Multi-granularity Decomposition of Componentized Network Applications Based on Weighted Graph Clustering

Ziliang Wang[1], Fanqin Zhou[1,*], Lei Feng[1], Wenjing Li[1,*], Tingting Zhang[2], Sheng Wang[2] and Ying Li[2]

[1]*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, 100876, China*
[2]*China Mobile Research Institute, Beijing, 100053, China*
E-mail: fqzhou2012@bupt.edu.cn; wjli@bupt.edu.cn
[*]*Corresponding Author*

Received 29 September 2021; Accepted 16 December 2021; Publication 08 March 2022

Abstract

With the development of mobile communication and network technology, smart network applications are experiencing explosive growth. These applications may consume different types of resources extensively, thus calling for the resource contribution from multiple nodes available in probably different network domains to meet the service quality requirements. Task decomposition is to set the functional components in an application in several groups to form subtasks, which can then be processed in different nodes. This paper focuses on the models and methods that decompose network applications composed of interdependent components into subtasks in different granularity. The proposed model characterizes factors that have important effects on the decomposition, such as dependency level, expected traffic, bandwidth, transmission delay between components, as well as node resources required by the components, and a density peak clustering (DPC) -based decomposition algorithm is proposed to achieve the multi-granularity decomposition.

Journal of Web Engineering, Vol. 21_3, 815–844.
doi: 10.13052/jwe1540-9589.21312

Simulation results validate the effect of the proposed approach on reducing the expected execution delay and balancing the computing resource demands of subtasks.

Keywords: Componentized network application, weighted graph clustering, density peak clustering, multi-granularity task decomposition.

1 Introduction

With the development of mobile communication and network technology, the rapid increase in the number of intelligent user terminals and the proliferation of smart Internet services are making network applications increasingly resource hungry [1]. At the same time, smart devices, like mobile phones, workstations, etc. boast the ever-growing computing capability whose computing resources are usually underutilized, while multi-access edge computing resources are usually overloaded. To improve the capability of computing service provision ubiquitously in mobile networks, one of the key issues is how to effectively utilize the fragmented resources distributed in nodes, such as mobile terminals, IoT devices, in mobile networks. Task decomposition aims to divide the entire application into different subtasks that can be processed independently in different nodes, which grants a new dimension of freedom to enable more flexible utilization of computing resources in the network.

With the increasing demand on the rapid development and flexible iteration of applications, the componentized application architecture gets prevailing, such as microservice, serverless. In this context, task decomposition is going to aggregate functional components in the entire application into different clusters to form subtasks with the lowest expected overhead when deployed and executed in a distributed manner. Task decomposition can facilitate the use of fragmented computing resources in mobile networks, to achieve a higher degree of freedom in resource allocation, and to facilitate load balancing among nodes and network links. The effective decomposition of tasks can help to achieve the distributed execution of tasks, thus reducing the overall running time and improving the utilization of fragmented resources in the network. Thus, task decomposition is of great significance [2].

The existing research of software-related task decomposition is mainly based on the functional structure of codes, and different functional parts are implemented in functions, forming a prerequisite for componentized application architecture. The existing approaches for task decomposition of componentized applications mainly include the ones based on design structure matrix, hierarchical task network, and graph theory [3]. However, these approaches usually focus on generating merely one decomposition scheme, which is not flexible enough to adapt to the heterogeneous computing resources in cloud-edge-terminal multi-layers of present networks. There are also some multi-granularity decomposition approaches in other fields, such as social relation discovery, image segmentation. These approaches are usually based on the relations between objects and cannot be directly applied to the scenarios where the properties of objects more than the relations between objects need consideration, as in the case of multi-granularity task decomposition of componentized applications.

This paper proposes a multi-granularity task decomposition method based on the vertex-weight improved density peak clustering (DPC) algorithm. According to the relationship between functional components, the task is reasonably decomposed into multi-granularity subtasks, which provides a reasonable choice for the scheduler to allocate the appropriate subtasks to the appropriate nodes in the mobile edge network. The main contributions of this paper are as follows. (1) The model is proposed to characterize the factors influencing task decomposition of component-based network applications. The model is mainly based on the relationship between functional components, such as the total flow of data exchange, bandwidth requirements, data dependence, etc., to meet the special needs. (2) A weighted graph clustering algorithm is proposed to decompose network applications into multiple subtasks with different granularity. Each set will represent a low latency and dependency decomposition scheme with good performance.

The rest of the paper is organized as follows. Section 2 describes the related work. Section 3 proposes the system model and some main concepts. Section 4 presents the proposed approach of task decomposition for componentized network applications. Evaluation scheme and numeric results are given in Section 5, and Section 6 concludes the paper.

2 Related Work

2.1 Task Decomposition

Task decomposition has the potential to facilitate task distributed scheduling in networks. In recent studies, task decomposition mainly serves as an early step to split a whole task and offload a part of the task onto the edge computing node or centre cloud for offloading. In these studies, C. Xia et al. in [4] adopted quotient space theory and proposed a hierarchical task decomposition algorithm for data grid resource scheduling. X. Liu et al. in [5] proposed to cluster and schedule IoT terminals according to the priority of computing tasks of service. Among them, the highest priority task is executed in the edge computing node, and the lowest priority task is executed locally in the terminal. Other tasks are real-time scheduled based on the task amount, available resources, and other factors in the environment. H. Li et al. in [6] studied the joint optimization of user computing task offloading and computing and communication resource allocation under the scenarios of multi-user and multi-edge computing nodes, aiming at minimizing the energy consumption of user equipment. A two-stage heuristic optimization algorithm based on a genetic algorithm is designed to solve the problem. S. E. Mahmoodi et al. in [7] investigated computing tasks offloading from mobile terminals to edge and centre cloud. The constraints of computing tasks and data offloading, such as communication delay, application execution time, and component priority, are comprehensively considered. Though componentized application architecture was assumed in this paper, the decomposition of componentized applications was not specially discussed. M. Z. Liu et al. in [8] proposed a cloud manufacturing task decomposition algorithm based on sequential task decomposition, gave the relevant definitions of cloud manufacturing task description model and task constraint structure, and studied the manufacturing task granularity analysis method, manufacturing task cohesion measurement method and manufacturing task correlation measurement method. Mohamad Roshanzamir in [9] tried to use GNP algorithm to save the experience of promising individuals in successive generations to form a task decomposition scheme. G. Latif et al. in [10] proposed a simplified routing mechanism along with the formulation and analyses of a nonlinearly constrained multi-objective optimization model for multi-user traffic engineering in networking and digital communication domains.

The above research mainly focused on cloud/edge computing task offloading. However, with the complexity of application growing, the component-dependency graph would be too complex for guiding the deployment of functional components distributed in the networks. There are emerging constraints, such as communication and computing resources demanded, and satisfaction of special needs between task components of applications, which should be taken care of when performing task decomposition. If the tasks are unproperly decomposed, the resource overhead and the performance of the deployed applications will be greatly affected. Therefore, it is necessary to build a new optimization model to decompose and allocate the tasks and resources of component-based network applications.

