Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 8899186, 31 pages
https://doi.org/10.1155/2021/8899186

Research Article

Dispersed Computing for Tactical Edge in Future Wars: Vision, Architecture, and Challenges

Haigen Yang,[1] Gang Li,[1] GuiYing Sun,[2] JinXiang Chen,[3] Xiangxin Meng,[4] HongYan Yu,[4] Wenting Xu,[4] Qiang Qu,[5] and Xiaokun Ying[5]

[1] *Engineering Research Center of Wider and Wireless Communication Technology of Ministry of Education, Nanjing University of Posts and Telecommunications, Nanjing 21003, China*
[2] *Chinese People's Liberation Army No. 61416, Beijing 100089, China*
[3] *Beijing Xinli Machinery Limited Liability Company, Beijing 100010, China*
[4] *Beijing Electro-Mechanical Engineering Institute, Beijing 100074, China*
[5] *China North Industry Advanced Technology Generalization Institute, Beijing 100089, China*

Correspondence should be addressed to Haigen Yang; yhg@njupt.edu.cn

Received 5 August 2020; Revised 6 December 2020; Accepted 17 December 2020; Published 7 January 2021

Academic Editor: Shaohua Wan

In the future, the tactical edge is far away from the command center, the resources of communication and computing are limited, and the battlefield situation is changing rapidly, which leads to the weak connection and fast changes of network topology in a harsh and complex battlefield environment. Thus, to meet the needs of communication and computing to build a new generation of computing architecture for real-time sharing and service collaboration of tactical edge resources to win the future war, the dispersed computing (DCOMP) seeks a new solution to satisfy the requirements of fast and efficient sensing, transmission, integrating, scheduling, and processing of various information in the tactical edge. Through the research of a traditional computing paradigm of mobile cloud computing (MCC), fog computing (FC), mobile edge computing (MEC), mobile ad hoc network (MANET), etc., it can be found that these computations have difficulty in meeting the high changing and complex battlefield environment and we propose a novel architecture of DCOMP to build a scalable, extensible, and robust decision-making system, to realize powerful and secure communication, computing, storage, and information processing capabilities for the tactical edge. We illustrate the fundamental principles of building a network model, channel allocation, and forwarding control mechanism of the network architecture for DCOMP called DANET and then design a new architecture, programming model, task awareness, and computing scheduling for DCOMP. Finally, we discuss the main requirements and challenges of DCOMP in future wars.

1. Introduction

With the evolution of the pattern of the major power relationships and the development of military science and technology, the military action may be far away from the homeland in the future called the tactical edge environment. These actions at the tactical edge usually lack support of communication and data processing capabilities, also known as the "first tactical mile," which is far away from the command center, with limited resources of communication and computing. The battle rhythm changes unexpectedly, leading to the frequent fluctuations of network connectivity and rapid changes of topology because of the highly dynamic and complex battlefield environment. These future military actions are a typical "uncertain fog of war" in the tactical edge, which is difficult to directly use the computing infrastructure at the command center in such harsh and complex war environment. It is extremely important to design and establish a new network and computing architecture by using advanced communication and computing technology oriented to the tactical edge, to realize the rapid perception of the situation of the battlefield, the quick integration and scheduling of the resources between the tactical edge nodes, and the efficient processing and transmission of battlefield information.

In recent years, the U.S. Army has always been at the forefront of leading military information technologies and proposed the concept of "net-centric operations and warfare," which is combined with the key role of "information superiority" and "decision superiority." In September 1999, the U.S. Department of Defense (DOD) proposed the concept of "global information grid (GIG)," which represents the third-generation development direction of the Internet [1]. In 2013, the U.S. Air Force proposed the concept of "combat cloud," which integrates the tactical communication network to realize quick exchange of data and resources of each combat unit in the Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance (C4ISR) system [2]. In 2014, the Office of Naval Research (ONR) of U.S. proposed the concept of "tactical cloud," to achieve the data and applications of real-time awareness and process in the battlefield [3]. The "tactical cloud" needs to solve the problems of maintaining information consistency in the condition of the "high dynamic, weak connection" communication environment, software and data security in the cloud environment, dynamic "application tailoring" in different physical platforms, and real-time or near real-time processing of tactical data. In 2016, the Defense Advanced Research Projects Agency (DARPA) proposed a novel Internet architecture: "Dispersed Computing (DCOMP)" program, which is considered the next-generation battlefield environment support technology for the U.S. Army.

In June 2016, Information Innovation Office of DARPA released the proposals of an innovative research soliciting project for DCOMP [4], which is aimed at producing software instantiations of algorithms and protocol stacks that leverage pervasive, physically DCOMP platforms to boost application and network performance by orders of magnitude. The program is comprised of the three technical areas (TAs) as follows and described in more detail below:

(i) *TA1*. Algorithms for dispersed task-aware computation is aimed at developing algorithms and control mechanisms to enable efficient use of networked, geographically dispersed, heterogeneous computing capabilities in a manner consistent with the user, application, and task requirements

(ii) *TA2*. Programmable nodes and protocol stacks want to demonstrate the unique value that accrues from the presence of the programmable protocol logic within the network, primarily at the transport and application layers (but also potentially at the network layer) of the five-layer protocol stack model. TA2 systems may include new functions on users' terminal devices that interact with networked computation points (NCPs) in the network to optimize the overall performance

(iii) *TA3*. Technology integration describes how to combine itself with potential technologies that implement concepts across TA1 and TA2

DARPA entrusted Raytheon BBN Technologies (the premier research and development centers of Raytheon), Vencore Inc., BAE Systems Inc., and LGS Innovations to implement the DCOMP plan. The purpose of the DCOMP project is to improve the ability of task data calculation by using local computing resources, improve the reliability of the field network, distinguish available computing resources, specify the computing tasks of data in order of importance, and try to redesign and innovate the network protocol of the traditional Internet architecture, so as to significantly reduce the delay and bandwidth consumption and improve the performance of applications in a complex and uncertain battlefield environment [5].

The works and contributions of this article are listed as follows:

(i) Based on the studies of DCOMP, we propose the network model, channel allocation, and forwarding control mechanism of the DCOMP called DANET to achieve in realizing a centerless, multihop, self-organizing, infrastructure-free tactical edge network

(ii) We propose the architecture, software stack, programmable model, programmable language, programmable network, task awareness, and computing scheduling for DCOMP

(iii) Furthermore, we illustrate that the advantages, application scenarios, and the simulation result of the improved routing protocol for DCOMP are given

2. Related Work

Recently, the concepts such as distributed computing, cloud computing, MCC, cloudlet, FC, EC, and MEC are emerging endlessly and unable to meet the requirements of the complex battle field environment (e.g., low latency and bandwidth and high error rate and dynamic) that are crucial for future wars.

2.1. Mobile Cloud Computing. Mobile cloud computing uses the cloud computing technology on a mobile device, which brings the services like on-demand access and no on-premises software. MCC uses network capabilities alone to deliver the desired service to customers, which could permit to reserve network bandwidth confirming timely delivery of information to customers. The typical architecture of MCC is shown in Figure 1.

There are various researches about MCC proposed in the literatures focusing on the computation offloading and resource scheduling. Guo et al. [6] provided an energy-efficient dynamic offloading and resource scheduling (eDors) policy to reduce energy consumption and shorten application completion time. Chen et al. [7] proposed a game theoretic approach for the computation offloading decision-making problem among multiple mobile device users for mobile edge cloud computing (MECC). Jo et al. [8] proposed a hierarchical cloud computing architecture to enhance performance by adding a mobile dynamic cloud formed by powerful mobile devices to a traditional general static cloud, which increased the overall capacity of a mobile network through improved channel utilization and traffic offloading

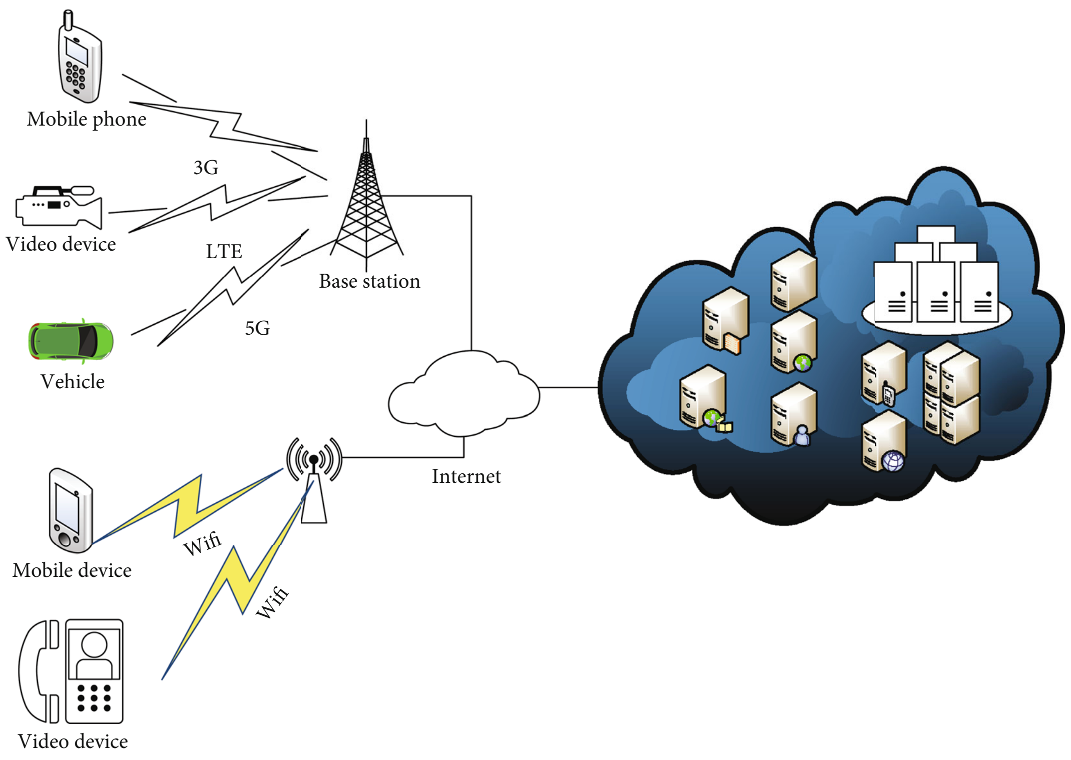


Figure 1: Typical architecture of mobile cloud computing.

from LTE-Advanced to device-to-device communication links. Han et al. [9] developed a unified framework that minimizes the overall outage probability in various mobile computation offloading scenarios. Miao et al. [10] put forward a new intelligent computation offloading based on MECC architecture in combination with artificial intelligence (AI) technology with the increasing requirements and services of mobile users, and an offloading strategy of simple edge computing is no longer applicable to MEC architecture. In 2019, the U.S. DOD officially announced that Microsoft will be responsible for building a cloud computing system for the U.S. military with a project cost of up to 10 billion dollars. In the next 10 years, Microsoft will build a core cloud computing system to achieve more efficient communication for the U.S. Army. The vision of MCC is an autonomous digital environment for different mobile devices to obtain their computation, storage, services, and other resources autonomously and efficiently anytime and anywhere [11].

2.2. Fog Computing. Fog computing is a concept proposed by Cisco Systems, introduced as a new network model to reduce data transfer within the IoT applications [12]. FC is a distributed paradigm that provides cloud-like services to the network edge, which leverages cloud and edge resources along with its own infrastructure [13]. FC involves the components of data processing or analysis applications running in distributed clouds and edge devices. It also facilitates the management and programming of computing, network, and storage services between the data center and terminal devices [14, 15]. In addition, it supports user mobility, heterogeneity of resources and interfaces, and distributed data analysis to meet the needs of widely distributed applications requiring low latency. Some architectures of the FC have been proposed, which were derived from the fundamental three-layer structure, extending cloud service to the network edge by introducing a fog layer between Internet of Things and cloud [16]. The typical architecture of FC is shown in Figure 2.

The hot topics in FC include a computation offloading and resource allocation scheme, scheduling policies, migration method, etc. Gao et al. [17] aimed to minimize the time-average power consumptions with stability guarantee for all queues in the system and exploited unique problem structures and proposed an efficient and distributed predictive offloading and resource allocation scheme for multi-tiered FC. Wu et al. [18] designed a value iteration algorithm of the semi-Markov decision process to maximize the total long-term reward for the task offloading problem of the vehicular fog and cloud computing system. Zeng et al. [19] considered a FC framework to support a software-defined embedded system, where task images lay in the storage server while computations can be conducted on either an embedded device or a computation server, which is significant to design an efficient task scheduling and resource management strategy with minimized task completion time for promoting the user experience. Bittencourt et al. [20] analysed the scheduling problem of FC, focusing on how user mobility can influence the application performance and how three different scheduling policies, namely, concurrent, FCFS, and delay priority, can be used to improve execution based on application characteristics. Osanaiye et al. [21] described an FC architecture, reviewed its different services

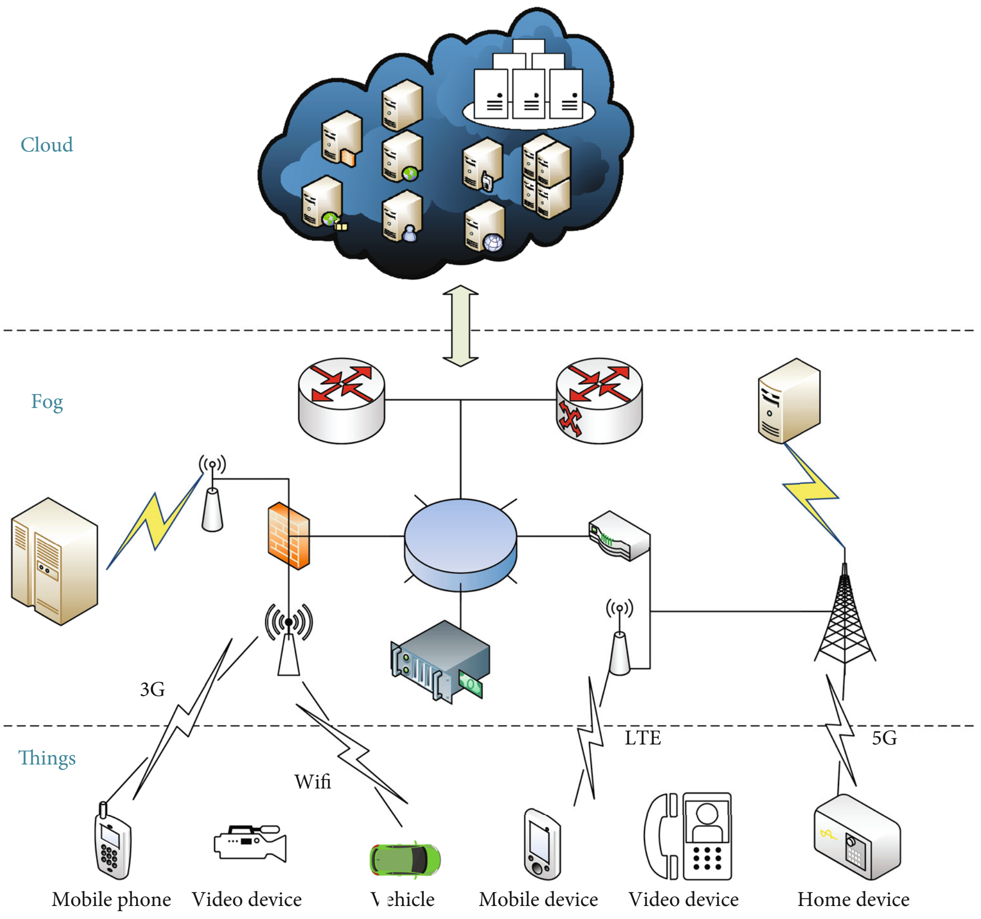


FIGURE 2: Typical fog computing network architecture.

and applications, and presented a conceptual smart precopy live migration approach for VM migration to estimate the downtime after each iteration to determine whether to proceed to the stop-and-copy stage during a system failure or an attack on a FC node.

FC brings many advantages including enhanced performance, better efficiency, network bandwidth savings, improved security, and resiliency. FC is not a replacement for cloud computing but extends the computation, communication, and storage facilities from the cloud to the edge of the networks [22], which is foreseen as a next computing paradigm and can be applied to a wide range of network applications.

2.3. Mobile Edge Computing. Mobile edge computing is a promising paradigm to offer the required computation and storage resources with minimal delays because of "being near" to the users or terminal devices. MEC is aimed at bringing cloud resources and services at the edge of the network, as a middle layer between the terminal user and cloud data centers, to offer prompt service response with minimal delay [23]. MEC refers to the enabling technologies allowing computation to be performed at the edge of the network, on downstream data on behalf of cloud services and upstream data on behalf of IoT services, which defines "edge" as any computing and network resources along the path between data sources and cloud data center. For example, a smartphone is the edge between body things and cloud, a gateway in a smart home is the edge between home things and cloud, and a microdata center and a cloudlet are the edge between a mobile device and cloud [24]. Many experiments are also depending on the environment of edge computing. The typical architecture of mobile edge computing is shown in Figure 3.

