.S. and Ph.D. degrees from Northeastern University, Shenyang, China.

He is currently an Associate Professor with the School of Software, Dalian University of Technology, Dalian, China. His current research interests include social computing, big data, and fog computing. He has authored or co-authored around 70 technical papers in the above-mentioned areas in international journals and conferences.

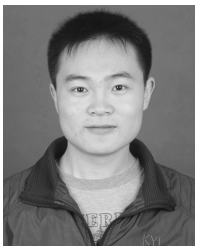
Dr. Ning was a Research Fellow with Kyushu University, Fukuoka, Japan.

Weigang Hou (M'13) received the M.E. degree in information and communication system and the Ph.D. degree in information and communication system from Northeastern University, Shenyang, China, in 2009 and 2013, respectively.

In 2012, he was an Associate Researcher with the Department of Computer Science, City University of Hong Kong, Hong Kong. He is currently an Associate Professor with the Smart Systems Lab, School of Computer Science and Engineering, Northeastern University, and the Deputy Director of the research office with the Technical Research Center of Science Popularization Informationization Engineering of China. He has authored or co-authored more than 120 publications. His current research interests include traffic grooming, optical network with data center, and optical network on chip. His research programs span the interest areas above.

Bin Hu (M'10–SM'15) is currently a Professor and the Dean of the School of Information Science and Engineering with Lanzhou University, Lanzhou, China, an Adjunct Professor with Tsinghua University, Beijing, China, and a Guest Professor with ETH Zurich, Zürich, Switzerland. He has authored or co-authored over 200 papers in peer-reviewed journals, conferences, and book chapters including *Science (Suppl.)*, the *Journal of Alzheimers Disease*, the IEEE TRANSACTIONS, the IEEE INTELLIGENT SYSTEMS, AAAI, BIBM, EMBS, CIKM, and ACM SIGIR. His work has been funded as a PI by the Ministry of Science and Technology, National Science Foundation China, European Framework Program 7, EPSRC, and HEFCE U.K.

Prof. Hu is a member at large of ACM China and an IET Fellow. He is the Co-Chair of the IEEE SMC TC on Cognitive Computing and the Vice President of the International Society for Social Neuroscience (China Committee). He has served as the Guest Editor for *Science* in a special issue on Advances in Computational Psychophysiology, an Associate Editor of the IEEE TRANSACTIONS ON AFFECTIVE COMPUTING, the IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, *Brain Informatics*, *IET Communications, Cluster Computing, Wireless Communications and Mobile Computing*, and *Wileys Security and Communication Networks*.



Pengxing Guo received the Bachelor's degree with the College of Engineering, Bohai University, Jinzhou, China, in 2014, and the Master's degree in communication and information systems from Northeastern University, Shenyang, China, in 2016, where he is currently pursuing the Ph.D. degree.

His current research interests include electronic/photonic network-on-chip, silicon-based optical interconnection chip design, and reliable networks design.