In addition, with the improvement of the computing and storage capability of the mobile terminals and the development of D2D communication technology, the available resources of the mass terminals in the mobile edge network have also become an important direction for the task offloading of network applications [11, 12]. However, the total resource of a single terminal is still insufficient compared to the resource requirement of some network applications. In the meanwhile, the resource capacity of different nodes is also different in the mobile edge networks owning to the heterogeneity of the nodes. Thus, a multi-level and multi-granularity decomposition scheme is needed to make the decomposition adaptive to specific resource conditions in the mobile edge networks.

2.2 Graph Clustering Algorithm

Clustering is an unsupervised learning method of pattern recognition [13], which means separating samples into several clusters according to the characteristics of the data samples. The similarity of the samples within each cluster is very high, and the similarity of different samples is very low. For various types of data, there are many clustering algorithms [14], which can be roughly divided into five categories: Partition-based clustering, density-based clustering, and grid-based clustering, partition-based clustering, and model-based clustering methods. In the past few years, density-based clustering algorithms have been widely used in many scientific fields because of their simplicity and ability to detect clusters of different sizes and shapes. Typical density-based clustering algorithms, like DBSCAN [15], optics [16], DBCLASD [17], and DENCLUE [18] have attracted much attention. The density peak clustering (DPC) algorithm is the most recently proposed one [19], which can automatically discover cluster centres and achieve efficient clustering of arbitrary shape data.

In the related research of improved DPC algorithms, a self-adaptive distance density peak clustering Algorithm (DADPC) was proposed in [20]. In the paper, a new distance measure, namely density adaptive distance, and the concept of the weight factor of distance mediation, are introduced, which consists of local and global density adaptive distance, the local density and the shortest distance of each point are calculated by the classical Floyd algorithm based on density adaptive distance, and the decision graph is drawn to select the cluster centre. In 2019, Han X. et al. proposed a new hybrid algorithm by improving the existing Chameleon clustering algorithm and combining the Chameleon algorithm with the K-medoids algorithm [21]. This method eliminates the first-stage clustering of the Chameleon algorithm and uses the K-medoids algorithm to divide original data and uses the second-stage clustering algorithm of the Chameleon algorithm to merge the clusters. The idea of a relative degree of similarity, the relative degree of association, and the relative degree of similarity are introduced in this method, which provides improved forms for the clustering algorithm. D.S. Sun in [22] proposed an efficient graph clustering algorithm based on spectral coarsening to deal with the large time complexity of the traditional spectral algorithm. H. H. Zhou et al. in [23] proposed a density peak clustering algorithm combining shared nearest neighbour and shared inverse nearest neighbour. The algorithm uses the shared nearest neighbour and shared inverse nearest neighbour of samples to construct a new similarity calculation method; Then, the local density formula is redefined to avoid the selection of truncation distance; Finally, the sample point allocation algorithm assigns the remaining sample points to the corresponding clusters. Z.W. Gu in [24] proposed a method for automatically selecting cluster centres based on Chebyshev's inequality. MG-DPC is implemented on the dataset of load data to realize load classification. However, the adjustment of truncation distance is not fully considered in the algorithm of cluster decomposition. At the same time, only the selection of cluster centre point is considered in the result of task decomposition, which is a lack of constraints on the scale of multi-granularity task decomposition.

Inspired by the aforementioned research, in this paper, we improve the DPC algorithm for task decomposition of functional componentized applications and propose the weighted multi-granularity decomposition (WMG-DCom) approach to produce multiple decomposition schemes in different granularities.

3 System Model

3.1 Componentized Network Applications

In this paper, task decomposition is to divide a large number of functional components of network application into multiple parts, which we refer to

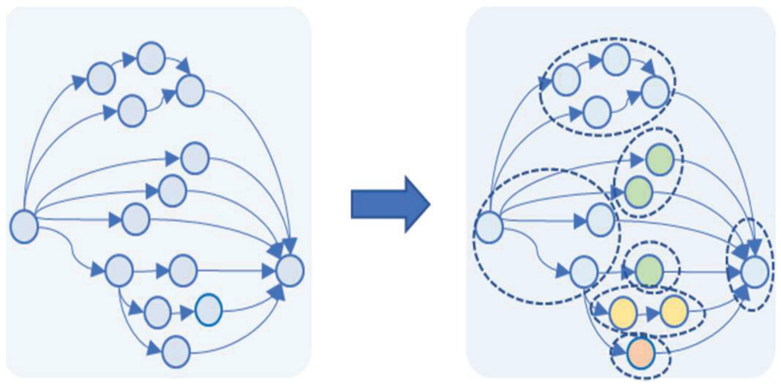


Figure 1 Task decomposition for componentized network applications.

as subtasks and can be executed in different computing nodes. Figure 1 illustrates the task decomposition of a componentized application. Then, the set of functional components in each subtask will be deployed in different network nodes as a whole. In this way, the granularity of resource allocation is in the granularity of subtasks instead of the task component granularity, and the complexity of computing resource scheduling can be reduced.

From Figure 1, we can also see the problem of task decomposition for a componentized network application resemble the graph clustering problem, so graph clustering algorithms can be used to divide the network application into subtasks. In the meanwhile, the multi-granularity task decomposition means the proposed method will decompose an application into different granularity levels. As shown in Figure 2(a), the coarse-grained level means that the subtask specifications obtained by task decomposition are relatively large, but the number of subtasks is relatively small, while the fine-grained level means that the subtask specifications obtained are relatively small, but the number of subtasks is large.

However, in the existing research, multi-granularity decomposition is only carried out for different quantities of granularity, and the granularity of decomposition is relatively average. However, the granularity requirements in different resource environments are not fully considered in a single scheme, and there are constraints. As shown in Figure 2(b), when there is a cloud-edge-terminal collaborative operation environment, the network tasks with smaller granularity can be assigned to t edge and terminal for operation, while the tasks with larger granularity can be carried out in the cloud. Making full use of resources can effectively improve the operation efficiency of tasks.

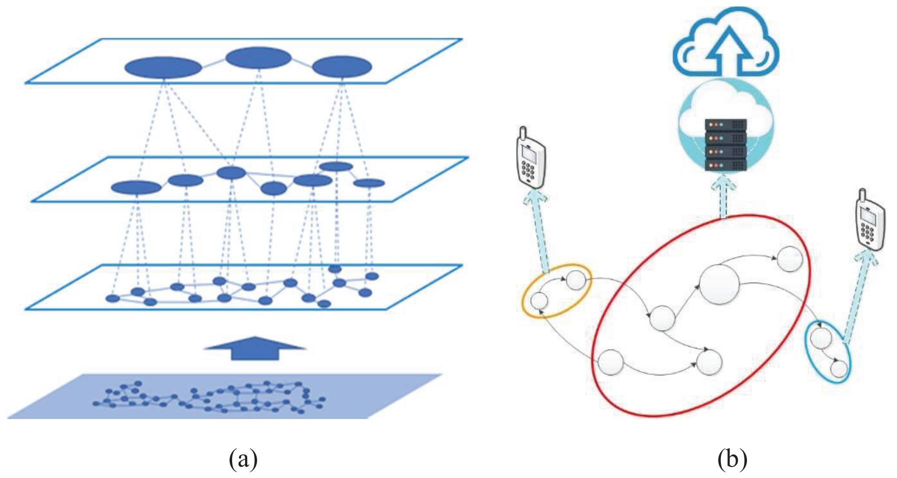


Figure 2 Multi-granularity task decomposition.