In recent years, the researches focus on edge computing including resource management, offloading strategy, and QoS enhancement. Mao et al. [25] provided a comprehensive survey of the state-of-the-art mobile edge computing (MEC) research with a focus on joint radio-and-computational resource management. Mao et al. [26] investigated a green MEC system with energy harvesting devices, developed an effective computation offloading strategy, and proposed a low-complexity online algorithm. Mach and Becvar [27] surveyed the existing concepts integrating MEC functionalities to the mobile networks and discussed current advancement in standardization of the MEC. Taleb et al. [28] proposed an approach to enhance users' experience of video streaming in the context of smart cities, whose approach relies on the concept of MEC as a key factor in enhancing QoS. Xu et al. [29] developed a collaborative method, named CQP, for the quantification and placement of edge servers (ESs). Zhou et al. [30] proposed a latency-aware microservice mashup

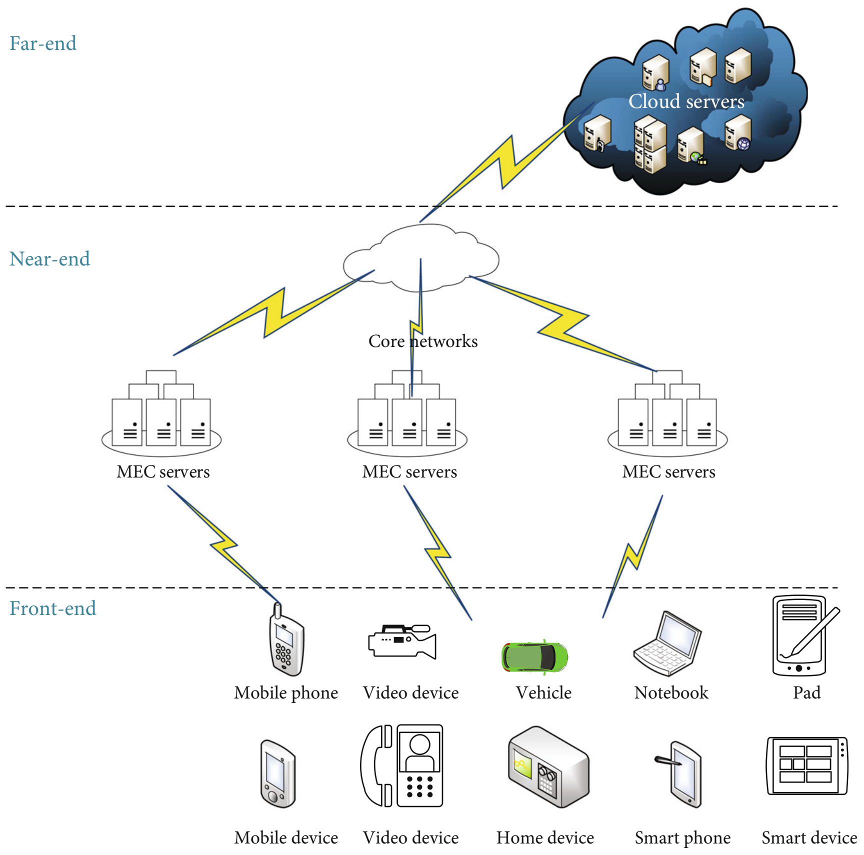


FIGURE 3: Typical architecture of mobile edge computing.

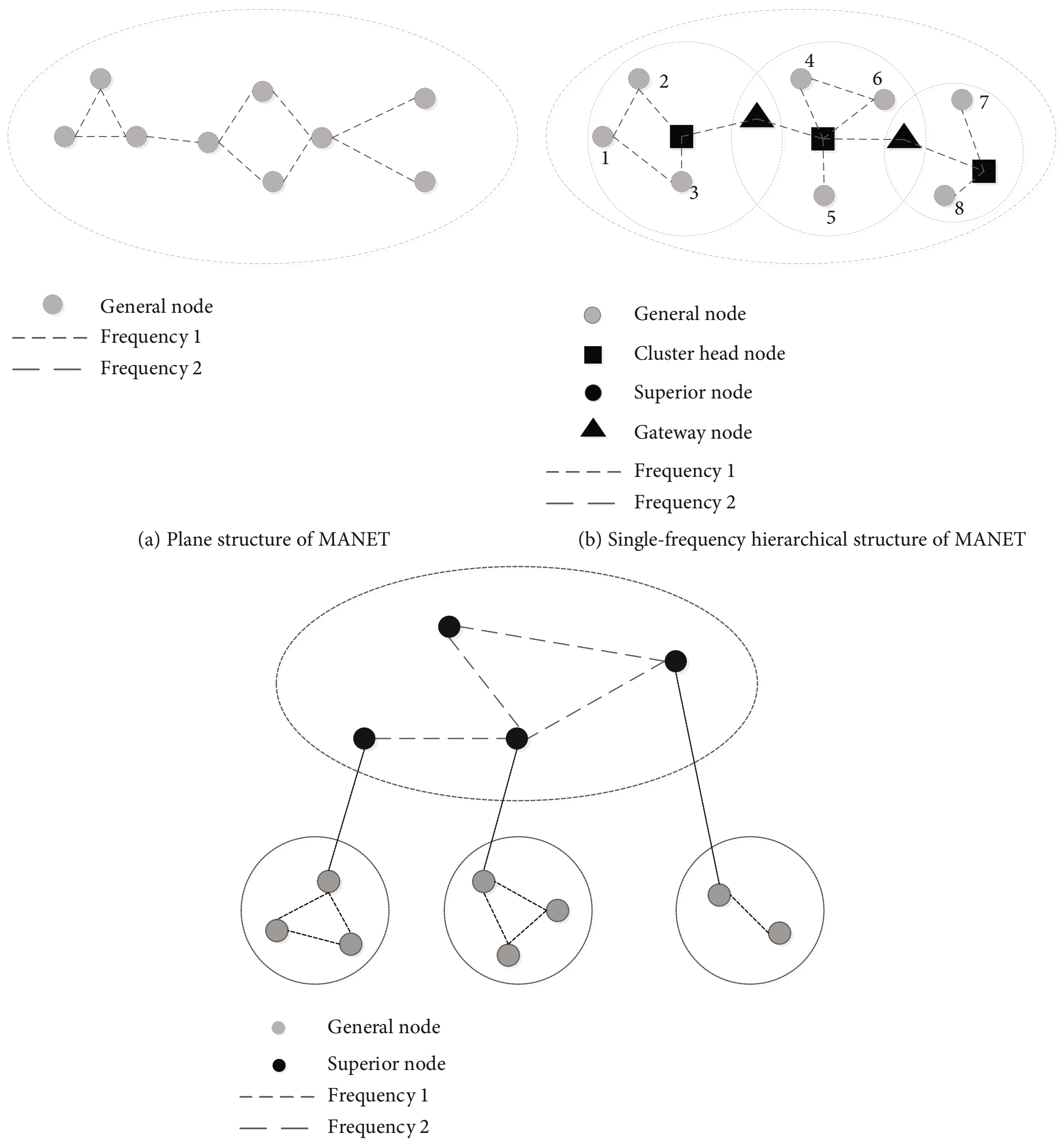
approach and the NP-hardness of the problem by reducing it into the delay-constrained least cost problem. Cao et al. [31] proposed a MEC-based system, in line with a big data-driven planning strategy, for charging stations.

MEC wants to put the computing at the proximity of data sources, has several benefits compared to the traditional cloud-based computing paradigm [32], and allows edge nodes to respond to service demands, reducing bandwidth consumption and network latency.

2.4. Mobile Ad Hoc Network. A mobile ad hoc network consists of multiple communication devices and terminals with transceivers, which can complete the communication process without infrastructure [33–36]. A multihop autonomous system built by MANET nodes has the advantages of no center, self-organization, and rapid networking [37]. The distributed network control and scalability of MANET bring great flexibility and convenience to the network deployment and practical applications [38]. However, it is difficult to predict the topology changes of MANET with the low frequency efficiency and large transmission delay [39]. The three typical architectures of MANET are shown in Figure 4.

The MANET researches focus on the routing protocol, QoS, energy management, etc. Zhang et al. [40] proposed a new greedy forwarding improvement routing method for MANET to calculate the reliable communication area and evaluate the quality of the link according to the relative displacement between the nodes and the maintenance time of the link. Sharifi and Babamir [41] presented a new clustering method and considered the good performance of Evolutionary Algorithms (EAs) good in finding proper head clusters and a specific EA-based method named Imperialist Competitive Algorithm (ICA) via numerical coding by considering the high efficiency of clustering methods among the routing algorithms. Zhou et al. [42] proposed a traffic-predictive QoS on-demand routing (TPQOR) protocol to support QoS bandwidth, delay requirements, channel assignment, and reuse schemes to reduce the channel interference and enhance the channel reuse rate. Ubarhande et al. [43] proposed a distributed delegation-based scheme to identify and allow the only trusted nodes to become part of the active path to improve the packet delivery ratio, packet loss rate, throughput, and routing overhead. Khosravi et al. [44] focused on the underwater communication and proposed a new method, using an intelligent 3D random node removal mechanism considering the traffic status of the network, to improve energy efficiency of the underwater acoustic wireless sensor networks and network reliability and better network lifetime. Pal and Jolfaei [45] proposed a software-defined wireless sensor network (WSN) architecture which conserves energy by applying asynchronous duty cycling.

MANET is conceived for military applications and aimed at improving battlefield communications and survivability originally; multihop MANET has lately been proposed in many civil scenarios, which is still difficult to achieve the large-scale applications [46]. As far as the application



(a) Plane structure of MANET

(b) Single-frequency hierarchical structure of MANET

(c) Multifrequency hierarchical structure of MANET

FIGURE 4: Typical architecture of MANET.

environments of these networks increase, the traditional communication paradigms of MANET need adequacy [47].

2.5. Delay/Disruption-Tolerant Network. A delay/disruption-tolerant network is proposed by DARPA and Internet Research Task Force (IRTF) to solve the communication problems in the network such as frequent interruption of communication links and large transmission delay [48–51]. DTN can be widely used in the area of interplanetary networks, military communications, wildlife tracking system, village-to-village networks in developing regions, vehicular networks, and nomadic communication networks [52]. In these networks, because of the frequent movement of network nodes, communication attenuation, and interference, the limitation of energy and storage space of nodes results in the long-time interruption of the network link, large delay, and high packet loss rate, which leads to the failure of the normal communication process and poor transmission performance [53, 54].

The research activities in DTNs recently are being investigated for the design of an application/convergence layer, routings, congestion/flow control, and security strategies, which are briefly introduced as follows: Banerjee et al. [55] presented a hardware and software architecture for energy-efficient throw boxes in DTNs and a novel paradigm for power management in DTNs provided for more efficient neighbor discovering by detecting the mobility of other nodes at a minimum cost and predicting the cost and opportunity of each possible contact, which can intelligently choose the most fruitful contact opportunities and limit the number of opportunities to meet energy constraints. Seligman et al. [56] proposed a congestion management solution of the last

Table 1: Summary of MCC, FC, MEC, MANET, and DTN.

Features	MCC	FC	MEC	MANET	DTN
Target user	Mobile user	Mobile user	Mobile device	Mobile device	General device
Server number	Few	Large	Large	Large	Large
Server deployment	Centralized	Near edge	Network edge	Device edge	Device edge
Reliability	High	Low	Low	Low	Low
Location awareness	Yes	Yes	Yes	Yes	Yes
Real-time interaction	Support	Support	Support	Support	Support
Latency	Low	Low	Low	High	High
Jitter	Low	Low	Low	High	High
Power	Ample	Limited	Limited	Limited	Limited
Storage capacity	Ample	Limited	Limited	Limited	Limited
Offloading	Yes	Yes	Yes	No	No
Programmable	No	Yes	YES	No	No
VM support	Yes	Yes	Yes	No	No

form for storage routing by employing nearby nodes with available storage to store data that would otherwise be lost when given uncontrollable data sources, which determines a collection of messages and neighbors to which they migrate using a set of locally scoped distributed algorithms, possibly incorporating loops that are known to be optimal for some DTN routing scenarios, and decouples storage management from global DTN route selection. Whitbeck and Conan [57] proposed a hybrid DTN-MANET routing protocol using DTN between disjoint groups of nodes while using MANET routing within these groups, which is fully decentralized and only makes use of topological information exchanges between the nodes. Mao et al. [58] proposed a new routing protocol, called scheduling-probabilistic routing protocol, using history of encounters and transitivity, and calculated the delivery predictability according to the encountering frequency among nodes to improve performance in both storage and transmission in DTN. Table 1 summarizes the features associated with MCC, FC, MEC, MANET, and DTN.

It can be seen from Table 1 that the current MCC, FC, MEC, MANET, and DTN have their own characteristics and advantages, but these computing paradigms are difficult to adapt to the requirements of high dynamic, low delay, and fast response at the tactical edge in the future.

3. The Features and Visions of DCOMP

The highly dynamic and complex battlefield environment results in the weak connection and highly dynamic characteristics of the tactical network, which puts forward strict requirements for the underlying computing and network infrastructure of the network. In the current technology, users usually rely on a large and highly shared data center to send data (such as image, video, or situation information) back to the data center for processing tasks with a large amount of computation. However, the rapid changes in the battlefield environment, the cost, and the delay of such backhaul may be problematic, especially when the network throughput is severely limited or user applications need near real-time response. Using the ability of "locally" available computing resources (from the perspective of latency, available throughput, task related, etc.) to calculate replication tasks could significantly improve the performance of applications and reduce the processing risk of tasks.

DCOMP provides a new way to improve the computing and communication capabilities in the harsh battlefield environment, which seeks scalable and robust communication and computing systems to meet users with competitive needs to safely and execute computing tasks collaboratively on a large number of heterogeneous computing devices running on a highly variable and degraded network environment.

DCOMP is different from the traditional network architecture completely, which does not regard the devices as the nodes of data transmission, but as distributed computing resources on the network. It can change the real-time dynamic network address at any time according to the needs. The distributed computing scenario consists of a group of NCPs with different computation abilities, such as a wireless access point, network router, handheld terminal, camera, and mobile computer. [59], as shown in Figure 5.

The concept of DCOMP has only been proposed in recent years, and systematic research on DCOMP is still relatively lacking, especially for network programmable protocols, computation offloading, task decomposition, etc.

García-Valls et al. [60] identified the new computing paradigm called social dispersed computing, analysed the key themes, and gave the outlook on its relation to agent-based applications, and examples include next-generation electrical energy distribution networks, next-generation mobility services for transportation, and applications for distributed analysis and identification of nonrecurring traffic congestion in cities. Hu and Krishnamachari [61] proposed a throughput optimized task scheduler, targeting applications (such as computer vision and video processing) where input data are continuously and steadily fed into the execution pipeline for DCOMP to perform computation on the edge leading to significant reduction in communication with the remote cloud. Fujikawa et al. [62] described DCOMP as a new vision of joint computation and communication resource

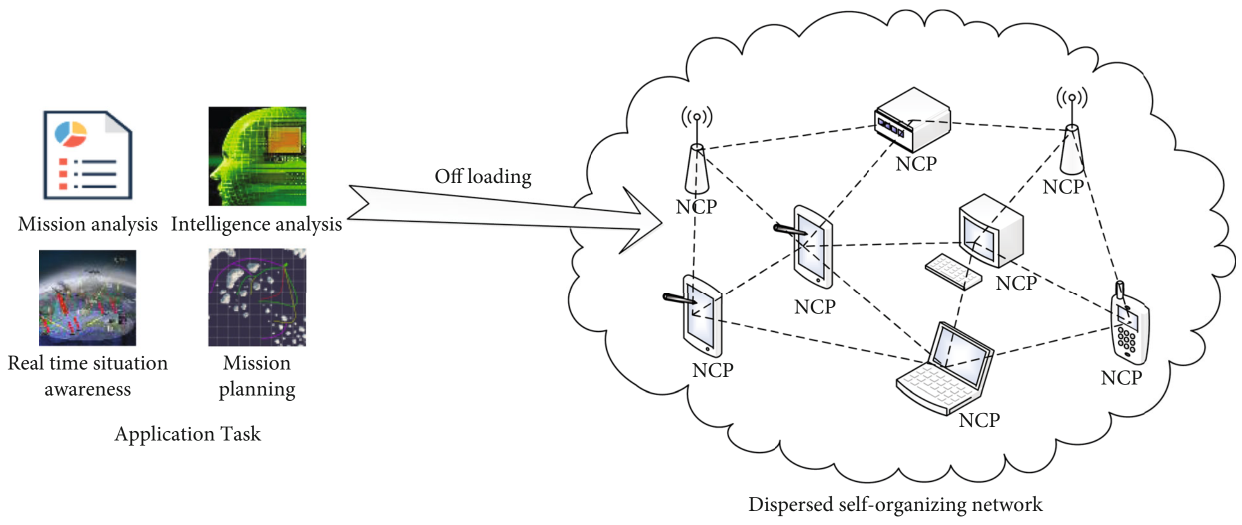


Figure 5: Schematic diagram of DCOMP.

management that goes beyond the end-to-end and client-server model of the current Internet and a new resource-centric architecture that leverages the diversity of networked computation points within the network and the heterogeneity of network links and protocol stacks that connect them. Yang et al. [63] considered the problem of task scheduling for such networks, in a dynamic setting in which arriving computation jobs are modelled as chains with nodes representing tasks and edges representing precedence constraints among tasks, and proposed a model motivated by significant communication costs in dispersed computing environments, and the communication times are taken into account. Knezevic et al. [64] designed a runtime scheduling software tool for DCOMP, which can deploy pipelined computations described in the form of a Directed Acyclic Graph (DAG) on multiple geographically dispersed compute nodes at the edge and in the cloud. Nguyen et al. [65] studied file transfer times between geographically dispersed cloud computing points using SCP (secure copy) and demonstrated via a set of real-world measurement experiments that the end-to-end file transfer time in a dispersed computing environment can be modelled as a quadratic function of the file size. Ghosh et al. [66] presented a container orchestration architecture for DCOMP and its implementation in an open-source software called Jupiter, which automates the distribution of computational tasks for complex computational applications described as a DAG to efficiently distribute the tasks among a set of networked compute nodes and orchestrates the execution of the DAG thereafter.