3.2 Constraints of Task Decomposition

In the process of task decomposition, the dependency between the components and business requirements constraints should be considered comprehensively. In the following part, we will introduce and discuss the factors to be considered in the task decomposition, which are roughly set into three types, i.e., node-specific, relation-specific, and business-specific factors.

(1) Node-specific factors

Computing resources: Different types of components have different requirements on the computing capability of server nodes in networks. The components or subtasks with higher requirements on computing capability should be deployed to server nodes with sufficient computing resources.

Storage resources: Different types of components also have different requirements on the storage capacity in server nodes in networks. The components or subtasks with higher requirements on storage should be deployed to server nodes with sufficient storage resources.

Processing delay: When the data is processed in a node, there is a processing delay. The higher the calculation of two functional components is, the more likely the two functional components will be placed in different subtasks, when deployed in the edge or terminal nodes. For example, in VR games, in the process of switching between two different scenes, music and videos should be processed in different subtasks at the same time as far as possible, to reduce the processing delay caused by the volume of computational data and avoid long waiting time. Therefore, the processing delay should also be considered in the clustering decomposition of functional components.

Task decomposition for applications to be deployed in mobile networks with many cooperative nodes can take finer granularity to enable the deployment in these nodes. Otherwise, the application can be deployed in a very limited number of nodes, so task decomposition can take coarser granularity to facilitate the rapid deploy the tasks.

(2) Relation-specific factors

Data volume for exchange: The interaction between two components of an application will produce some amount of communication data, which demands communication performance. The higher the total amount of data is, the closer the relationship between the two components will be, and the two components are more likely to be in the same subtask. For example, in the intelligent security business, the video capture function component and the function component of pre-processing the original video will produce a large amount of data for exchange, and these two function components should be executed together.

Communication bandwidth: For a digital communication system, generally, the larger the bandwidth provided by the system, the better the real-time performance of its service. There are some tasks with low delay and fast response. For example, MEC is used to realize the purpose of Internet of things. Basically, IoT devices are connected through different communication protocols through various wireless technologies (such as 3G, LTE, WiFi, etc.). Therefore, in the process of information transmission, the bandwidth requirements are relatively large. Generally, the task elements with large bandwidth requirements will be placed in the same subtask.

Communication delay: The higher the communication delay is, the more likely two functional components will be to be classified into the same subtask. For example, on the Internet of Vehicles, vehicles and roadside devices are connected through various wireless technologies. Low latency is a prerequisite for completing various protocol processing, message distribution, and message processing. Roadside applications need to receive local information directly from applications on vehicles and roadside sensors, Analyse them and broadcast warnings (such as accident information, etc.) to nearby vehicles with very low delay. In particular, the communication delay between these functional components needs to be considered.

Data dependency: The application of different components may read and write a unified data block, and divide the functional components reading, and writing the same data block into the same subtask, which can reduce the repeated data transmission in the physical network and reduce the overall bandwidth cost of the task. For example, in the intelligent security scene, for the video stream provided by the public camera, the video processing unit of the public security department will focus on the people in the video, while the video processing unit of the traffic police department will focus on the vehicles. If the two functional units are executed independently on different nodes, they need to transmit two identical video streams to the two departments. If these two functional units can be classified into the same subtask and executed in the same node, the video stream data can be reused to avoid the network transmission of two identical data.

Special needs: In most cases, users would not like their sensitive data and critical information collected by terminals and uploaded to the data centre but want them processed in a timely and efficient manner at the edge nodes, such as computing, encryption, and access control. We call these special requirements as special needs. And for such cases, the satisfaction of special needs should be considered.

(3) Business-specific factors

Business-specific factors are originated from the service requirement of the application. A typical factor is service delay, which measures the expected time consumed when a request of an application gets served. The service delay includes processing delay in those functional components and the communication delay between components for data delivery. Different services have different delay tolerance, and the produced decomposition scheme should satisfy the delay tolerance of the application. Thus, the service delay can be used as a metric to evaluate the effectiveness of the decomposition scheme.

3.3 Definitions Used in Algorithm Design

The density peak clustering algorithm [25] is based on two basic assumptions: (1) The local density of the cluster centre (the peak of density) is greater than that of its neighbours. (2) The distance between the different cluster centres is relatively far. To find the cluster centre which satisfies these two conditions, the definition of local density is introduced.

Assume a task has J task elements, and task decomposition is to be performed on it to get several subtasks. Some key variables are defined as follows.

(1) Correlation coefficient

The correlation coefficient T_{ij} between two functional components (i.e., task elements) X_i and X_j, and $i \neq j \in \{1, \ldots, J\}$, is defined as in Equation (1) taking account of the normalized data volume for exchange D_{ij}, data dependence R_{ij}, communication bandwidth demand S_{ij}, and special needs λ_{ij} between functional components X_i and X_j.

$$T_{ij} = \frac{P_1 D_{ij} + P_2 R_{\mathrm{ij}} + P_3 S_{\mathrm{ij}}}{\lambda_{ij}}, \tag{1}$$

Where P_1, P_2, P_3 are the proportion of data volume for exchange, data dependence, and communication bandwidth demand. It should be noted that all the parameters D_{ij}, R_{ij}, and S_{ij} are normalized to the range of $[0, 1]$, while λ_{ij} is normalized to $(0, 1]$ and the stronger the special needs are, the smaller the value of λ_{ij} will be. If two elements have no interaction, the values of D_{ij}, R_{ij}, and S_{ij} will be 0, while λ_{ij} equals 1. We assume there are N pairs of task elements that interact with each other. Thus, N equals the number of elements in $\{D_{ij} | D_{ij} \neq 0, i \neq j \in \{1, \ldots, J\}\}$.

(2) Weight coefficient

The weight coefficient is taken as the reciprocal of the amount of calculation C_i and the average amount of calculation is normalized to get the weight coefficient.

$$W_i = \frac{\overline{C}}{C_i}, \tag{2}$$

Where $\overline{C}$ is the average amount of calculation of all task elements, and C_i is the calculation of task element X_i, Weight coefficient reflects the effects of components' properties on decomposition. Intuitively, the smaller the amount of calculation of task elements, the easier it is to form clusters with other task elements.

(3) Distance mapping

The factors affecting task decomposition include the calculation of task elements, as well as the correlation coefficient between task elements. This algorithm considers that the weight of task element points affects the weight of the edge. In order to effectively decompose tasks and form reasonable clusters, we convert the weight coefficient of task elements and correlation coefficient between task elements into the distance relationship d_{ij} between task elements X_i and X_j.

$$d_{ij} = \frac{W_i W_j}{{T_{ij}}^2} \tag{3}$$

(4) Local density

The local density of task element X_i is

$$\rho_i = \sum_j \exp\left(-\frac{d_{ij}^2}{d_c^2}\right), \tag{4}$$

Where d_{ij} is the distance between task element X_i and X_j while d_c is the cut-off distance and is usually set manually. The value of d_c impacts the effect of the clustering algorithm. It can't be too big or too small. If d_c is too large, the local density of each data point will be very large, resulting in low discrimination. If d_c is too small, the density properties of each data point will be hard to be accurately identified, resulting in improper cluster results. Empirically, the cut-off distance is set as the minimum.

(5) The minimal relative distance

The minimal relative distance refers to the distance between the task element X_i and the nearest task element X_j whose local density is greater than that of X_i. That is to say, X_j is the closest point with larger density compared with X_i. The relative distance δ_i can be represented as

$$\delta_i = \min_{j:\rho_j > \rho_i} (d_{ij}). \tag{5}$$

If task element X_{i^*} has the largest local density among all task elements, the minimal relative distance δ_i of X_{i^*} is defined as the distance from X_{i^*} to the farthest point, which can be expressed as Equation (4).

$$\delta_i = \max_j (d_{i^* j}). \tag{6}$$

That is to say,

$$\delta_i = \begin{cases} \min\limits_{j:\rho_j > \rho_i} (d_{ij}), & i \neq i^* \\ \max\limits_j (d_{ij}), & i = i^* \end{cases}$$

Let H denote the set $\{(X_j, X_i)|\forall i \in J/\{i^*\}, j = \text{argmin}_{j:\rho_j>\rho_i}(d_{ij})\}$. As there are J tasks elements, there are $J-1$ tuples in the set H.

(6) Gini coefficient

The Gini coefficient of the data set G can be defined as

$$G = 1 - \sum_{i=1}^{n} \left(\frac{\gamma_i}{Z}\right)^2, \tag{7}$$

Where $\gamma_i = \rho_i\delta_i$, and Z is calculated as the summation of γ_i, i.e., $Z = \sum_{i=1}^{n}(\gamma_i)$. Gini coefficient can be used to characterize data impurity. The less impure the data is, the less the uncertainty of the data distribution will be, and the more like the task elements are set in the proper clusters. In the adaptive DPC algorithm, an adaptive method based on the minimization of the Gini coefficient is proposed to adaptively select the cut-off distance.

4 Multi-Granularity Task Decomposition Approach

This section presents an improved DPC algorithm by enhancing the accuracy of clustering through self-adaptation and similarity calculation and then introduces the multi-granularity task decomposition algorithm based on the improved DPC algorithm.

4.1 An Improved Density Peak Clustering Algorithm with Adaptive Cut-off Distance

In the improved DPC algorithm, an adaptive mechanism for cut-off distance selection based on minimizing the Gini coefficient is introduced. The Gini coefficient is used to measure the data impurity.

In the selection of optimized cut-off distance, d_c^* corresponds to the minimum value of the Gini coefficient. The optimized cut-off distance is used as the parameter for the next iteration of clustering process, instead of a fixed cut-off distance. In this way, the improved DPC can avoid subjectivity in the manual selection of cut-off distance and achieve the goal of self-adaptation.

The specific steps are as follows:

Step 1: According to the data volume for exchange, data dependency, communication bandwidth, and special needs, calculate the correlation coefficient T_{ij} between task elements X_i and X_j, $i \neq j \in \{1, \ldots, J\}$.

Step 2: Get the weight coefficient W_i according to Equation (2) with the calculation C_i.
Step 3: The distances of N task-element pairs are calculated according to Equation (3), and are sorted in ascending order. The ordered sequence of distances is denoted as ord_D.
Step 4: Take the minimum distance as the cut-off distance, and calculate the local density and the minimal relative distance of each task element.
Step 5: Get the initial Gini coefficient G according to Equation (7) with a default d_c.
Step 6: Increase d_c to obtain a new Gini coefficient G' and the corresponding sequences of $\{\gamma_i\}$ according to $\gamma_i = \rho_i\delta_i$.
Step 7: Take the scheme under the minimum value of Gini coefficient G^* to obtain the appropriate cut-off distance d_c^*.
Step 8: Get the local density ρ_i and the minimal relative distance δ_i when the cut-off distance is d_c^*.
Step 9: Get set H according to the local density ρ_i and the minimal relative distance δ_i.
Step 10: Select N_q task elements to form the set $Q = \{X_s\}$ corresponding to the N_q largest elements in $\{\gamma_i\}$ as the cluster centres.
Step 11: For each $X_s \in Q$, get the set of tuples $K = \{(X_j, X_i) \in H | X_j = X_s, X_i \notin Q\}$, and all the second elements of the tuples in K form a set Γ. Iteratively pop out an element X_t from Γ, and get all the tuples $(X_j, X_i) \in \{H/K, X_j = X_t, X_i \notin Q\}$, push all the X_i into Γ, repeat until Γ is empty. Put all the manipulated X_i, X_j in Set Φ_{X_s} for each iteration.
Step 12: Get clustering results $\varphi = \{\Phi_{X_s}, X_s \in Q\}$.

The improved DPC algorithm is summarized in Algorithm 1.

4.2 Proposed multi-granularity task decomposition algorithm

The constraint relationship of each task element is mapped into a distance relation in space. The local density ρ and relative distance δ are calculated by the distance relationship between each task element. Then use $\gamma_i = \rho_i\delta_i$ to select the sampling point of the task element. The centre sets and the centre cluster of the task elements are determined by Algorithm 1. Due to the different number of computing nodes in the existing network environment, each subtask calculated at the edge node must meet the carrying capacity of the node. We assume that the carrying capacity of each edge node is C_{max}. Therefore, in the task decomposition, we should consider

Algorithm 1 The improved DPC algorithm

```
Input: $C_i, D_{ij}, R_{\mathrm{ij}}, S_{\mathrm{ij}}$, and $\lambda_{ij},\ i, j \in \{1, \ldots, J\}, N_q$
Output: $\varphi$
1:  Calculate $T_{ij}$ and $N$ according to Equation (1)
2:  Calculate $W_i$ according to Equation (2) and $d_{ij}$ according to Equation (3)
3:  $i = 1; G = 1;$
4:  while $i = N$                    \\Calculate the minimum value of Gini coefficient $G^*$
5:      Determine $d_c = ord_D(i)$;
6:      Calculate $\rho_i$ and $\delta_i$ according to Equations (4) and (5);
7:      Calculate Gini coefficients $G'$;
8:      if $G' < G$ then
9:          $G = G'; d_c^* = d_c;$
10:     end if
11:     $i = i + 1;$
12: end while
13: find the minimal Gini coefficient $G^* = G$;
14: find the cut-off distance $d_c{}^*$ with Gini coefficient $G^*$;
15: Calculate $\rho_i$ and $\delta_i$
16: Calculate $\gamma_i$ and $H$;
17: Find the $N_q$ largest elements in $\{\gamma_i\}$ to form the set $Q = \{X_s\}$;
18: for $X_s \in Q$
19:     $K = \{(X_j, X_i) \in H | X_j = X_s,\ X_i \notin Q\}$;
20:     $\Gamma = \{X_i | (X_s, X_i) \in K\}$;
21:     while $\Gamma \neq \emptyset$
22:         pop out $X_t \in \Gamma$,
23:         get all the tuples $(X_j, X_i) \in \{H/K, X_j = X_t, X_i \notin Q\}$ put all $X_i$ in $\Gamma$
24:         put $X_s$, $X_t$, $X_i$ in set $\Phi_{X_s}$
25:     end while
26:     $\varphi = \{\Phi_{X_s}, X_s \in Q\}$;
27: end for
        return $\varphi$.
```

the multi-granularity task decomposition algorithm suitable for the network environment which will be described in detail next.