We believe that DCOMP has the following characteristics and advantages [67–69]:

(i) DCOMP should have the ability to coordinate various computing resources in heterogeneous networks for collaborative computing according to specific tasks and environment

(ii) DCOMP nodes have programmable capabilities and can respond dynamically according to the change of the network conditions, and the overhead associated with probing or message transmission between nodes must not significantly reduce throughput to support scalable, robust operation

(iii) DCOMP can quickly sense network bandwidth and topology changes with the movement of the nodes to ensure fast response to data service requests without having to send the data back to the rear data center to process the data locally, which can reduce the delay in data processing and improve the real-time capability of the combat system

(iv) DCOMP has the ability of crossing heterogeneous computing platforms to achieve more computing capabilities, by performing centralized task distribution. The various network components, radios, smartphones, sensors, and portable cloud computing equipment in DCOMP can be reasonably applied in the programmable execution environment to maximize the computing capacity of the battlefront

4. Network Model, Channel Allocation, and Forwarding Control Mechanism of DCOMP

In the future, the hostile battlefield needs to establish a centerless, self-organized tactical communication network consisting of tactical radio stations and individual handheld/-vehicle/airborne/shipborne wireless terminals, which can be regarded as a MANET.

The network of DCOMP is a centerless, multihop, self-organizing, infrastructure-free, peer-to-peer communication network composed of various wireless communication nodes without a strict central node. All nodes use on-demand routing protocols and a proactive routing mechanism to coordinate their behaviours, which form an independent network quickly and automatically and perform calculations

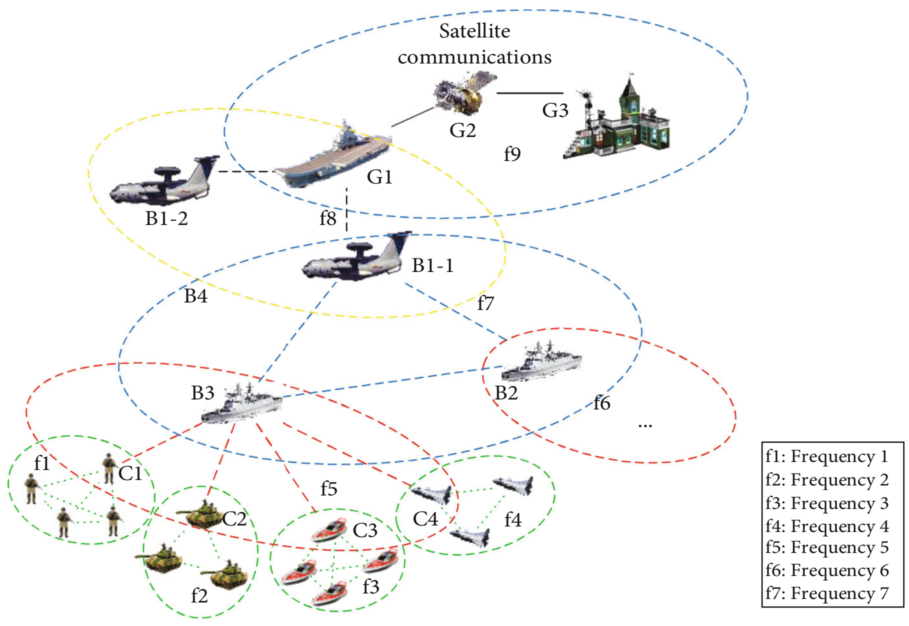


Figure 6: The typical structs of DANET.

spontaneously and collaboratively. We call them the dispersed ad hoc network (DANET).

4.1. Network Model of DANET. Each tactical end node in the DANET has both the role of a router and computing host. The nodes in a DANET environment need to run combat-oriented applications and corresponding routing protocols as routers. The DANET is different form MANET for the reasons of topology changing faster, moving at different speed, lower latency required, and nodes joining or exiting the net at any time.

It is difficult to communicate the real-time battlefield environment, situation, and combat command data through DANET in a hostile battlefield combat environment. The typical structs of DANET in the battlefield environment is shown in Figure 6, which is a hierarchical network composed of independent communication subnets (in each virtual box) with the original characteristics maintained. A node is selected as a cluster head in each subnet, and the cluster head is not only a member of the subnet at this level but also a member of a superior network.

This struct of DANET enables battlefield information to be sent directly from the lowest level to the upper level and enables command orders to be sent directly from the upper level to the lowest level, reducing manual forwarding though intermediate links and improving timeliness. The multifrequency hierarchical structure adopted by the DANET implements functions such as command communication, situational forwarding, and backbone network connection.

4.2. Channel Sharing and Allocation of DANET. The network is shown in Figure 6, and each subnet uses the same frequency and shares the same channel. Nodes can perform unicast, multicast, and broadcast communications in the same subnet. Different subnets use different frequencies and channels to avoid interference between the nodes and improve the frequency reuse rate. The cluster head (such as the nodes of C1, C2, C3, C4, B3, B2, B1-1, and G1 in Figure 6) is responsible for forwarding data to achieve communication between different subnet nodes. The cluster head has a duplex frequency and can work on two channels at the same time, of which one is responsible for communicating with the subnet and the other channel joins the upper-level network. Therefore, the cluster head plays a role as the gateway node connected to other subnets.

The DANET can use fixed channel allocation to solve the problem of channel allocation and enable the network to achieve a good balance in reducing interference and improving network connectivity. The channels of the entire network are uniformly allocated before communication, and the available channels are divided into working channels and standby channels allocated to each subnet. The different adjacent subnets have different channels to avoid interference. If the working channel is subject to external interference, each node within the subnet will use the standby channel according to the relevant agreement. At the same time, multiple subchannels are allocated to each subnet, and the channel is changed when external interference is encountered or at a specified time according to the agreement. Each subnet is used strictly in accordance with the rules of the agreement to avoid problems of blind channel use and channel resource contention.

In the tactical edge network, a large number of real-time applications (such as voice, video, or urgent tasks) are generated with the constant change of network requirements. In the traditional MANET, the mobile nodes move in the network at will, which will cause strong dynamic characteristics. The topology of DANET work changes with the movement of nodes in unpredictable ways and at unpredictable speeds, which may cause communication interference between adjacent nodes leading to seriously affecting throughput and transmission delay of the network.

Table 2: Comparison of different routing protocols.

Protocol	AODV	OLSR	DSDV	CGSR	LANMAR
Proactive/reactive routing	Reactive	Mix	Proactive	Proactive	Proactive
Location based	No	Yes	No	No	Yes
Hierarchy	Plane	Hierarchy	Plane	Plane	Hierarchy
Multicast	Yes	Yes	No	No	No
Routing discovery delay	High	Low	Low	Low	High
Routing overhead	Low	General	High	High	Low
Advantages	Compression control information	Available immediately	Respond quickly	Small time delay	Small routing table size
Disadvantages	Best performance at low congestion & high density	More resource than AODV	More useless routing information	Weak adaptability of topological change	Large storage control

The traditional IEEE 802.11 standard uses the shared channel model, and the interference increases with the increase in the number of mobile nodes, resulting in a significant decrease in network performance. In the environment with dense communication nodes, the application of Multiple Access Control Protocol (MACP) will adversely affect network performance. When a mobile node enters the communication range of another pair of node pairs, each node can carry out channel switching, which will cause the mobile network to continuously receive interference, and channel allocation is needed for dynamic channel management. In order to effectively solve the channel interference problem caused by exposed nodes in DANET, the channel allocation control algorithm with power control can be used to dynamically negotiate the channel to carry out adaptive channel allocation. It can realize multiple communications of different channels in the same area, reduce the negative effect caused by the exposed node problem, and improve the capacity of the network [70, 71].

4.3. Routing Protocol and Forwarding Control of DANET. Due to the feature of poor connection and high dynamic of the DANET, the routing protocols of traditional MANET cannot adapt to the low latency and high reliability requirements at the tactical edge. The routing protocols of DANET must satisfy the actual operational requirements of fast topology changes, high real-time performance, and strong anti-interference capabilities. Therefore, it is necessary to establish a dispersed organization-aware network programmable routing protocol and forwarding control strategy in a harsh battlefield environment.

A hierarchical routing protocol needs a complex cluster head election algorithm, and the cluster heads are easy to become the bottleneck of the network for the reason of most of data forwarded by cluster heads. The typical hierarchical routing protocols include CGSR (Cluster-head Gateway Switch Routing Protocol), HSR (Hierarchical State Routing Protocol) [62], LANMAR (Landmark Ad Hoc Routing Protocol) [72], etc., and the common table-driven routing protocols are DSDV (Destination-Sequenced Distance Vector) [73], WRP (Wireless Routing Protocol) [74], OLSR (Optimized Link State Routing) [75], AODV (Ad hoc On-demand Distance Vector Routing) [76], etc. It can be found that there are multiple routing protocols used in the current MANET, and different types of protocols have different mechanisms with their own characteristics and different environmental adaptabilities. Table 2 shows the different application scenarios and advantages and disadvantages of the typical routing protocols.

Form Table 2, we can see that the different types of routing protocols have different mechanisms in MANET; each protocol has its own characteristics and adapts to different environments. For example, WRP and GSR have fast convergence speed, but the node burden is heavy, which is not suitable for the network with strict requirement on node energy; OLSR is suitable for the networks with small mobility, frequent internode communication, and large scale; AODV has better performance in high-load, high-mobility networks. The AODV and OSLR are typical and mature routing protocols of the MANET, which are very suitable for the traditional MANET. Because the topology change speed in traditional MANET is less than the tactical edge in future wars, the original AODV and OLSR routing protocols are no longer suitable for this highly dynamic tactical edge environment. We propose a routing protocol combination of OLSR and AODV to solve hierarchical routing of networks in harsh battlefield environments for DANET. The routing protocol of AODV can be used as a routing protocol for small wireless ad hoc networks within a subnet, and OLSR as a routing protocol between subnets. The original AODV and OLSR routing algorithm is difficult to adapt to complex battlefield environments and needs to introduce an improved algorithm of OLSR and OLSR protocols for DANET.

4.3.1. Improvement of OLSR Protocol Based on Dynamic Link Cost. Applying the OLSR protocol as a tactical edge backbone network (between tactical edge subnets) is more conducive to the rapid forwarding of command and control information. The workflow of the OLSR protocol is shown in Figure 7.

However, in the MPR (Multipoint Relays), selection of the original OLSR protocol only considers the state of physical connectivity of the link, but the bandwidth status of the link is not considered. In the path calculation, the original

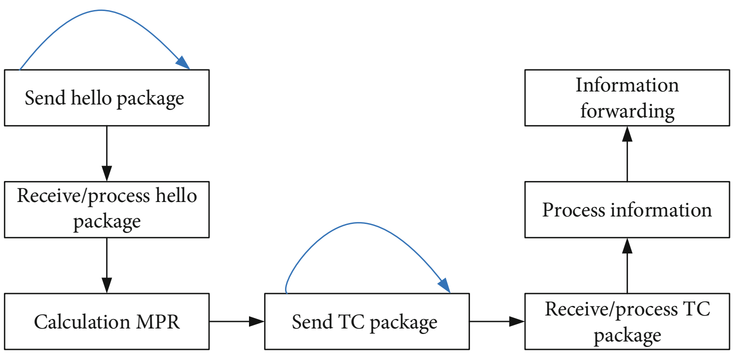


Figure 7: Workflow of the OLSR protocol.

OLSR protocol only considers the minimum number of hops without link latency, but the delay of the link path is not considered. In the harsh battlefield environment, the amount of various types of burst data is very large transmitted along the shortest path, which can easily cause network congestion, resulting in the decrement of network throughput and packet transmission rate and increment of transmission delay. The original OLSR routing protocol only takes the physical connectivity of the link into account, not considering the bandwidth of the link, which obviously cannot meet the requirement of high service quality in DANET. Therefore, in order to improve this situation of possible congestion, the original OLSR protocol needs to improve by considering link cost of message flooding and path selection to avoid the bottleneck of the network for data transmission.

In order to implement the improved OLSR protocol based on the dynamic link cost, it is necessary to modify the Hello packet, TC packet, routing topology table, etc. and add "Cost" information into the package. In the complex battlefield environment, the delay requirement of information transmission is very strict. The dynamic link cost function should not be too complicated, which needs to adjust according to changes of network conditions (such as bandwidth, delay, packet loss rate, and energy consumption) to reduce routing congestion. Therefore, the "Cost" can be set as

$$\text{Cost} = 1/\text{B\_link}, \tag{1}$$

where "Cost" is the link weight and "B_link" is the link bandwidth.

We can add "Cost" to the Hello packet (a), TC packet (b), and routing topology table (c), as shown in Figure 8.

In the Hello packet, the "Cost" is the cost of the link from the node to the neighbor node. In the TC packet, the "Cost" is the link cost between the node and the MPR node. In the routing topology table, the "Cost" is the link cost between "T_dest_Address" and "T_last_Address." During path calculation, the cost of the link is obtained from the network topology table, which is used as the link weight value, and the Dijkstra algorithm is called to obtain the optimized path after considering the link bandwidth status.

4.3.2. Improvement of the AODV Protocol Based on Dynamic Link Cost. Each router (the cluster head) in the DANET can access the node in the subnet, and the subnet can form multiple relatively small wireless ad hoc networks through wireless connections. The AODV protocol is used at the inside subnet of the MANET. The original AODV protocol has well performance when the nodes move faster and a large amount of business data uses the minimum hop path between the source node and the destination node for routing selection. The minimum hop path is not necessarily the best path, which can cause network congestion and affect network performance. Therefore, the routing algorithm based on dynamic link cost is also needed to improve the original protocol of AODV in DANET.

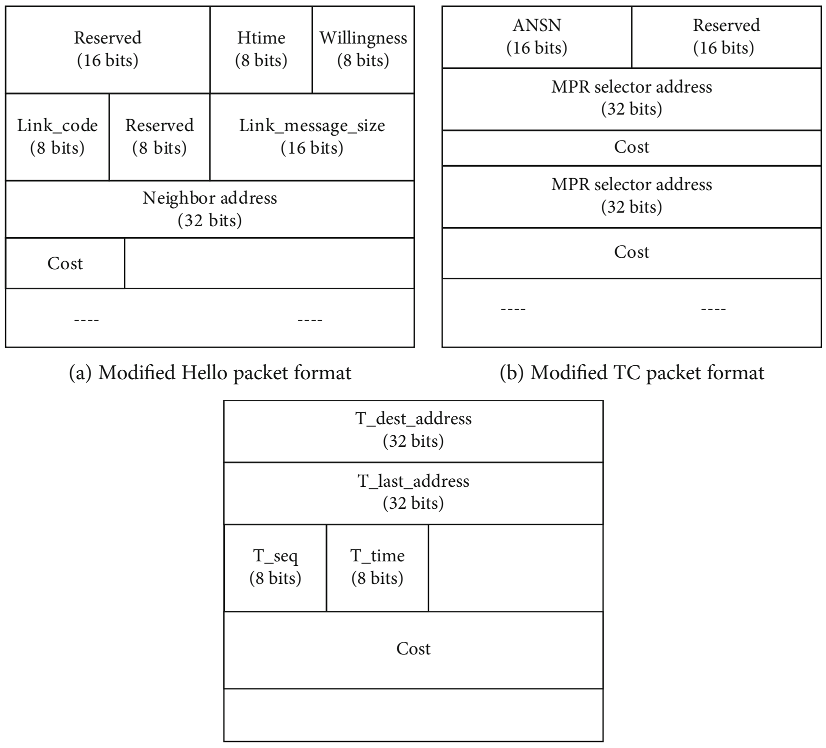
Define $\text{Path}(s, d)$ as all feasible paths from source node s to destination node d, set k as the number of elements in the path, and $\text{Path}_i(s, d)$ represents a feasible path from node s to node d $(i \leq k)$. The hop count of the path is $\text{hop}(\text{Path}(s, d))$, and e_{ij} represents the link between nodes V_i and V_j. Set the bandwidth of the link to be B_link, the real-time bandwidth of the link is B_use, and the request bandwidth of the service is B_req.

The improved model must satisfy

$$\begin{aligned} \text{Bf} &= \min\left\{B\_\text{req} - (B\_\text{link} - B\_\text{use})\right\}, \\ \text{Br} &= \max\left\{\frac{B\_\text{link} - B\_\text{use}}{B\_\text{link}}\right\}. \end{aligned} \tag{2}$$

(i) Bf: bandwidth fragment. When selecting a link, one should try to select a link with a request bandwidth that is close to the actual available bandwidth value, so that the bandwidth fragmentation is as small as possible to achieve the purpose of improving bandwidth utilization

(ii) Br: bandwidth ratio. Under the condition that the requested bandwidth and the actual available bandwidth are as close as possible, the links with a large ratio of the actual available bandwidth to the link's own bandwidth should be selected to reduce the probability of congestion on the link caused by sudden bandwidth requirements



(a) Modified Hello packet format

(b) Modified TC packet format

(c) Modified routing topology table

Figure 8: Add dynamic link "Cost" to improve the OLSR protocol.

Based on the above two indicators, the link cost function is determined as

$$\mathrm{Cost}=\begin{cases}\dfrac{\alpha\left|(B_{\mathrm{link}}-B_{\mathrm{use}})/B_{\mathrm{link}}\right|+\beta\lg\left|B_{\mathrm{reg}}-(B_{\mathrm{link}}-B_{\mathrm{use}})\right|}{\mathrm{Hop}(\mathrm{Path}_i(s,d))}(B_{\mathrm{r}}\neq 1),\\[2ex] \dfrac{1}{\beta\left|B_{\mathrm{reg}}-(B_{\mathrm{link}}-B_{\mathrm{use}})\right|}(B_{\mathrm{r}}=1),\end{cases} \tag{3}$$

where α and β are adjustable proportion coefficients of bandwidth fragmentation and bandwidth ratio, and the size of the two can be adjusted according to the demand; $\mathrm{hop}(\mathrm{Path}_i(s,d))$ is the number of hops from node s to node d. The link "Cost" value can be adjusted according to the link bandwidth status and traffic demand status in real time.

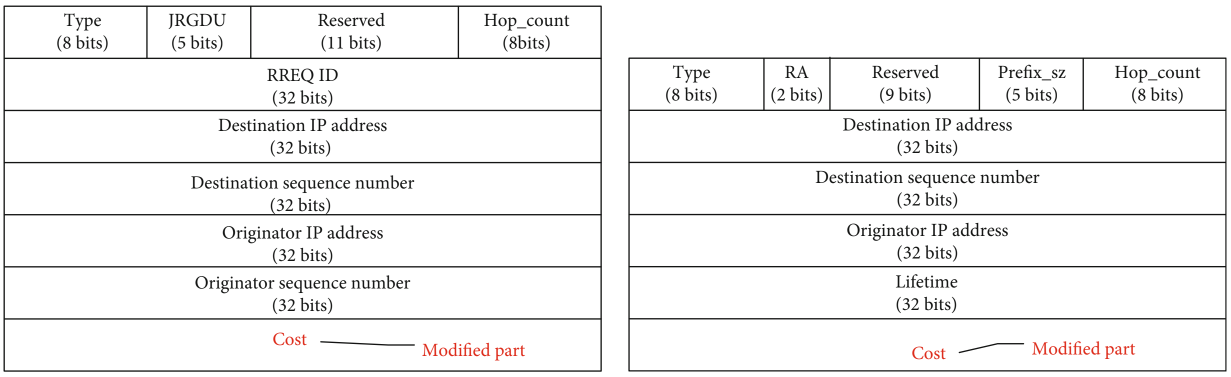
The "Cost" is added to the routing request packet and response packet, respectively, and the cost information of the path can be obtained when the routing calculation is finally performed, as shown in Figure 9.

4.4. Congestion Control Strategy of DANET. The DANET also can be considered a typical DTN, and the congestion control of this kind of network has the following challenges: (1) there may be no communication opportunities in the future, (2) the received save transmission messages cannot be dropped, (3) reserve a cache space for different service types, and (4) reject new connections when storage is full. The research of congestion control strategies in DTN mainly includes node packet loss strategies and message weight calculation models. The entire process of the congestion control strategy includes the processes of receiving messages and comparing weights of sending messages. To judge the message discarding according to the current congestion and weight comparison, we design a specific method shown in Figure 10. If there is no congestion, the message is put into the buffer; if it is congested, the preservation weight of the message is compared with the preservation weight of the node. If the message weight is greater than the node weight, the message with the smallest weight in the node buffer is discarded until there is enough space in the buffer to accommodate the new message.

For the calculation of the weight of a message, the size of the message, the initial lifetime, and the remaining survival time are mainly considered. Formula (4) gives the calculation method of message weight:

$$\mathrm{W}_{ri}=\frac{\mathrm{TTL}_{mi}}{\mathrm{TTL}_0}\times\frac{\mathrm{BS}_j}{\mathrm{S}_{mi}}, \tag{4}$$

where W_{ri} represents the weight when the message i arrives at node j, TTL_{mi} represents the remaining survival time for message i, TTL_0 represents the initial lifetime of all messages in the network, BS_j is the buffer size of node j, and S_{mi} is the size of message i. As can be seen from Equation (4), the shorter the remaining survival time, the lower the probability of the message reaching the destination node; the smaller the weight of the message, the larger the size of the message; and


Type (8 bits)
JRGDU (5 bits)
Reserved (11 bits)
Hop_count (8bits)
RREQ ID (32 bits)
Destination IP address (32 bits)
Destination sequence number (32 bits)
Originator IP address (32 bits)
Originator sequence number (32 bits)
Cost
Modified part
Type (8 bits)
RA (2 bits)
Reserved (9 bits)
Prefix_sz (5 bits)
Hop_count (8 bits)
Destination IP address (32 bits)
Destination sequence number (32 bits)
Originator IP address (32 bits)
Lifetime (32 bits)
Cost
Modified part

(a) Routing request packet

(b) Routing response packet

Figure 9: Add dynamic link "Cost" to improve the AODV protocol.

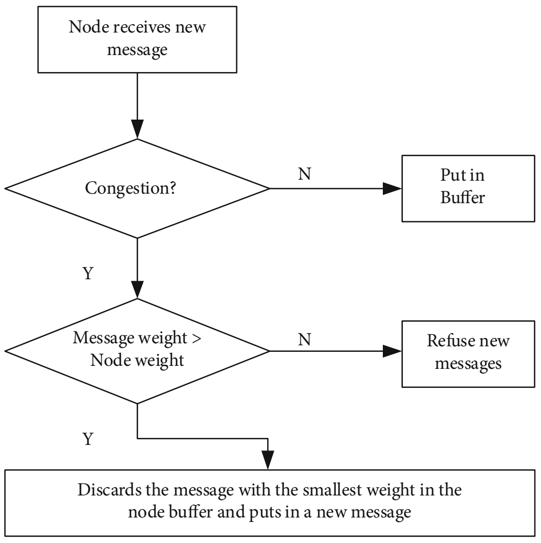

Node receives new message
Congestion?
N
Put in Buffer
Y
Message weight > Node weight
N
Refuse new messages
Y
Discards the message with the smallest weight in the node buffer and puts in a new message

Figure 10: The process of the congestion control strategy for DANET.

the greater the preservation cost, the lower the transmission success rate and the larger the weight of the message.

The formula for calculating the weights of nodes is shown in

$$\mathrm{EW}_j = \frac{\sum_i^{M_j} W_{ri}}{M_j}, \tag{5}$$

where M_j is the total number of messages in the buffer of node j and EW_j is the average preservation weight of node j.

When a node tries to send a message, check whether there are any messages in the buffer that are smaller than the transmission time threshold. If it exists, it uses the greedy strategy to preferentially forward the message with the smallest transmission time; otherwise, it does not forward it. The transmission time threshold is derived from the node's moving speed and communication range.

For each node j, the transmission time threshold is difficult to calculate and can be estimated by

$$\mathrm{Flag}_j = \frac{S_{Lj}}{V_{Nj}} \times \delta, \tag{6}$$

where S_{Lj} represents the communication range of node j, V_{Nj} represents the moving speed of node j, and δ represents the weighted value of transmission. When the communication range of node j is larger, the communication time between other nodes and node j is longer. When the moving speed of node j is faster, the contact time between node j and other nodes is shorter.

Formula (7) gives the calculation method for the forwarding delay of message i:

$$W_{ij=\frac{S_{mi}}{V_{Tj}}}, \tag{7}$$

where W_{ij} represents the forwarding delay of the message i and buffered in node j, S_{mi} is the size of the message i, and V_{Tj} represents the data transmission speed of node j. The forwarding delay of message i is obtained by the ratio of the size of message i to the data transmission rate of node j. When the size of message i is larger, the transmission delay at a certain transmission speed is longer and the buffer of node j is larger, resulting in the greater cost of node j to forward message i and the lower success rate of forwarding. When the forwarding speed of node j is faster, the forwarding delay of each message in the buffer is shorter, resulting in the forwarding success rate being higher.

5. The Architecture Design of DCOMP

In the scenario of DCOMP, the communication bandwidth between different nodes is very limited and heterogeneous, and the computing nodes support different computing capabilities. Figure 11 shows the dataflow and architecture of DCOMP, which connects nodes with computing capabilities to the DANET to communicate directly or indirectly, greatly reducing the time for data transmission and processing. In the paradigm of DCOMP, the nodes are both consumers of data and producers of data.

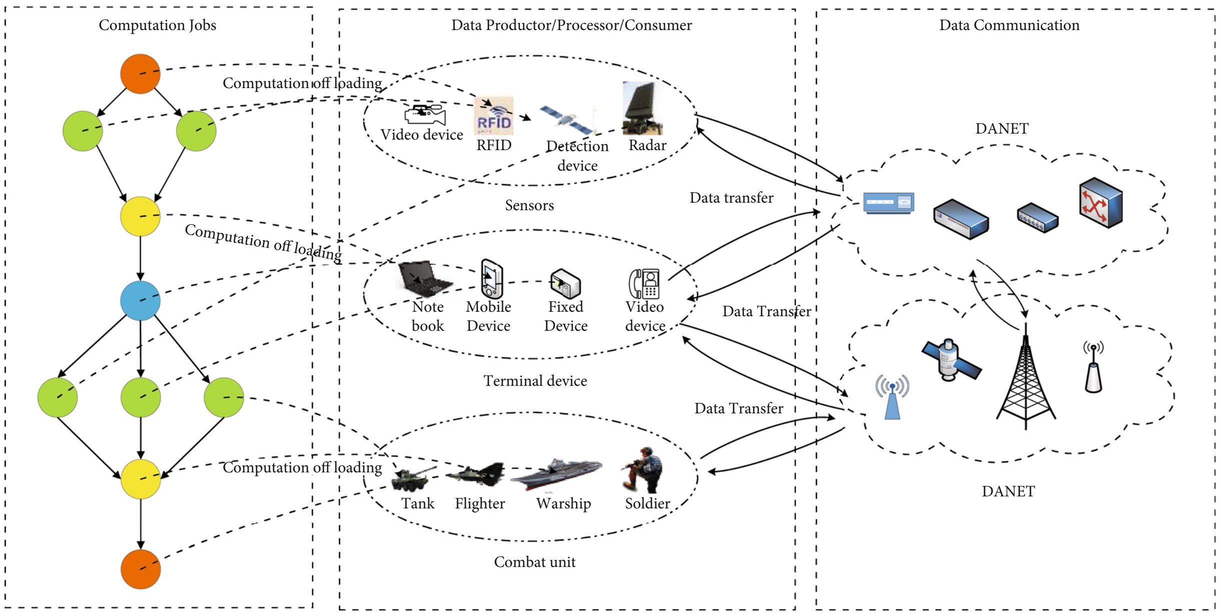


Figure 11: The dataflow and architecture of DCOMP.

As shown in Figure 11, the architecture of DCOMP can be seen that the data can be processed not only at local but also on geographical NCPs. These dispersed computing nodes can perform data collection/transfer/caching/processing, computation offloading, and task scheduling to realize the function of traditional distributed computing.

5.1. The Computing Core and Unit of DCOMP5. Aiming at the computing model in DANET, we intend to promote the computing model in the distributed environment, which expands the concept of the core of distributed computing and defines the role of NCPs as two types according to the available software and hardware:

(i) Computing core (CC): it mainly includes the main program running on the cluster head node, which is responsible for managing the execution process of the entire computing task and abstracted as the application computing core in DCOMP

(ii) Computing unit (CU): it mainly includes the real computing program running on general nodes, responsible for specific computing tasks and abstracted as a computing unit in DCOMP

Different from the traditional distributed computing architecture execution flow, the calculation of the main control program in CC and the calculation program in CU are performed at a different node. The CC may also assume the role of CU in DCOMP. Figure 12 shows the struct of the CC and CU.

When the main program (CC code) at the cluster head node (C1) starts the main program and executes the first calculation program function (xFuction1), C1 actively queries available computing resources (software and hardware) from the local resource database to schedule and allocate computing resources on a general node (Un) according to the request of the CC code. The main program notifies the management program on the general node (Un) of the code and data and starts the computing program (CU code) after offloading the code and data to Un and returns the status to the CC code after completing the calculation task to maintain the data consistency. The CC code continues to execute the second calculation program function (xFuction2). It still needs to apply for resources again and notify the assigned general node (Um) management program to offload the code and data and start the calculation. The CC code continues to execute the calculation process until the entire computing task is completed.

The CC code and the CU code are usually not on the same node in DCOMP, but their roles are interchangeable. The code in traditional distributed computing architecture is usually deployed by the master node to the slave node at one time. The master node controls the program fragments in the slave node to perform related calculations, and the calculation results are summarized in the master node regardless of the current position, resources, and status of the slave node. In the DCOMP, the CC is responsible for maintaining all flow and CU positions and resource information list and updated in time after the execution of the CC code to ensure that all calculation data are correctly accessed to maintain the data consistency.

5.2. The Software Stack of DCOMP. The software stack of DCOMP includes an application software layer, programming framework layer, resource management layer, network communication layer, operating system layer, and device layer, as shown in Figure 13.

The programmable computing framework includes the TCL (Tool Command Language) [77] interpreter and its programming model formed by the runtime environment. It provides a set of available interpretation instructions for programmers to write applications with distributed

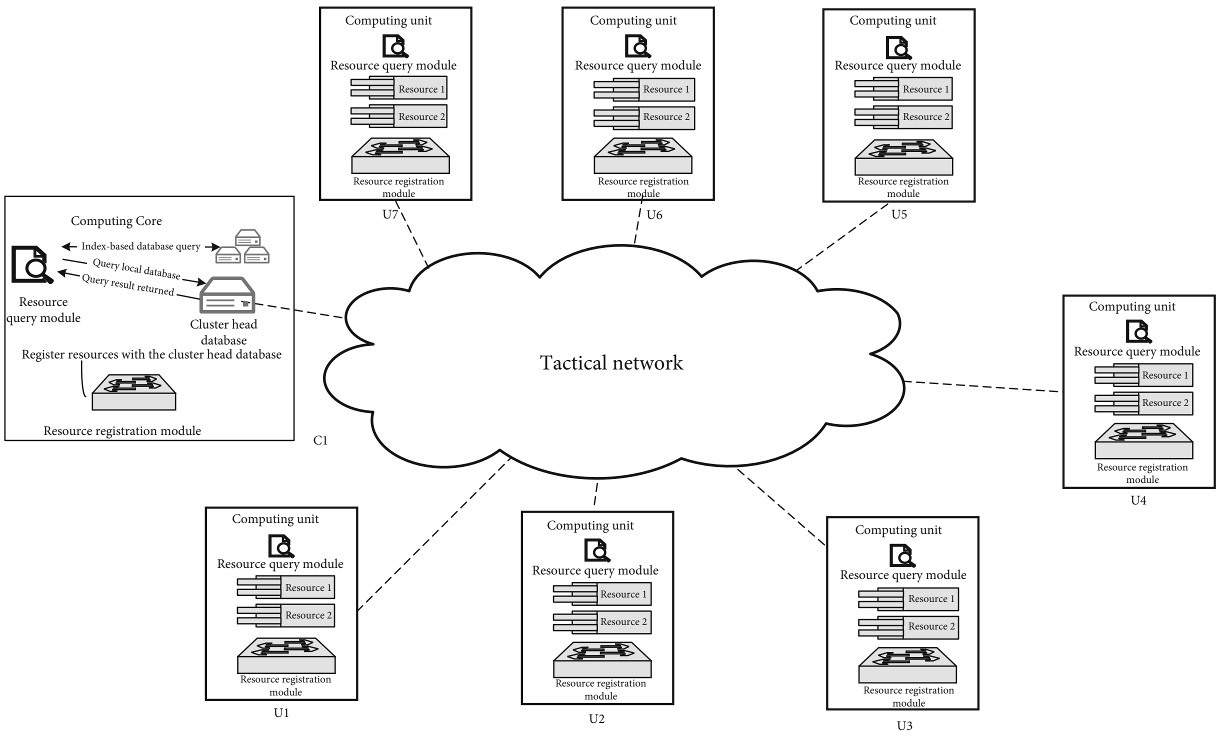


Figure 12: The struct of the computing core and computing unit.