The specific steps are as follows:

Step 1: According to the total data amount in interaction, data dependence, communication demand and special needs, calculate the correlation coefficient T between task elements.

Step 2: Get the weight coefficient W_i according to Equation (2) with the calculation C_i.

Step 3: Calculate the distances d_{ij} of N pairs of task elements according to Equation (3).

Algorithm 2 Weighted Multi-granularity decomposition

```
Input: $C_i, D_{ij}, R_{\text{ij}}, S_{\text{ij}}$, and $\lambda_{ij},\ i,j \in \{1,\ldots,J\}, N_q, C_{max}$
Output: $\varphi$
1:  Calculate $T_{ij}$ and $N$ according to Equation (1);
2:  Calculate $W_i$ according to Equation (2)
3:  Calculate $d_{ij}$ according to Equation (3);
4:  Find $d_c{}^*$ with Gini coefficient $G^*$ according to Algorithm 1;
5:  Calculate $\rho_i$ and $\delta_i$;
6:  Calculate $\gamma_i$ and $H$;
7:  Find $N_q$ largest elements in $\{\gamma_i\}$ to form the set $Q = \{X_1,\ldots,X_s,\ldots,X_{N_q}\}$;
8:  get $\varphi = \{\Phi_{X_s}, X_s \in Q\}$ by Algorithm 1, and $\varphi' = \emptyset$
9:  for $\Phi_{X_s} \in \varphi$
10:     $\tau_2 \leftarrow \{X_i \in \Phi_{X_s} | (X_s, X_i) \in H\}$
11:     while $\tau_2 \neq \emptyset$ and $C_{\varepsilon_{X_s}} \leq C_{max}$
12:         $\tau_1 \leftarrow \tau_2, \tau_2 \leftarrow \emptyset$
13:         while $\tau_1 \neq \emptyset$ and $C_{\varepsilon_{X_s}} \leq C_{max}$
14:             find $X_i = \text{argmin}_{X_i \in \tau_1}\{C_{X_i}\}$
15:             if $C_{\varepsilon_{X_s}} + C_{X_i} \leq C_{max}$
16:                 $C_{\varepsilon_{X_s}} += C_{X_i}$
17:                 $\varepsilon_{X_s} \leftarrow X_i$
18:                 $\tau_2 \leftarrow \tau_2 \cup \{X_t \in \Phi_{X_s} | (X_i, X_t) \in H\}$
19:                 $\varepsilon_{X_s} \leftarrow X_i$
20:                 Delete $X_i$ from $\tau_1$
21:                 $\varepsilon_0 \leftarrow \Phi_{X_s}/\varepsilon_{X_s}, \varphi' \leftarrow \varphi' \cup \{\varepsilon_{X_s}\}$
22:                 $\varphi' \leftarrow \varphi' \cup \{\varepsilon_0\}$;          $\backslash\backslash \varphi' = \{\varepsilon_0, \varepsilon_{X_s} | X_s \in Q\}$
23:             end if
24:         end while
25:     end while
26: end for
        return $\varphi'$.
```

Step 4: Calculate ρ_i, δ_i at the appropriate cut-off distance $d_c{}^*$ according to Algorithm 1.
Step 5: Get set H according to the local density ρ_i and the minimal relative distance δ_i.
Step 6: Select N_q task elements to form the set $Q = \{X_1,\ldots, X_s,\ldots,X_{N_q}\}$ corresponding to the N_q largest elements in $\{\gamma_i\}$ as the cluster centres, where N_q is the number of expected edge nodes.
Step 7: For each $X_s \in Q$, put X_s in set ε_{X_s}, get the set of tuples $K = \{(X_j, X_i) \in H | X_j = X_s, X_i \notin Q\}$, and all the second elements of the tuples in K form a set Γ. Iteratively pop out an element X_t from Γ, and get all the tuples $(X_j, X_i) \in \{H/K, X_j = X_t, X_i \notin Q\}$, push all the

X_j into Γ, repeat until Γ is empty. Put all the manipulated X_i, X_j in Set Φ_{X_s}. Get $\varphi = \{\Phi_{X_s}, X_s \in Q\}$ and let $\varphi' = \emptyset$

Step 8: For each subtask $\Phi_{X_s} \in \varphi$, decide whether all the task elements in the subtask satisfy the computing resource limit, and select a proper subset of task elements from Φ_{X_s} to form ε_{X_s} and other task elements are grouped to set ε_0 which represents a special subtask that will be processed in the central cloud. To select the proper subset of tasks, we iteratively check whether adding a task element $X_i \in \Phi_{X_s}$ will break the resource limit. ε_{X_s} will accept X_i if not break the resource limit, and we delete the task element X_i from Φ_{X_s}; otherwise, the iteration with regard to Φ_{X_s} is finished. If Φ_{X_s} is not empty, put $X_p \in \Phi_{X_s}/\varepsilon_{X_s}$ into set ε_0, and put ε_{X_s} into the set φ'.

Step 9: Get clustering results φ', which is in fact the set $\{\varepsilon_0, \varepsilon_{X_s}| X_s \in Q\}$.

The overall procedure of the proposed multi-granularity task decomposition algorithm is summarized in Algorithm 2.

From the above process details, we can see that the proposed WMG-Dcom borrows the concepts of local density and correlation distance in IDPC-Dcom to select the task centre point and cluster, while in the clustering process, WMG-Dcom additionally considers the computing resource consumption of subtasks and selects appropriate small-scale clusters to meet the resource utilization of computing nodes.

5 Simulation and Comparison

5.1 Evaluation Scheme Description

The algorithm proposed in this paper is evaluated in the cases where a series of CDGs with 9 components whose parameters, such as normalized total traffic of exchanged data, data dependency, bandwidth requirements, special needs between components, and the calculation of each task elements are generated in predefined reasonable ranges randomly for each simulation. Table 1 shows the range of each parameter in the experiment. Shown in Table 2 is an instance of the 9-elements case. An expected overall cost indicator that averaged the four factors is used to measure the general performance. We assume that there are 1–3 edge nodes in the existing network environment, that is, the task can be divided into 2–4 subtasks, which are placed in the edge node and cloud space respectively for computing.

Table 1 Key parameters and the values in the simulation

Parameter	Range
D	[0,1]
R	[0,1]
S	[0,1]
λ	(0,1]
C_i	(0,5000]
N_q	[0,n]
C_{max}	3000

In order to verify the effectiveness of the algorithm, this paper takes the IDPC-Dcom algorithm as the comparison scheme and compares all task elements fully decomposition (Fully Dcom) approach in which each component is treated as a subtask, as the baseline. These approaches are evaluated against the chosen metrics, such as the total volume of data, data dependency, bandwidth requirement, the satisfaction of special requirements, transmission waiting time, and the maximum size of computational data of subtasks. In this case, the decomposition scheme with a lower expected total cost has better performance.

In the sequel, an exemplary 9-element case is used as an illustration of task decomposition for componentized network applications.

5.2 Illustration of the Proposed Task Decomposition Approach

The illustrative procedures are given as follows.