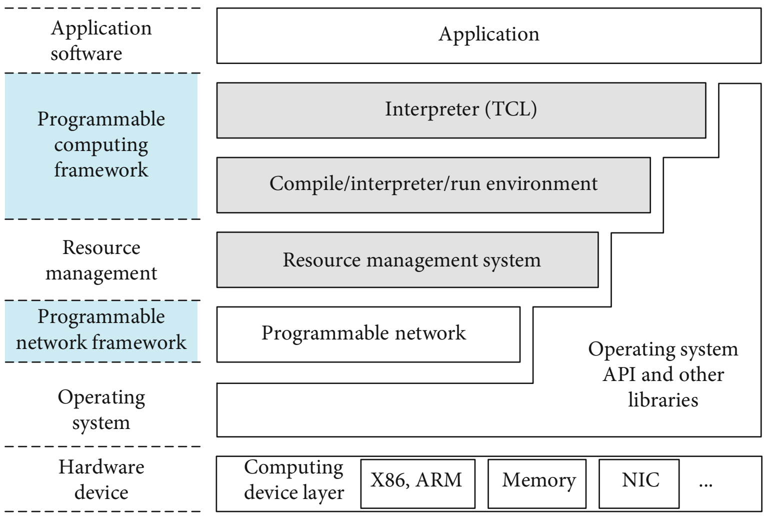

Application software
Programmable computing framework
Resource management
Programmable network framework
Operating system
Hardware device
Application
Interpreter (TCL)
Compile/interpreter/run environment
Resource management system
Programmable network
Operating system API and other libraries
Computing device layer
X86, ARM
Memory
NIC
...

Figure 13: The software stack of DCOMP.

semantics. By using these interpretation instructions explicitly, users can concentrate on program performance optimization, such as the development of parallelism between CC and CU, without paying too much attention to details such as the heterogeneity of underlying resources, dynamic resource binding, and load balancing.

In the programmable network framework, network scheduling and forwarding control of NCPs in DANET is a difficult problem. Programmable network programming model and protocol specifications for interactive use between each node can provide network-aware programming interfaces and scheduling programming interfaces, connectivity and bandwidth utilization awareness, path state threshold calculation and failure analysis, real-time path changes, and other programmable capabilities [78, 79].

6. The Programming Model for DCOMP

Traditional distributed computing environments generally hard code all kinds of algorithms and software to each node and transmit commands with adjustable parameters to call and control the software of each node, which will not be able to meet the requirements of the rapid and flexible call of various resources in future war. Is it feasible to download the

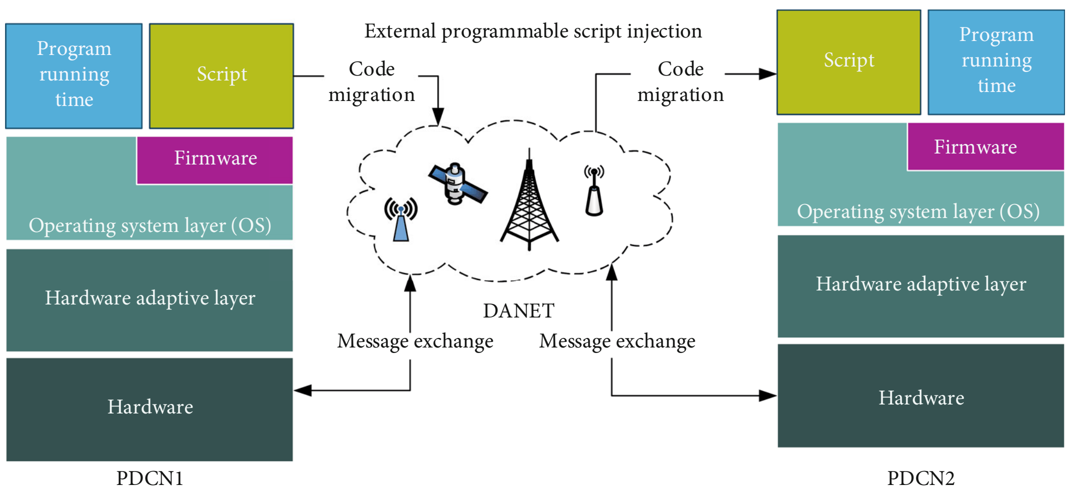


Figure 14: The framework of programmable of PDCN.

compiled executable image to each node? The answer is negative, because most nodes are sometimes inaccessible in physical resources or file transfer at a very high cost. Generally, the energy efficiency of transmitting executable compiled executable images to each node through the network is very low (high communication costs and limited node energy).

In the harsh battlefield environment, it is required that DCOMP be able to provide task computing as an aggregated whole, not just to provide services as a single node, needing to achieve DCOMP nodes that have the ability to be programmable dynamically. This means that an NCP connected to the network at any time will be able to inject instructions into the network to perform a given task. The instructions will call a single node based on the needs of the combat mission, network status, and physical resources.

6.1. *Programmable Computing Model for DCOMP*

6.1.1. Programmable Model. This paper proposes a programmable framework for DCOMP, which attempts to complete the computing tasks by a user-defined way in the dispersed networking environment. We called this kind of node a programmable dispersed computing node (PDCN). The framework of the programmable computing model is shown in Figure 14. The framework can be divided into the hardware layer, hardware adaptive layer (device driver), operating system (OS) layer, programmable software layer, and PDCNs in DCOMP to achieve data communication and computation offloading through DMANT.

(i) Hardware layer includes a wireless transceiver module, various sensors, timers, and other computing hardware resources

(ii) Hardware adaptive layer provides various types of equipment drives

(iii) Operating system layer provides all standard functions and services of the multithread environment required for the programmable software layer. The firmware provides the functions necessary for hardware resource calls, thread/file/communication operations, etc,

(iv) Programmable software layer provides the running time environment for the programmable language

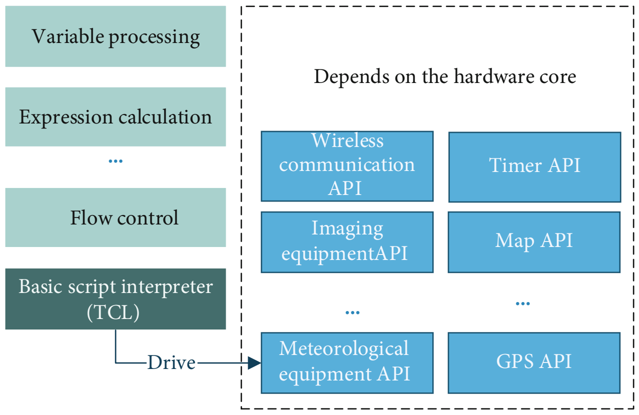

Variable processing
Expression calculation
...
Flow control
Basic script interpreter (TCL)
Drive
Depends on the hardware core
Wireless communication API
Timer API
Imaging equipmentAPI
Map API
...
...
Meteorological equipment API
GPS API

Figure 15: The composition of the programmable language.

6.1.2. Programmable Language. The basic idea of programmability is to make nodes have the ability to be programmable through control scripts, including the scripting language and the corresponding programming model. The scripting language needs to define and implement the appropriate functions/commands in order to use them as building blocks (such as the basic commands of a script). Each of these commands will abstract a specific task from the tactical end node (communication with other nodes, sensors to obtain battlefield data, etc.). The scripting language also needs to construct the syntax of the corresponding control script, including syntax for flow control (such as loops and conditional statements), syntax for variable processing, and syntax for expression calculation. Figure 15 shows the composition of a programmable language.

As we can see from Figure 15, the PDCN first parses the statement of the programmable code and then judges the

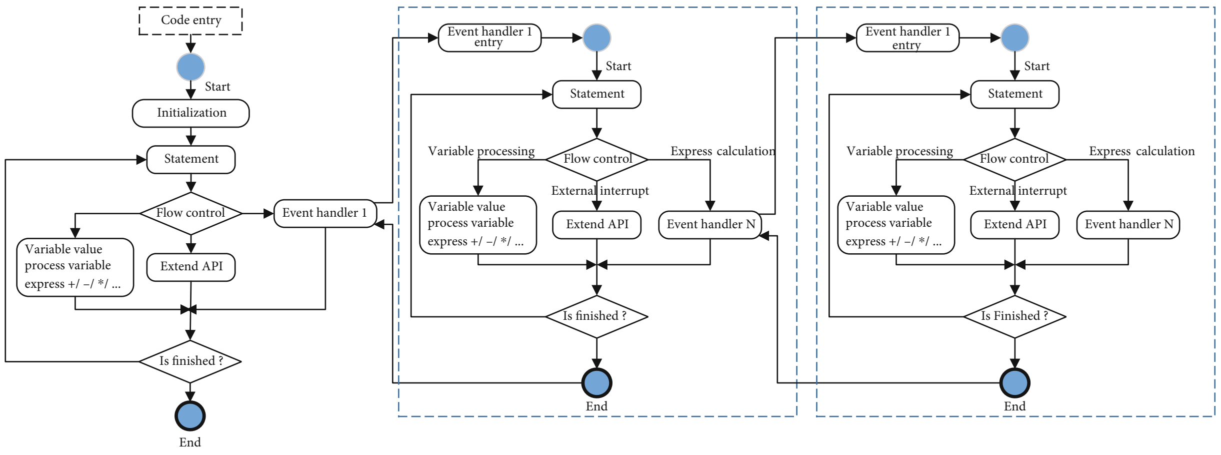


Figure 16: The workflow of the programmable model.

execution flow of the statement. If the statement is an "event handler," then the workflow will enter the function of the "event handler program." If the statement is "variable processing," "expression calculation," or "execute external API," the program will execute the corresponding operations, until the end of the program.

The basic script interpreter can use the open-source syntax interpreter TCL, which provides well extensibility and portability for programmable languages. All basic commands (such as switch, if, while, and other commands) can be defined as the new programmable script syntax using the standard TCL.

According to the model of DANET, the script can be viewed as a state machine affected by external events (like interrupt messages) including network messages from DANET, sensed data, and timer period timeout. The programming model is used to define an event progress, which determines the execution flow based on the current state, and the new event or state will be processed. Figure 16 shows the workflow of the programmable model of a typical PDCN.

6.1.3. Runtime Environment. As important as programmable languages and models is the runtime environment running the script. Different PDCNs provide different resources including different hardware and software resources.

For example, PDCN A has a transceiver and a magnetometer, PDCN B has two transceivers and a camera, and their operating systems are different. The programmable language framework solves the heterogeneity of PDCNs through abstraction and defines and adds arbitrary tasks in the runtime environment through the extended API provided by the abstract module/service interface constructed by the programmable framework.

All devices have fixed interfaces to operate on events through four commands: Query, Act, CreateEventID, and DisposeEventID. Query requires the device to obtain device hardware and software information; Act instructs the device to perform actions (modify some parameters of the device, perform some actions); CreateEventID describes the specific event that the device can generate, provides a name or ID for this event, and waits for the specific event returned with that

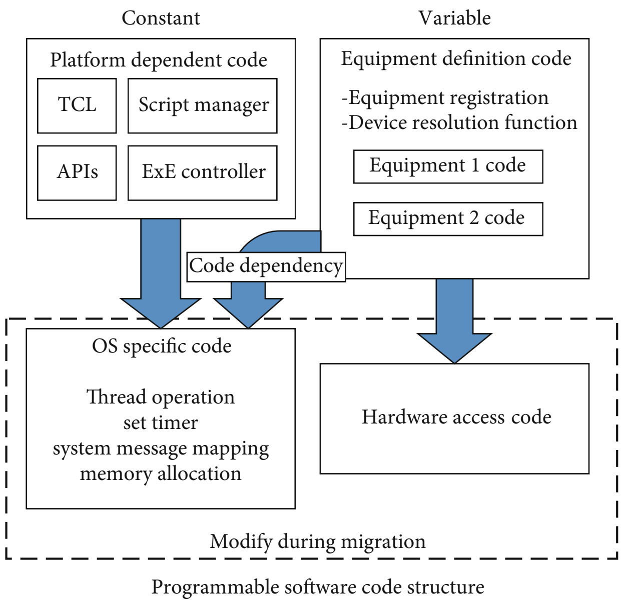


Figure 17: Programmable software code structure.

name in the command; and DisposeEventID unregisters the event and releases the resources.

Even two PDCNs have the same functionality, and the hardware or operating systems may be different. In order to facilitate the migration process, it is necessary to clearly separate the OS layer- and hardware layer-specific code from the solidified code and function definition code. In order to implement the functions defined in programmable software, it needs to identify the code's dependencies with the operating system and the hardware and create associated abstract constructor interfaces. The constructor interface is defined separately in the code file (such as .c file), and developers can easily identify and migrate a code to suit the MANET environment. The operating system needs to support for creating and starting threads/tasks, as well as for defining, publishing, and suspending message queues. In addition, the hardware access code is defined to realize low-level hardware resource call and manage. Figure 17 illustrates the code structure of the programmable software.

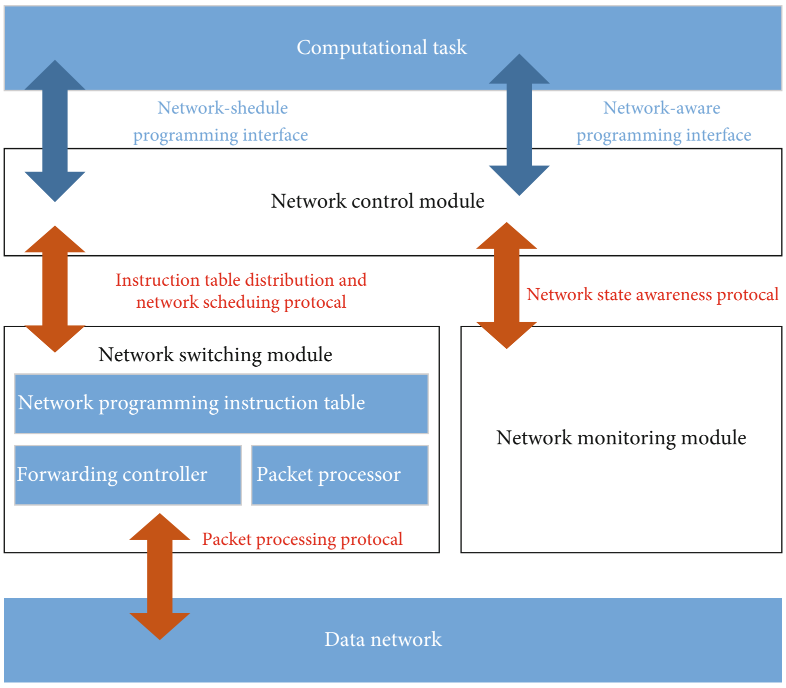

Computational task
Network-shedule programming interface
Network-aware programming interface
Network control module
Instruction table distribution and network scheduling protocal
Network state awareness protocal
Network switching module
Network programming instruction table
Forwarding controller
Packet processor
Network monitoring module
Packet processing protocal
Data network

Figure 18: The components and dataflow of the programmable network model.

6.2. Programmable Network Model for DCOMP. In order to solve the network scheduling and forwarding control of PDCN in DANET, it is necessary to design a programmable network model for DCOMP and a protocol specification for interactive use among entities according to the model. The model provides the network-aware and scheduling programming interface, which supports programmability of connectivity and path bandwidth awareness, state threshold calculation and failure analysis, and real-time path choice. Figure 18 shows the components and dataflow of the programmable network model in DANET.

6.2.1. Network Control Module. The module provides network-aware and network scheduling programming entry and receives the task distribution plan issued by CC, which is combined with the network connectivity state diagram to generate the instruction table and then sent to the relevant forwarding controller and packet processor to implement network scheduling.

6.2.2. Network Switching Module. The module receives the network programming instruction table generated by the network control module and performs data packet processing and forwarding according to the table. It includes the following:

(i) Packet processor filters the data according to the matching rules in the instruction table, then splits, assembles, and adjusts the data packets according to the control commands, and sends them to the forwarding controller

(ii) Forwarding controller forwards the data according to the control commands and instruction table

(iii) Network programming instruction table is a command list consisting of forwarding and packet processing instructions, which includes data match description in each line. The forwarding and packet processing instruction is distributed by the network control module to the network switching module

6.2.3. Network Monitoring Module. The module is responsible for monitoring the status of the component, receiving and sending the statistics and exchanging information, querying the statistical commands sent by the network-aware programmable interface, and communicating between the network control module and network switching module. The monitoring information mainly includes the dataflow, quality and theoretical bandwidth of the channel, and resource occupation of each node of the channel.

6.2.4. Network-Aware Programming Interface. The interface drives the monitoring module to collect information such as the connection status, bandwidth, and dataflow in the network and calculates path bandwidth utilization and status threshold, analyses and optimizes the network path, locates failures, and implements network-aware programming.

6.2.5. Network Schedule Programming Interface. The computational task uses this interface to drive the network control module to generate, publish, and adjust the network programming instruction table according to the network status and the computing task, to realize the real-time dynamic scheduling of the network on demand.

The programmable protocol of DANET is used to regulate the transmission of data, control commands, and monitoring awareness information; the red part is shown in Figure 18.

6.2.6. Instruction Table Distribution and Network Scheduling Protocol. The protocol is used between the network control module and the network switching module to transmit scheduling commands and programming instruction tables, which mainly include the following: collect, update, release, merge, and serialize the instruction table.

6.2.7. Packet Processing Protocol. The protocol is used between the network switching module and data network to transmit and standardize the processing mechanism for messages, including the definition of the structure of the data package format and the packaging/unpacking methods for data packets in the DANET.

6.2.8. Network State Awareness Protocol. The protocol is used between the network control module and the network monitoring module to collect network status, monitor commands, and collect network awareness information, including the network status collection command format, the monitoring data message definition format, the regulation of the flow of messages and commands, and the monitoring data aggregation.

7. Task Awareness and Computing Scheduling of DCOMP for the Tactical Edge

7.1. Discovery and Decomposition Task in the Tactical Edge for DCOMP. In the harsh battlefield environment, the mismatch between the resources of the computing nodes and the requirements of the combat mission will lead to the low utilization rate of the resources. The relationship mapping model between the attributes and the potential requirements of the task should be established, and the task group should be found according to the multiple attributes of the task, to provide the support for the computing in the PDCN.

In order to cope with the sudden increase in resource requirements for combat mission in the harsh battlefield environment of the future tactical edge, it is necessary to offload the computing tasks to the PDCNs in DANET for collaborative computing to reduce the load of a single computing node. The goal of the computation offloading in the nodes of DCOMP is to reduce the resource excessive consumption in a single computing node and the network load on the premise of ensuring the QoS of task quality in the tactical edge. To effectively solve the problem of resource competition and quality QoS degradation caused by the surge of computing tasks, the computation offloading of DCOMP should consider the participation of multiple computing nodes, multiparty collaboration, and multiple factors, which include computing, storage, and the cost of communication resources. In order to minimize the cost and maximize the benefit for DCOMP, it is necessary to build the task awareness and computing scheduling mechanism by considering the constraints of benefit and cost functions. On the premise of satisfying the constraint conditions, the benefit and cost of each computing node in the DANET will be calculated, which can maximize the benefit and performance of the whole calculation platform to complete the task.

Table 3: Combat mission attribute table.

Attributes	Central characteristic
Task time	Time or period to perform a task
Task location	Locations frequently performing operational task
Resources required	Resources frequently needed by task
Resource offset of task requirements	Task-specific operational resources needed

7.1.1. Relationship Mapping Model of Task Attributes and Potential Requirements. There is a relationship between the attributes of most combat tasks on the tactical edge and their potential resource requirements. Due to the large scale of combat tasks and attributes, it is necessary to build a model to simplify the mapping relationship between task attributes and potential requirements to facilitate the discovery of computing tasks with typical central characteristics in multiple dimensions. For example, the frequency of combat location and the specific times have typical central characteristics. Therefore, in addition to the conventional attributes of time, space, and priori knowledge, many can be added to design more behavioural attributes around the centrality of the combat missions. The centrality of the combat missions is shown in Table 3:

It can use the centrality discovery algorithm to obtain the centrality information of the combat mission under multiple attributes from several records and select the central offset or distance of multiple attributes to describe the tactical task.

Firstly, the information such as time period t and location l can be extracted, and then, calculate the center offset of each attribute according to the information. The original spatio-temporal information is projected into a new vector space $d_1(e_1)$, where the coordinates of each point represent the offset of the corresponding center point from it.

The mapping model of the potential requirements and the attributes of combat missions can be described as a nonlinear model, and the multidimensional attributes of combat mission access content, location, time, and traffic extracted at time t can be represented as

$$X_t = (x_1(t), x_2(t), x_3(t), \cdots, x_n(t)), \tag{8}$$

where $x_1(t), x_2(t), x_3(t), \cdots, x_n(t)$ represent multidimensional attributes, respectively.

The potential demand of the combat mission is expressed as the required resource of the combat mission at the next moment $t+1$, expressed as $Y(t)$. The importance of multiple attributes to the potential requirements of a combat mission is

$$A = (a_1, a_2, a_3, \cdots, a_n), \tag{9}$$

where $\mathrm{a}_1, a_2, a_3, \cdots, a_n$ represent the importance of multidimensional attributes, respectively.

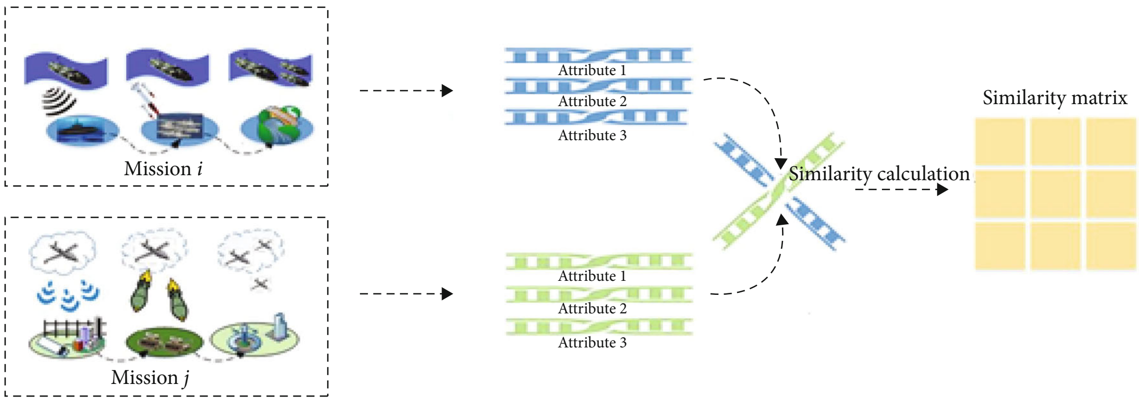


Figure 19: The calculation process of the time series similarity matrix.

The potential relationship between tactical tasks and multiattribute mapping can be described as

$$\begin{aligned} Y(t+1) &= f(a_1x_1(t), a_2x_2(t),\cdots,a_nx_n(t)),\\ Y(t+1) &= f\left((a_1, a_2,\cdots,a_n)^T \bullet (x_1(t), x_2(t),\cdots,x_n(t))\right),\\ Y(t+1) &= f\left(A^T \bullet X\right), \end{aligned} \tag{10}$$

where f is a nonlinear function, which can be approximated as

$$f(x) = \sum^{M} P_i(x)w_i\,(w_i \in R), \tag{11}$$

where $P_i(x)$ is a polynomial function that is generally used as a basis function in models for nonlinear system identification.

It can choose an entropy-based classification algorithm (random forest or decision tree algorithm) to build the relationship between multiple attributes and potential requirements of the combat mission, which can take $X(t) = (x_1(t), x_2(t),\cdots,x_n(t))$ as input and $Y(t+1)$ as output for model training.

The random forest algorithm or decision tree algorithm constructs the decision tree for the extracted training data set and feature subset, and the final classification result is classified by the decision tree voting, as shown in the following formula:

$$H(x) = \arg\left(\max_v\left(\sum^{k}(I(h_i(x)=Y))\right)\right), \tag{12}$$

where $H(x)$ is a combined classification model, h_i is a single decision tree, I is the characteristic function, and Y is the target variable. The split indicator of the decision tree is Gini. The importance of each attribute can be obtained by calculating the mean of the Gini index.

7.1.2. Large-Scale Efficient Similarity Matrix Calculation. The time, location, access content sequence, and traffic of the task attributes with different dimensions and different value ranges need to eliminate the differences between the attributes and characterize the dynamic behaviours. By calculating the similarity of the specific attribute time series of different tasks and eliminating the difference in attribute dimensions, the similarity of the time series containing the dynamics of the combat mission can be obtained to form a similar matrix of task behaviour.

The similarity matrix of combat mission of time series is used to form the similarity matrix between combat missions. Figure 19 shows the calculation process of the time series similarity matrix for combat missions.

The attribute sequence A_i and A_j of the combat mission is extracted from the temporal composition sequence of the combat mission. According to the attribute sequence of the combat mission, the appropriate similarity calculation method can be used to calculate the similarity P_{ij} between the two combat missions. For all combat missions, the similarity under different attributes is calculated in pairs, and the similarity matrix P_S and P_T under multiple attributes of the combat mission can be obtained. P_S and P_T are the space location and time of the combat mission, respectively:

$$\begin{aligned} P_S &= \begin{bmatrix} p_S^{11} & \cdots & p_S^{1m} \\ \vdots & \ddots & \vdots \\ p_S^{m1} & \cdots & p_S^{mm} \end{bmatrix},\\ P_T &= \begin{bmatrix} p_T^{11} & \cdots & p_T^{1n} \\ \vdots & \ddots & \vdots \\ p_T^{n1} & \cdots & p_T^{nn} \end{bmatrix}. \end{aligned} \tag{13}$$

It is assumed that similar combat missions in geographical location will be more similar, while those far away in geographical location will be less similar. According to the geographical position of the combat mission, the combat mission can be divided to calculate the similar matrix preferentially similar in geographical position, which can greatly reduce the amount of calculation of the similarity matrix. The specific algorithm is shown in Figure 20.

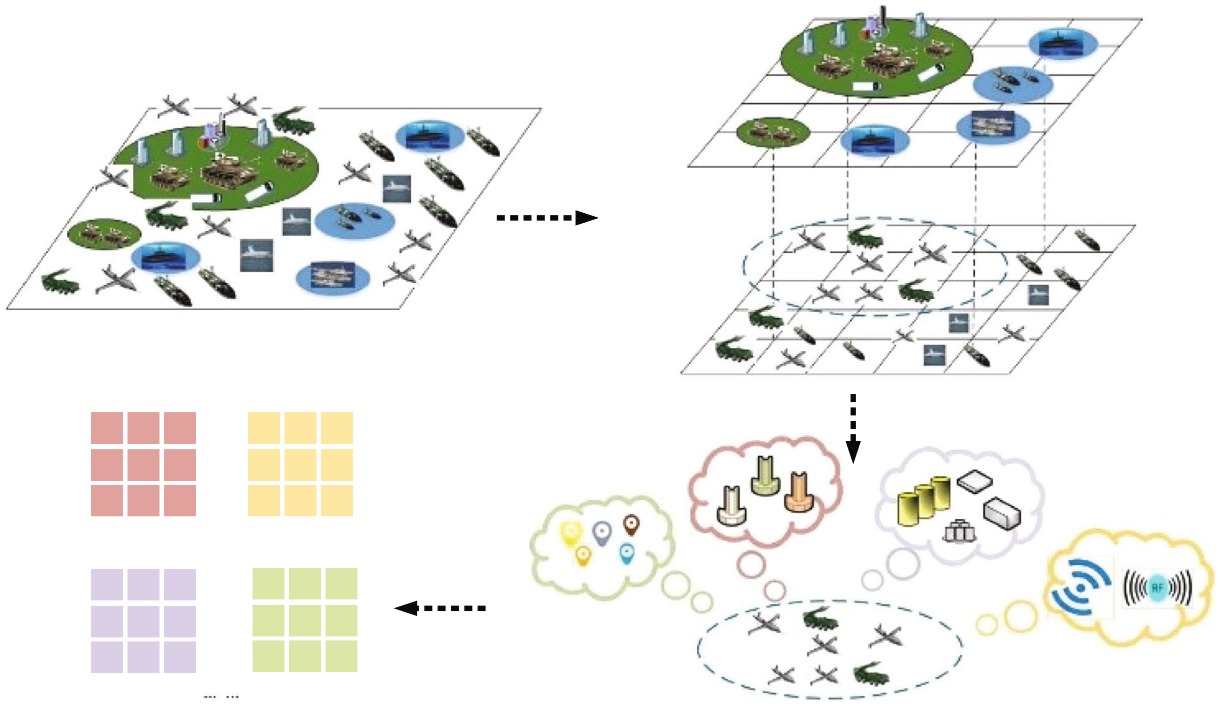


Figure 20: Large-scale efficient similarity matrix calculation method.

7.1.3. Resource Discovery Based on Weighted Similarity Matrix Fusion. After obtaining the mapping model of combat mission attributes and potential resource requirements, in order to eliminate the parameter dimension differences between the attributes and quickly find the combat mission group, it is necessary to fuse the similarity matrices of multiple attributes and segment the fused similarity matrix.

The similarity matrix fusion method can effectively improve the system's resolution ability by fusing similarity matrices with different attributes, while retaining the information of the original similarity matrix to the greatest extent. Therefore, the sum of the distance between the fused similarity matrix and the original similarity matrix should be the smallest, as shown in Figure 21.

Set P to be the similarity matrix after fusion, the weights α, β, γ reflect the behaviour attribute, and S, T, I reflect the importance of the potential needs of tactical task. The goal of calculation of the similarity matrix is to minimize the sum of the distances between P and the original similarity matrix P_S, P_T, P_I of multiple attributes. The specific model is described as follows:

$$\min \alpha\|P - P_I\|_2^2 + \beta\|P - P_T\|_2^2 + \gamma\|P - P_S\|_2^2 + \lambda\|P\|_1, \quad (14)$$

$$\text{s.t.}\, \alpha + \beta + \gamma = 1, \quad (15)$$

$$\alpha \geq 0, \beta \geq 0, \gamma \geq 0, \quad (16)$$

$$1 \geq P_{ij} \geq 0, i, j \in [1, N]. \quad (17)$$

The second norm term in formula (14) indicates the distance of the similarity matrix of the corresponding data

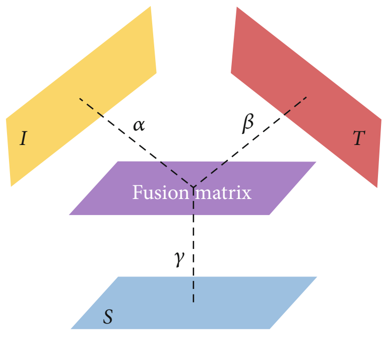


Figure 21: Schematic diagram of fusion similarity matrix calculation.

relative to the fusion result matrix, and λ is the sparse rule operator. Formula (15) represents the normalized weights of the three types of data relative to the final fusion result. Formula (16) limits the weights of various types of data in the optimization formula and the range of sparse rule operator weights. Formula (17) shows the value range of each element in the fusion result matrix P. $\lambda\|P\|_1$ is a continuous convex function, and $\alpha\|P - P_I\|_2^2 + \beta\|P - P_T\|_2^2$ is a continuously differentiable function and satisfies the Lipschitz conditions. The Lipschitz continuous gradient is $L(f)$. Set $f(P) = \alpha\|P - P_I\|_2^2 + \beta\|P - P_T\|_2^2$ and $g(s) = \lambda\|P\|_1$, then

$$\left\|\nabla f\left(P'\right) - \nabla f(P)\right\|_2^2 \leq L(f)\|P' - P\|_2^2. \quad (18)$$

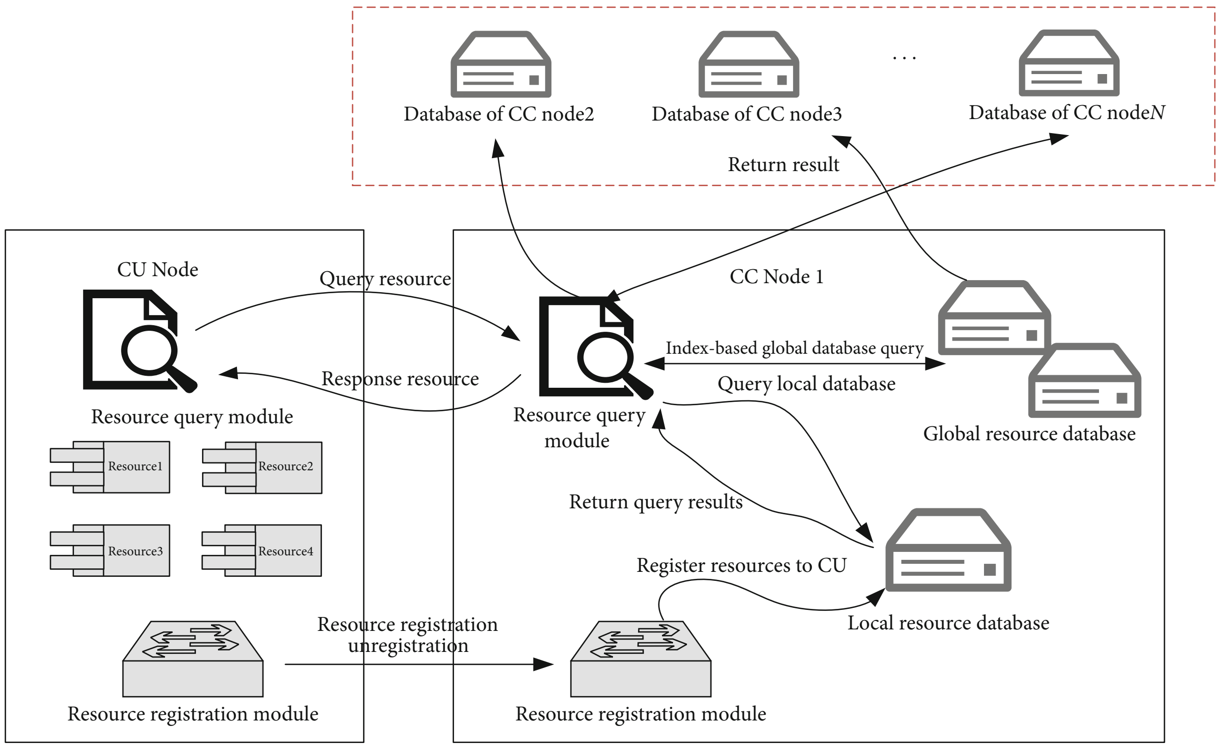


Figure 22: Structure diagram of a resource-aware process.