Step 1: According to the relationship between task elements, the positions of functional components in the CDG of the whole task (application) are exported, as shown in Figure 3.

Step 2: Form the decision diagram by deriving the appropriate cut-off distance through the adaptive constraint distance d_c, get the constraint density and the minimal relative distance.

Step 3: The number of clusters is 2, the data centre is selected automatically according to ord_γ, and a proper cluster centre is verified by the graph clustering algorithm.

The decomposition schemes with 2 subtasks are shown in Figure 4.

The limitation of the IDPC-Dcom algorithm is that it can only divide equally according to the distance relationship, which may lead to that the divided subtasks can not meet the resource carrying capacity of the edge node

Table 2 The data of an exemplary 9-element application

(a) Data of individual functional components

Ca	The Amount of Calculation	Weight Coefficient
1	1000	0.476190476
2	900	0.428571429
3	1100	0.523809524
4	2000	0.952380952
5	5000	2.380952381
6	1400	0.666666667
7	2500	1.19047619
8	3000	1.428571429
9	2000	0.952380952

(b) Data of relation between two components

Ca	Cb	Total Volume of Data	Data Dependency	Bandwidth Requirement	Satisfaction of Special Needs	Correlation Coefficient
1	2	0.1	0.025	0.25	0.4	0.75
1	3	0.133	0.025	0.3	0.5	0.917
1	4	0	0.0025	0	1	0.003
1	5	1	0	1	0.4	5
1	6	0	0.00025	0	1	0.00025
1	7	0	0.00025	0	1	0.00025
1	8	0	0.00025	0	1	0.00025
1	9	0	0.00025	0	1	0.00025
2	3	0	1	0	0.5	2
2	4	0.333	0	0.5	0.5	1.667
2	5	0	0.00025	0	1	0.00025
2	6	0	0.00025	0	1	0.00025
2	7	0	0.00025	0	1	0.00025
2	8	0	0.00025	0	1	0.00025
2	9	0	0.00025	0	1	0.00025
3	4	0.3	0	0.5	0.5	1.6
3	5	0	0.00025	0	1	0.00025
3	6	0	0.00025	0	1	0.00025
3	7	0	0.00025	0	1	0.00025
3	8	0	0.00025	0	1	0.00025
3	9	0	0.00025	0	1	0.00025
4	6	0.167	0.0025	0.25	0.5	0.838
4	7	0	0.00025	0	1	0.00025
4	8	0	0.00025	0	1	0.00025
4	9	0	0.00025	0	1	0.00025
5	6	0.233	0.0025	0.2	0.5	0.872
5	7	0.267	0.0025	0.2	0.5	0.938
5	8	0.2	0.0025	0.2	0.5	0.805
5	9	0	0.00025	0	1	0.00025
6	7	0	0.00025	0	1	0.00025
6	8	0	0.00025	0	1	0.00025
6	9	0.667	0.0025	1	0.1	16.692
7	8	0	0.875	0	0.5	1.75
7	9	0.3	0.0025	0.25	0.5	1.105
8	9	0.3	0.0025	0.25	0.5	1.005

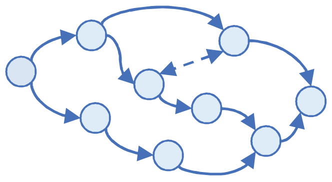


Figure 3 The CDG of the whole task.

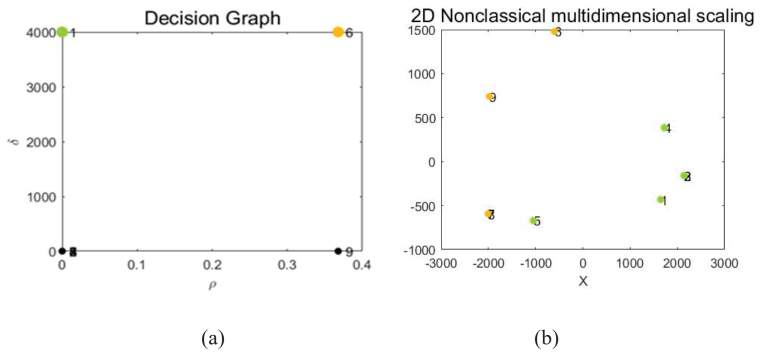


Figure 4 Tasks decomposition schemes with 2 subtasks, (a) The decision graph, (b) The result of task decomposition.

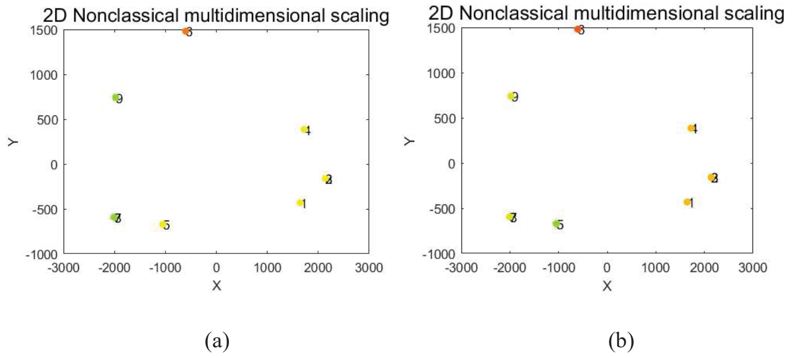


Figure 5 Tasks decomposition results in different granularity (a) 3 subtasks of manually selected task centre, (b) 4 subtasks of manually selected task centre.

so that they can not be calculated at the edge node. The result with 3 and 4 subtasks is shown in Figure 5.

In WMG-DCOM algorithm, we select an appropriate number of task centre points according to the resource carrying capacity and number of network edge nodes, and reasonably divide the subtasks according to the

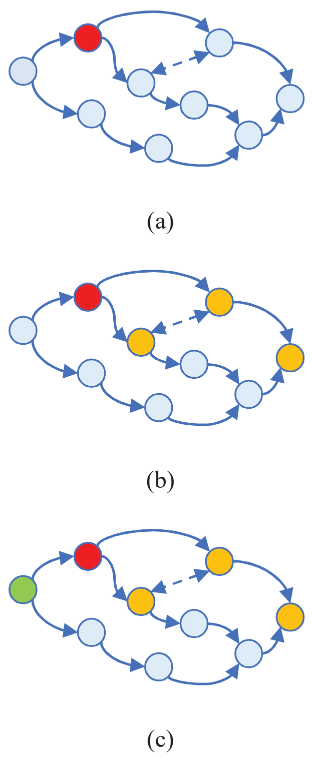


Figure 6 Tasks decomposition results by WMG-Dcom, (a) 2 subtasks (b) 3 subtasks (c) 4 subtasks.

calculation amount of task elements. The divided subtasks form a small-scale cluster, and the undivided task elements form a large-scale cluster. The results are as shown in Figure 6.