For such problems, one can use a fast-iterative shrinkage-thread should algorithm (FISTA) for calculation at S as follows:

$$P(S)+<\nabla f(S), S'-S>+\frac{L}{2}\|S'-S\|^2=\frac{L}{2}\left\|S'-S+\frac{1}{L}\nabla f(S)\right\|^2+\text{const}, \tag{19}$$

where L is the Lipschitz constant, and the available iteration value by ignoring the constants is

$$S_{k+1}=\text{argmin}_s\left\{\frac{L}{2}\left\|S-\left(S_k-\frac{1}{L}\nabla f(S_k)\right)\right\|_2^2+\lambda\|S\|_1\right\}. \tag{20}$$

The initial value is given according to the actual situation during the calculation and iterates according to the above formula until convergence, to obtain the final similarity matrix after fusion.

In the process of calculation and fusion of similar matrices, various attributes of combat missions have been included and the information of requirements of potential combat mission and activity locations can be obtained, providing resource scheduling and matching for DCOMP.

7.2. Real-Time Perception and Management of Resources for DCOMP. At the tactical edge, various wireless links are widely used and caused the different transmission qualities. Obviously, the DANET is a heterogeneous network with a mixed network of broadband and narrowband in a complex battlefield environment, which can be subject to various kinds of aggression such as battle and electronic interference impact. In DANET, the resource management is a dynamic process that can be dynamically perceived and discovered. The resources mainly include various resources of different network bandwidths, services, and hardware in DANET. As the topology of DANET changes dynamically, the resource-aware algorithm should adapt to the plug and play of available resources, and the resources of nodes can be perceived in time by other nodes in the entire network of DCOMP. The purpose of resource awareness is when a resource is provided by a PDCN in DANET, other nodes in the network can identify the resource, and when the resource is revoked, other nodes can also know that the resource is out of date and no longer provided.

Table 4: Data package structure of resource description information.

Serial number	Packet definition	Message description
1	ResName	Name of the resource
2	ResId	Category ID of the resource
3	ResHostId	Host address of the resource provider
4	ResExpTime	Expiration time of the resource
5	ResPosition	Location of the resource provider
6	ResDes	Description of the resource function
7	ResQuality	Resource quality

By considering the limitations of nodes on memory, CPU, storage, and other resources, the algorithm running

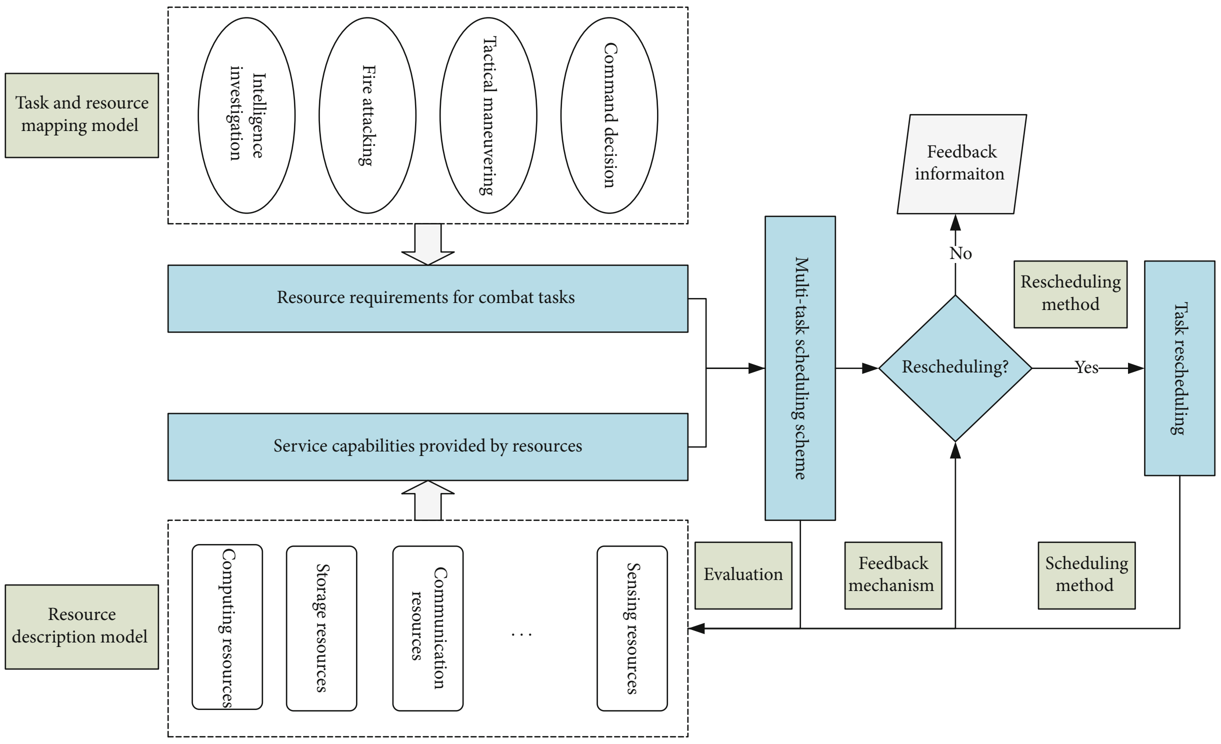


FIGURE 23: Task-coordinated planning and scheduling mechanism.

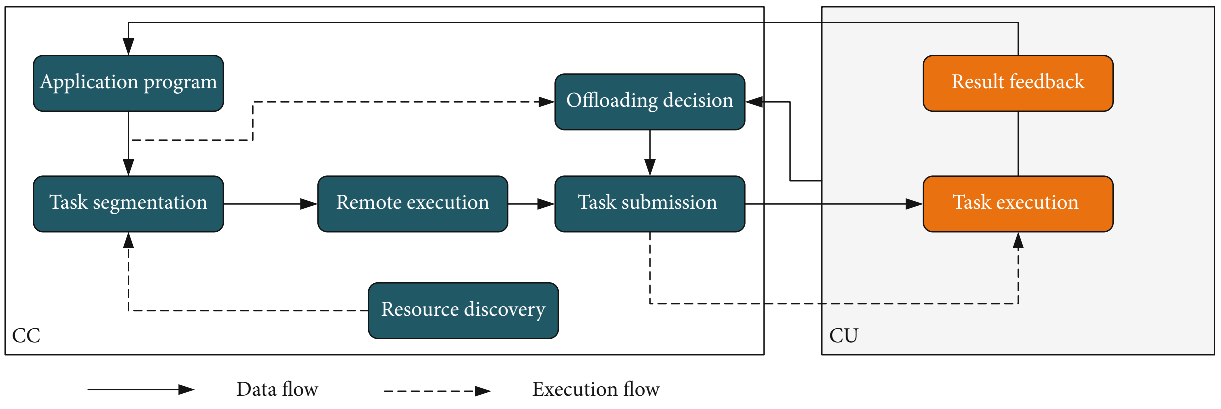


FIGURE 24: The workflow of computation offloading.

on PDCN should not be too complicated. According to the different statuses of nodes in DANET, the protocol can be divided into resource awareness modules, client agents on the common node (CU node), and directory agents on cluster head nodes (CC node). Figure 22 shows the structure diagram of a resource-aware process.

As shown in Figure 22, the CU node is generally responsible for the calculation of various tasks, which needs to register its own fixed resources (camera, storage, communication resources, and various sensors) to the CC node through the resource registration module. Various real-time changing resources (real-time CPU/memory usage, network bandwidth) can be queried/responded by the CC node through the resource query module. The CC node is a cluster head node responsible for managing the resources of the CU nodes and collaborative communicating and computing with external CC nodes. Generally, a CC node has a local resource database, a resource registration module, a resource query module, and a global resource database. The resource registration module obtains the resource data of the CU nodes within the range of the CC node and stores in the local resource database. A CC node regularly manages and backs up the computing resource data of the entire DANET by collecting resource data from other CC nodes. Other CC nodes also can obtain the resource data from the resource query mode of this CC node. The resource database of the CC node mainly stores various resource description data including the name, node address, and some related descriptions of the resource. The resource description information in the traditional SDP (Service Discovery Protocol) only contains the resource name and the address of the resource provider [80]. The resource description of DANET should contain the information such as function description, resource category, provider identification, network environment parameters, performance state parameters, and location. Therefore, when multiple resources can meet task needs in terms of functions, the optimal node can be selected based on the information to perform the computing task. For a DANET, resource description information needs to include the following aspects: name, category ID, host address, expiration time, location, resource function description, and quality information of the resource. Therefore, resource description information is the basis of the resource aware protocol, and the carrier of resource description information transmission in the protocol is the definition of the data package. Table 4 shows the data package structure of resource description information.

7.3. Resource Planning and Scheduling for DCOMP. In the complex and changeable battlefield environment, according to the objective, determination, and mission of the combat mission, starting from the integration support of information acquisition, integrated processing, and strike command communication, it is necessary to perform reasonable metatask scheduling according to the status and resource capabilities of DANET and achieve reasonable sharing of resources and mutual cooperation for completing the combat mission. The scheduling problem of computing resources is a mixed constraint planning problem that involves both time and resource allocation. Based on the resource description, task, and resource mapping model, the formal definition is established from the ability of resources to meet mission requirements and the priorities of the defined process and integrity constraints.

In order to realize the dynamic optimal scheduling and conflict avoidance of computing tasks and resources, reduce the time complexity of the whole network resource collaborative processing and load of resource providers and improve the efficiency and robustness of DCOMP, to achieve the goal of global optimization and comprehensive combat mission execution successfully. The collaborative planning and scheduling of computing tasks are shown in Figure 23.

Computing task planning and scheduling are determined based on the characteristics of combat mission requirements and the resource characteristics. Different computing tasks have different requirements for resources, and different resources have different capabilities to meet the task, which leads to multitask planning requirements. In multitasking planning and scheduling, the satisfaction of task requirements and resource consumption determine the comprehensive efficiency of a multitasking system. The satisfaction of task requirements has a positive impact on comprehensive efficiency, and excessive resource consumption has a negative impact. According to the parameters of computing task and resource provider, it is necessary to define planning objectives and related constraints and establish a multitasking planning model. According to the comprehensive efficiency and its changing rule of "task number" and "resource number" in different application scenarios, the optimal ratio of "task number" and "resource number" can be obtained, and the relative optimal multitasking arrangement scheme needs to be established.

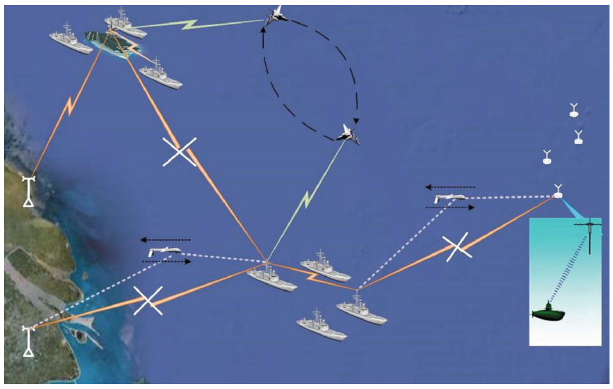


Figure 25: A typical tactical edge combat scenario.

In the actual task execution process, the possible problems such as adding, deleting, and changing tasks and running faults may occur frequently, and a new task scheduling plan needs to be recalculated according to the changes and the original task planning.

7.4. Computation Offloading for DCOMP. The calculation, storage, and battery capacity of a single computing node is very limited, but the tasks that need to be processed on the node are becoming more and more complicated in the tactical edge. There is a contradiction between the large amount of computing resources consumed by computing intensive applications and the limited resources of a single computing node. In order to solve the contradiction between the limited service resources and the unlimited computing task requirements, the task needs to be distributed to the node at DANET for computational offloading [81]. Computation offloading is mainly performed on the CC nodes, which can offload some computing tasks to the CU nodes on DANET. The nodes performing computation offloading not only need to send computing tasks and receive computational results but also need to execute computing tasks, to reduce response delays for time-intensive tasks, which is different from the traditional computing paradigm of MEC, FC, etc. The major steps include computing node discovery, task segmentation, offloading decision, task submission, remote task execution, and calculation result feedback. The traditional computation offloading uses virtual machine migration to deploy the entire application to the service platform for execution, requires high network bandwidth, and results in low efficiency, which is not suitable for the environments of complex, highly dynamic, and weakly connected at tactical edges which is not suitable for complex and highly dynamic environments, leading to weak connection at tactical edges. [82]. The process of computation offloading is shown in Figure 24.

(i) *Resource Discovery*. Find collaborative computing nodes that can perform computing tasks in the current DANET. The computing node can be a high-performance computer located in a remote

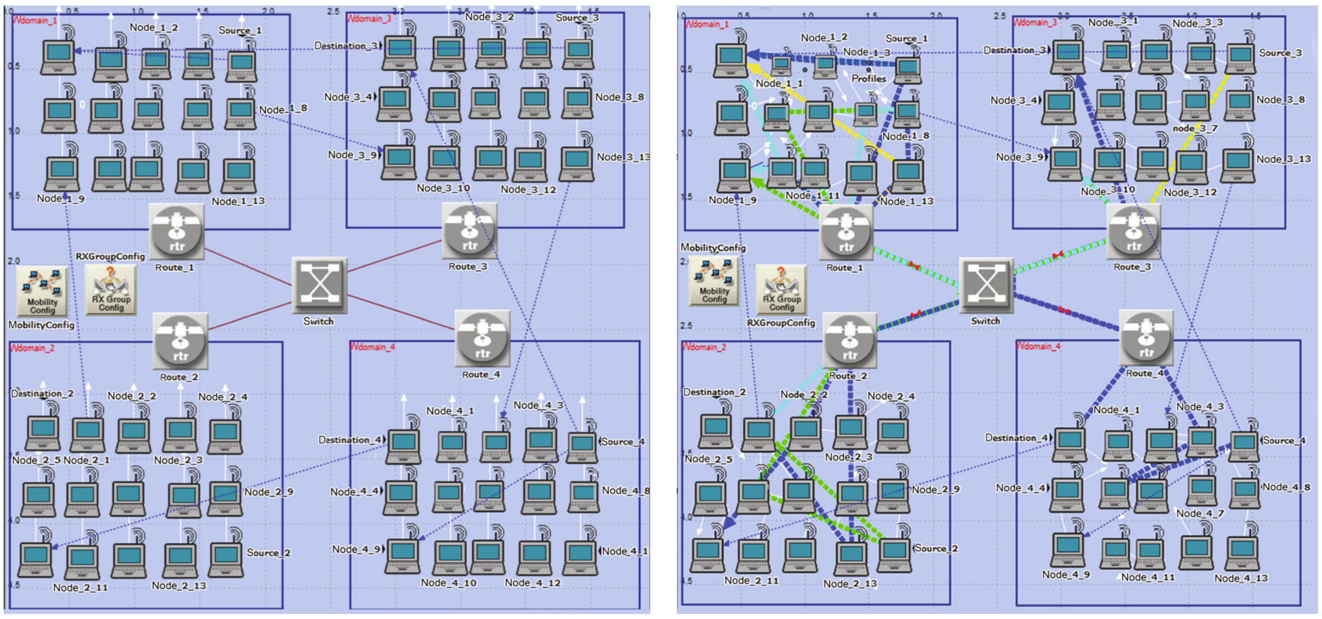

(a) Network model (b) Routing path

FIGURE 26: The OPNET network model of DANET.

data center or a handheld terminal with limited computing ability

(ii) *Task Segmentation*. In the preparation phase of computation offloading, the results of the segment of computational tasks have a significant impact on the performance of computation offloading. The granularity of task segmentation can be divided into method, module, and thread levels based on the discovered resources

(iii) *Offloading Decision*. Deciding whether to perform computation offloading to a CU node mainly depends on the overhead of network delay, the energy consumption, the current status of the CU node, and the computing task

(iv) *Remote Execution and Task Submission*. This module is responsible for packaging the programmable code (described in Section 4) and data that needed to be calculated and sent to the CU node

(v) *Task Execution*: Execute the programmable code offloaded to the CU node according to the programmable model described in Section 5

(vi) *Result Feedback*. The calculation result feedback is the last step of the computation offloading. After the CU node feeds back the calculation results to the CC node, the network connection between the CU node and the CC node is released and the computation offloading is finished

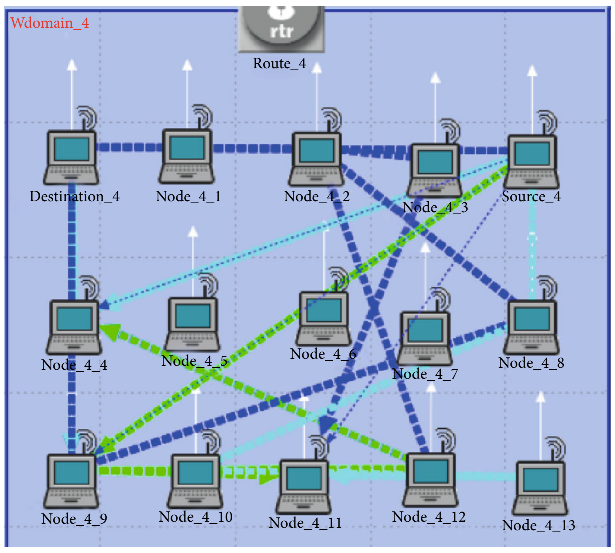
All the resources of computing nodes are deployed in a distributed manner on the DANET, and each computing node can upload calculation results anytime and anywhere and at any movement speed [83].