5.3 Numeric Results and Analyses on More Cases

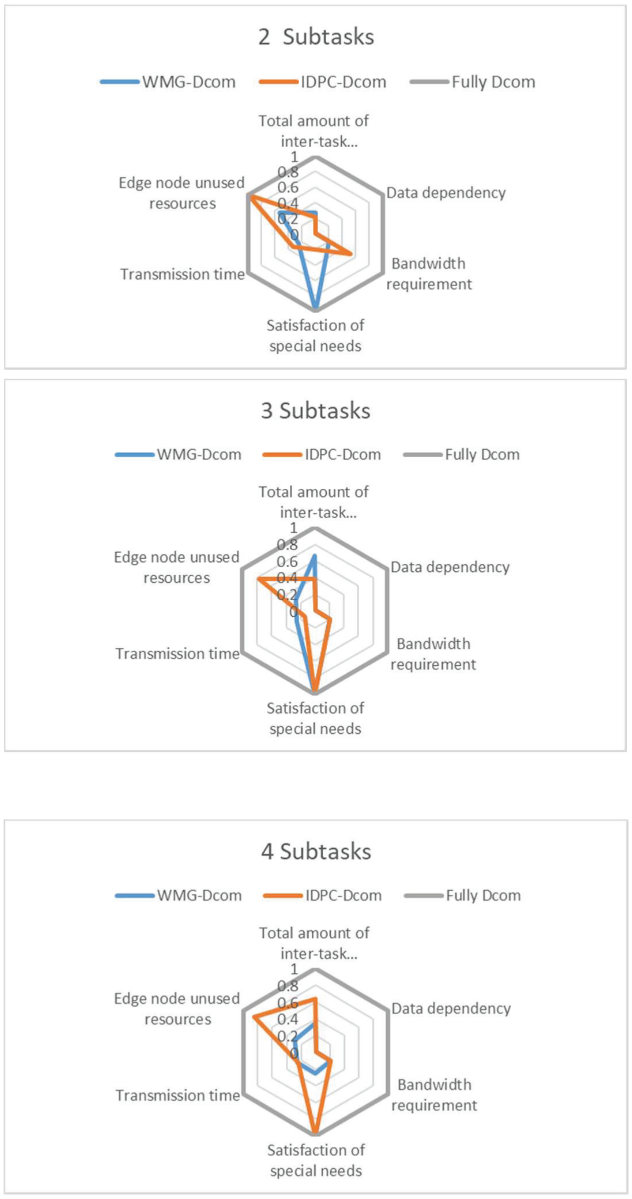
By comparing the total volume of data for exchange between subtasks, data dependency, bandwidth requirements, satisfaction of special needs, transmission time, and the resource utilization of edge nodes, the subtasks are clustered. The numerical results are in Table 3.

Through the table, we can find that WMG-Dcom algorithm has higher advantages than the IDPC-Dcom algorithm in clustering comprehensive data, and the effect of clustering optimization is more obvious. Figure 7 presents the comparison more intuitively.

Table 3 Comparison of data of clustering performance results

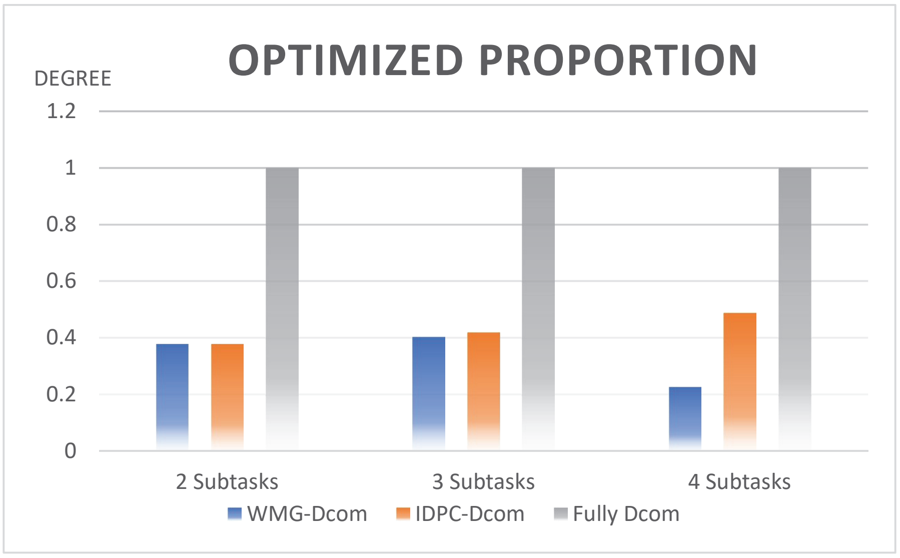
Approaches	Total Amount of Inter-task Communication	Data Dependency	Bandwidth Requirement	Satisfaction of Special Needs	Transmission Time	Edge Node Unused Resources	Optimized Proportion
			(a) 2 subtasks				
WMG-Dcom	0.2666666	0.0044866	0.20618556	1	0.244514107	0.533333	0.375864311
IDPC-Dcom	0.2166666	0.007178566	0.5154639	0.2	0.326018809	1	0.377554646
Fully Dcom	1	1	1	1	1	1	1
			(b) 3 subtasks				
WMG-Dcom	0.675	0.006794	0.206185567	1	0.253918495	0.26666666	0.401427454
IDPC-Dcom	0.3833333	0.008716831	0.206185567	1	0.144200627	0.766666666	0.418183832
Fully Dcom	1	1	1	1	1	1	1
			(c) 4 subtasks				
WMG-Dcom	0.35	0.007947699	0.206185567	0.25	0.2476489	0.288888889	0.225111842
IDPC-Dcom	0.63333333	0.010383284	0.206185567	1	0.238244514	0.844444444	0.48876519
Fully Dcom	1	1	1	1	1	1	1

**The data of ‘fully Dcom’ is normalized to 1 for a baseline of the numerical comparison.



(a)

Figure 7 Continued



(b)

Figure 7 Performance comparison (a) radar chart on individual indicators, (b) on a comprehensive indicator of sub-task optimization degree.

After taking the Fully Decomposed data as the standard for measurement and normalizing other parameters, it can be seen from the radar chart that the algorithm proposed in this paper has a good performance in the optimization of resource allocation and load balancing.

From the above results, we can see that the proposed WMG-Dcom approach in this paper is better than the IDPC-Dcom algorithm-based decomposition approach in terms of generating multi-granularity decomposition schemes, showing obvious advantages in the effectiveness of task decomposition in a specific network environment.

6 Conclusion

This paper investigates task decomposition for componentized network applications in mobile edge networks. It summarized the factors impacting task decomposition, and then proposed the weighted multi-granularity decomposition (WMG-Dcom) approach to decompose the task of the whole network application into multi-granularity subtasks. The schemes in multiple granularities are to gain task scheduler a new dimension of freedom to choose the proper granularity of decomposed tasks for resource scheduling to be adaptive to the diverse computing condition in mobile edge networks and reduce the complexity of task scheduling. Numeric results demonstrate the effectiveness of the proposed WMG-Dcom in deriving multi-granularity decomposition schemes with expected overall execution cost reduced in each granularity.

Acknowledgement

The work is funded by Beijing University of Posts and Telecommunications-China Mobile Research Institute Joint Innovation Centre.

References

[1] J.X. Liu, Z.H. Xia, "An approach of web service organization using Bayesian network learning", Journal of Web Engineering, Vol. 16, No. 3&4 (2017) 252–276

[2] X. Larrucea, I. Santamaria, C. Ebert, et al., "Microservices," IEEE Software, vol. 35, pp. 96–100, June 2018.

[3] S.P. Yi, Z.Z. Tan, Z.L. Guo, P.H. Wen, J. Zhou, et al., "Optimization of manufacturing task decomposition mode in cloud manufacturing service platform," Computer integrated manufacturing system, vol. 8, pp. 2201–2212, January 2015.

[4] C.Z. Xia and S.L. Song, "Resource scheduling algorithm for hierarchical data grid based on quotient space," Journal of communications, vol. 6, pp. 146–155, June 2013.

[5] X. Liu, J. Yu, J. Wang, et al., "Resource Allocation with Edge Computing in IoT Networks via Machine Learning," IEEE Internet of Things Journal, vol. 7, pp. 3415–3426, April 2020.

[6] H. Li, H. Xu, C. Zhou, et al., "Joint Optimization Strategy of Computation Offloading and Resource Allocation in Multi-Access Edge Computing Environment," IEEE Transactions on Vehicular Technology, vol. 69, pp. 10214–10226, June 2020.

[7] S. E. Mahmoodi, K. Subbalakshmi, and R. N. Uma, "Spectrum-Aware Mobile Computing: Convergence of Cloud Computing and Cognitive Networking,". Springer International Publishing, 2019.

[8] M.Z. Liu, Q. Wang, et al., “Task decomposition method of cloud manufacturing based on Hierarchical Task Network”, China Mechanical Engineering, vol. 28, pp. 924–930, 2017.

[9] Mohamad Roshanzamir, Maziar Palhang, et al., “Efficiency improvement of genetic network programming by tasks decomposition in different types of environments”, Genetic Programming and Evolvable Machines, 2021: 1–38.

[10] G. Latif, N. Saravanakumar, J. Alghazo, et al., “Scheduling and resources allocation in network traffic using multi-objective multi-user joint traffic engineering,” Wireless Networks, vol. 26, pp. 5951–5963, November 2020.

[11] P. Balakrishnan and C.K. Tham, “Energy-efficient mapping and scheduling of task interaction graphs for code offloading in mobile cloud computing,” 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing, vol. 23, pp. 34–41, December 2013.

[12] W. Zhang, Y. Wen, D. Wu, et al., “Collaborative task execution in mobile cloud computing under a stochastic wireless channel,” IEEE Transactions on Wireless Communications, vol. 14, pp. 81–93, January 2015.

[13] J. Han, J. Pei, M.Kamber, et al., “Data mining: Concepts and Techniques”, Elsevier, New York, 2011, pp. 228–321.

[14] Y Li, W.J Zhou, H.K Wang. “F-DPC: Fuzzy Neighborhood-based Density Peak Algorithm”, IEEE Access, 2020.

[15] T.N. Tran, K. Drab, M. Daszykowski, et al., “Revised DBSCAN algorithm to cluster data with dense adjacent clusters,” Chemometrics and Intelligent Laboratory Systems, vol. 120, pp. 92–96, January 2013.

[16] M. Ankerst, M.M. Breunig, H. Kriegel, J. Sander, et al., “Optics: ordering points to identify the clustering structure,” Proc ACM Sigmod Rec, vol. 28, pp. 49–60, June 1999.

[17] X.W. Xu, M. Ester, H.P. Kriegel, J. Sander, et al., “A distribution-based clustering algorithm for mining in large spatial databases,” Proceedings of the Fourteenth International Conference on Data Engineering, vol. 10, pp. 324–331, February 1998.

[18] W. Wang, J. Yang, R. Muntz, et al., “STING: a statistical information grid approach to spatial data mining,” VLDB ’97: Proceedings of the 23rd International Conference on Very Large Data Bases, Athens, pp. 186–195, August 1997.

[19] S. L'Yi, B. Ko, D.H. Shin, et al., "XCluSim: a visual analytics tool for interactively comparing multiple clustering results of bioinformatics data," BMC Bioinformatics, vol. S11–S5, August 2015.

[20] T. Li, H.W. Ge, S.Z. Su, et al., "Density Peaks Clustering Based on Density Adaptive Distance," Microcomputer, vol. 6, pp. 1347–1352, June 2017.

[21] X.X. Han, "Analysis of Chameleon Algorithm based on K-medoids," Modern Trade Industry, vol. 34, pp. 195–196, November 2019.

[22] D.S. Sun, Fast graph clustering in large-scale systems based on spectral coarsening, International Journal of Modern Physics B, 2021, 35(09).

[23] H.H. Zhou, Z. Zhang, Q. Zhang, et al., "Density peak clustering combining shared nearest neighbour and shared inverse nearest neighbour", Journal of China West Normal University, ISSN 1673-5072, CN 51-1699/N, October 2021.

[24] Z.W. Gu, P. Li, X. Lang, et al., "A Multi-Granularity Density Peak Clustering Algorithm Based on Variational Mode Decomposition", Chinese Journal of Electronics, Vol. 30, No. 4, July 2021.

[25] Z.H. Lv, L. Qiao, A.K. Singh, et al., "Advanced Machine Learning on Cognitive Computing for Human Behavior Analysis," IEEE Transactions on Computational Social Systems, vol. 10, pp. 1–9, July 2020.

Biographies

Ziliang Wang received his Bachelor of Engineering degree from Beijing University of Posts and telecommunications in 2017. He is currently studying for a master's degree in network management centre, School of computer science, Beijing University of Posts and telecommunications.

Fanqin Zhou, received his Ph.D. degree in automation from the Beijing University of Posts and Telecommunications (BUPT), China, in 2019. He is currently a Postdoctoral Fellow with the State Key Laboratory of Networking and Switching Technology in BUPT. His current research interests include network slicing and resource management of mobile edge networks.

Lei Feng, received his B.Eng. and Ph.D. degrees in Communication and Information Systems from Beijing University of Posts and Telecommunications (BUPT) in 2009 and 2015. He is an Associate Professor at present in State Key Laboratory of Networking and Switching Technology, BUPT. His research interests are resources management in wireless network and smart grid.

Wenjing Li, is currently a professor at BUPT and serves as a director in the Key Laboratory of Network Management Research Centre. Meanwhile, she is the leader of TC7/WG1 in the China Communications Standards Association (CCSA). Her research interests are intelligent network management, knowledge-driven management and control of B5G/6G networks. Prof. Li is hosting the China first 6G research project, and published more than 120 papers in prestigious journals (e.g., IEEE Transactions, IEEE IoT-J) and conferences (e.g. INFOCOM, ICC, PODC).

Tingting Zhang, received the Master of computer applications degree from University of Chinese Academy of Sciences in 2011. She is currently working in China Mobile Research Institution as a Technical Director. Her current research interests include network virtualization, cloud computing, edge computing, ubiquitous computing and etc.

Sheng Wang, received the Master of Engineering degree from Beijing institute of technology in 2010. He is currently working in China Mobile Research Institution as a Technical Director. His current research interests include NFV/SDN, ubiquitous computing, heterogeneous computing and cloud-network convergence, etc.

Ying Li, received the Master of Engineering degree from Beijing University of Posts and Telecommunications in 2017. She is currently working in China Mobile Research Institution as a project manager.