8. The Application Scenario Analysis of DCOMP in Future Wars

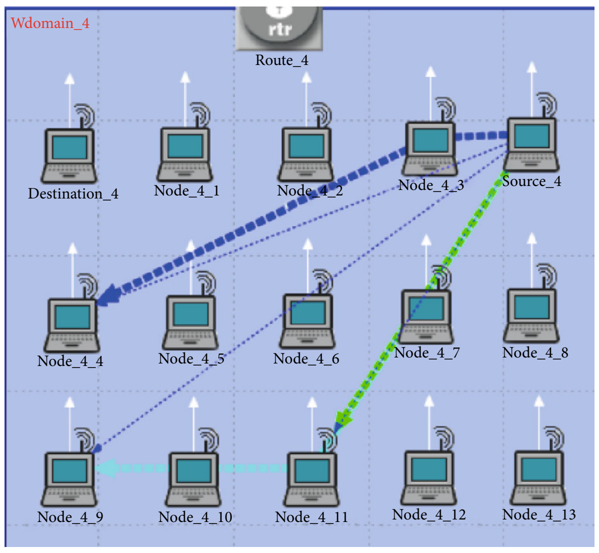
In the future, military operations such as evacuation, peacekeeping, and counter-terrorism will be far away from the homeland, and the operation process will lack the support of infrastructure with strong communication and data processing capabilities, which need to rely on a network that can handle dynamic tasks. The traditional computing paradigms such as cloud computing, edge computing, and wireless sensor networks cannot satisfy this kind of far way form command center, which network throughput is severely limited and the battlefield situation changes rapidly.

Figure 25 shows a typical battle scenario of the tactical edge away from the homeland, which consists of aircraft carriers, frigates, destroyers, fighter jets, UAVs, etc. The UAVs are used for reconnaissance, fighter jets for strikes, aircraft carriers for commanding operations, and other warships for escorts.

For example, when an UAV carries out reconnaissance, the images of the enemy are collected, which need to be analysed and processed. At this time, the data link received enemy interference and the communication between the UAV and the carrier was interrupted. However, the communication link between UAVs and fighter jets is still available. The DANET can be immediately established between the UAVs and fighter jets, which will distribute the collected image and offload the image processing program to the PDCN for calculation, and the calculated results will be sent to the UAV or other combat units for decision-making. The role of DCOMP is to quickly organize these combat units due to the lack of computing infrastructure support to conduct various real-time task calculations at the tactical edge. Therefore, DCOMP will play a very import role in future wars, and the large-scale application of DCOMP will change the way



(a) Traditional protocol



(b) Improved protocol

Figure 27: Comparison of routing calculation results between the traditional and improved protocols.

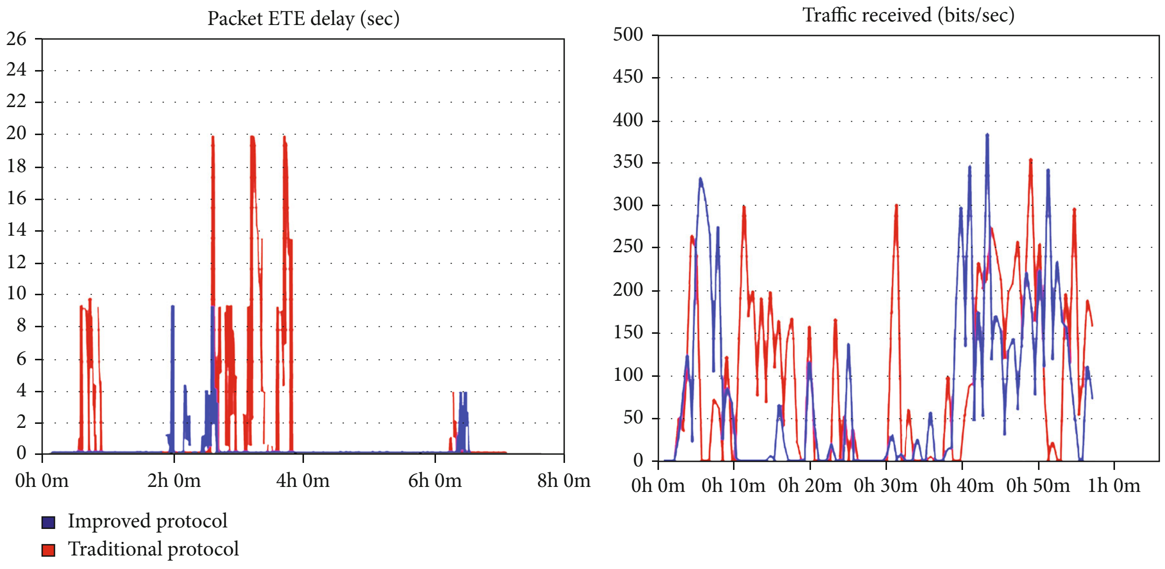


Figure 28: Simulation result of packet ETE delay and traffic received.

and even the course of warfare. In order to verify the performance of the DANET proposed in this paper, we designed a network model to simulate the communication progress of DCOMP at the tactical environment by OPNET network simulation software, as shown in Figure 26.

The network model constructed in Figure 26(a) contains 4 subnets with 15 PDCNs and a router using the AODV or OLSR routing protocol connected by a switch. The PDCN moves within a certain range at the tactical edge to achieve collaborative computing and data sharing with DANET. Since the AODV protocol is an on-demand protocol, when creating a routing table, different time sequences for service requests can be considered and different transmission paths can be established. As can be seen from Figure 26(b), when the existing link in the network is under heavy load and receive the routing request again, the link load status of the existing traffic should be considered. Therefore, the routing protocol should avoid the overloaded link and choose the “idle” or “heavy” path as much as possible to forward data packets. Figure 27 shows the simulation result of the routing path in a subnet using the traditional protocol and the improved protocol algorithm proposed in this paper.

As can be seen from Figure 27, it can be easily found that with the standard protocol, the link load is already heavy and there are still multiple routing requests whose final data forwarding still passes through the link, which will easily cause the loss of a large number of data packets and cause large delay of the network. The statistics of package end-to-end delay and traffic received between the traditional and improved protocols is shown in Figure 28.

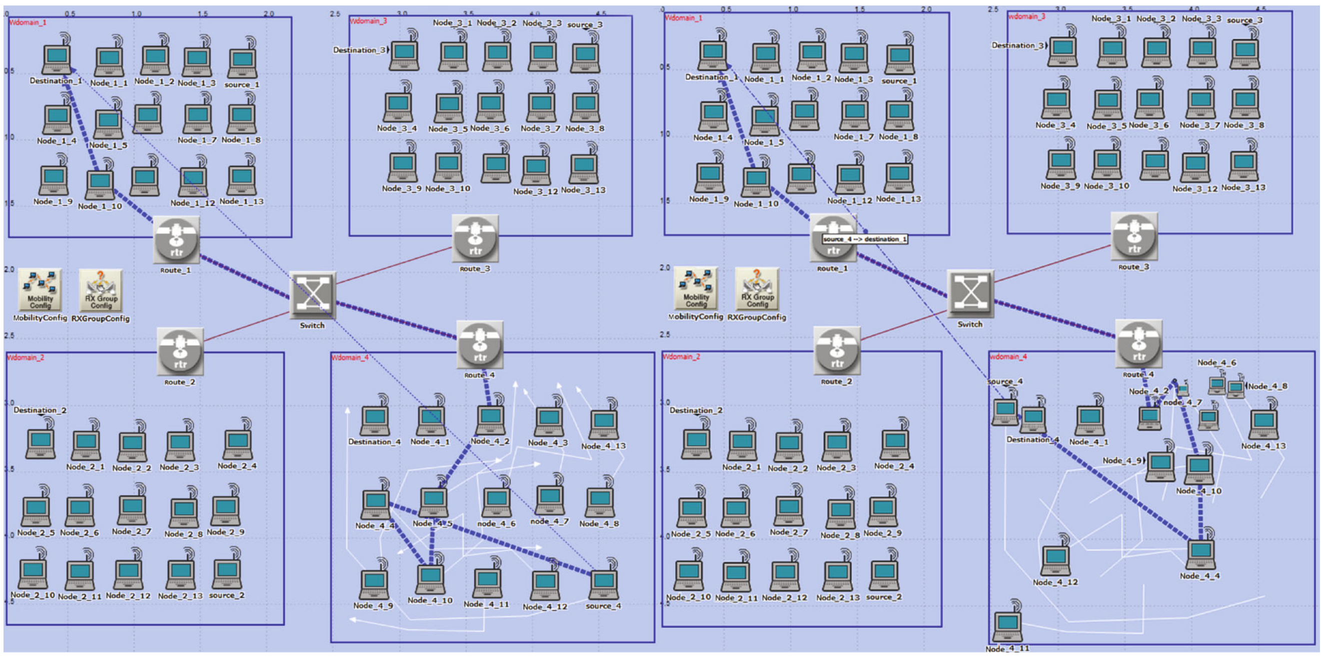


FIGURE 29: Simulation result of fast movement of the computing node.

As can be seen from Figure 28, the packet ETE delay of the improved algorithm is much lower than that of the traditional algorithm and the package of traffic received is not much difference.

At the tactical edge, the speed and path of each computing node are different, which has a great impact on the selection of network routing, throughput, and packet loss rate. With the movement of nodes, the nodes that could not be connected may become connected due to the change of relative position and the topology of the whole network will change with the movement of each node. As shown in Figure 29, which is the schematic diagram of network topology after simulation for 0 s and 30s, with the improved AODV protocol, the link cost between nodes will be changed due to the change of topology structure, and the routing path of services will also change accordingly.

It can be seen from the simulation that the improved protocol can greatly reduce the network delay and improve the network performance, providing a good network environment support for DCOMP. Further, we will perform a large-scale experiment of DCOMP to substantiate our findings and strive to be able to be applied in practical engineering and on a real environment.

9. Conclusion and Challenges in the Future

The rapid transformation of warfare, the continuous upgrading of weapons, and the diversity of combat missions have led to the continuous change of the battlefield network environment, and new requirements have been imposed on environmental elements such as computing, network, and command. We investigated in detail a range of the main technical concepts, mechanisms, paradigms, and important features of DCOMP to meet the requirements and development trends of future battlefield computing and communications. In this article, we have proposed the architecture for DCOMP, a network model called DANET, and the programming model and language for DCOMP. We have also presented methods of task awareness and computing scheduling for the tactical edge in a harsh battlefield environment. Moreover, we have discussed a typical scenario and a vision of DCOMP for the tactical edge in future wars.

As a new computing paradigm, the research on DCOMP in the academia has just started, and the DCOMP has certain advantages over the traditional computing paradigm such as MCC, FC, and MEC. However, limited by the complicated tactical edge network environment which caused discontinuous communication between devices, simulation in real scenes is somewhat difficult and complex. There are still many problems and challenges that need to be studied in the future:

(i) The development of highly dynamic and programmable network protocols for DCOMP should be accelerated as soon as possible to enable it to adapt to weakly connected, highly dynamic, and vulnerable environments at the tactical edge to win the future wars

(ii) Accelerate the research on the technology of programmable computing nodes, programmable networks, and computation offloading. According to different application requirements and network conditions, it is possible to develop a migration solution for nodes adapted to various types of equipment and networks in DCOMP. In addition, it is necessary to establish relevant stands and upgrade existing equipment to meet the requirements of DCOMP

(iii) Many current applications used in the tactical edge should be modified to be adapted for DCOMP, such as tactical target recognition and tracking, target damage analysis, and trajectory analysis

(iv) It is a great challenge to find the right balance between data transmission and computation resources in the degenerate network, which requires in-depth study of the optimal control strategy between real-time network state, dispersed computing resources, and computing tasks

(v) How to ensure the quality of service (Qos), the efficiency of computation, and the security of data is the key research direction in the future under the condition of limited and variable bandwidth and highly dispersed heterogeneous computing resources

(vi) Coded computing is a recent technique that will enable optimal trade-offs between computation load, communication load, and computation latency due to stragglers in DCOMP [63], designing joint task scheduling and coded computing in order to leverage trade-offs between computation, communication, and latency, which is an important aspect in DCOMP

The tactical edge is generally far from the homeland, the nodes of DCOMP are generally powered by batteries, and the battery life is limited. Therefore, the technology for energy consumption management of DCOMP will also need to be studied as a key direction in the future. In the future, the technology of DCOMP can be applied not only in military fields at the tactical edge but also in civil fields such as fire rescue, flood relief, and environmental monitoring.

Data Availability

All data, models, and codes generated or used during the study appear in the submitted article.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This work was supported by the National Defense Basic Scientific Research Program of China under Grants JCKY2019210B005, JCKY2018204B025, and JCKY2017204 B011, the Key Scientific Project Program of National Defense of China under Grant ZQ2019D20401, the Chinese Defense Advanced Research Program of Science and Technology under Grant 41401030301, and the Open Program of National Engineering Laboratory for Modeling and Emulation in E-Government (item number MEL-20-02).

References

[1] L. Yun-Tung, "A unified service description for the global information grid," *CrossTalk*, vol. 20, no. 4, pp. 23–26, 2007.

[2] Z. Guo-Hong, "Architecture of combat clouds," *Journal of Command and Control*, vol. 1, no. 3, pp. 292–295, 2015.

[3] J. George, C. Chen, R. Stoleru, G. G. Xie, T. Sookoor, and D. Bruno, "Hadoop MapReduce for tactical clouds," in *2014 IEEE 3rd International Conference on Cloud Networking (CloudNet)*, pp. 320–326, Luxembourg, Luxembourg, 2014.

[4] A. E. Conway, M. Wang, E. Ljuca, and P. D. Lebling, "A Dynamic Transport Overlay System for Mission-Oriented Dispersed Computing Over IoBT," in *MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM)*, pp. 815–820, Norfolk, VA, USA, 2019.

[5] M. R. Schurgot, M. Wang, A. E. Conway, L. G. Greenwald, and P. D. Lebling, "A dispersed computing architecture for resource-centric computation and communication," *IEEE Communications Magazine*, vol. 57, no. 7, pp. 13–19, 2019.

[6] S. Guo, J. Liu, Y. Yang, B. Xiao, and Z. Li, "Energy-efficient dynamic computation offloading and cooperative task scheduling in mobile cloud computing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 2, pp. 319–333, 2019.

[7] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2016.

[8] M. Jo, T. Maksymyuk, B. Strykhalyuk, and C.-H. Cho, "Device-to-device-based heterogeneous radio access network architecture for mobile cloud computing," *IEEE Wireless Communications*, vol. 22, no. 3, pp. 50–58, 2015.

[9] D. Han, W. Chen, B. Bai, and Y. Fang, "Offloading optimization and bottleneck analysis for mobile cloud computing," *IEEE Transactions on Communications*, vol. 67, no. 9, pp. 6153–6167, 2019.

[10] Y. Miao, G. Wu, M. Li, A. Ghoneim, M. Al-Rakhami, and M. S. Hossain, "Intelligent task prediction and computation offloading based on mobile-edge cloud computing," *Future Generation Computer Systems*, vol. 102, pp. 925–931, 2020.

[11] M. R. Rahimi, J. Ren, C. H. Liu, A. V. Vasilakos, and N. Venkatasubramanian, "Mobile cloud computing: a survey, state of art and future directions," *Mobile Networks and Applications*, vol. 19, no. 2, pp. 133–143, 2014.

[12] D. Tychalas and H. Karatza, "A scheduling algorithm for a fog computing system with bag-of-tasks jobs: simulation and performance evaluation," *Simulation Modelling Practice and Theory*, vol. 98, article 101982, 2020.

[13] A. V. Dastjerdi and R. Buyya, "Fog computing: helping the Internet of things realize its potential," *Computer*, vol. 49, no. 8, pp. 112–116, 2016.

[14] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, pp. 27–32, 2014.

[15] M. Peng, S. Yan, K. Zhang, and C. Wang, "Fog-computing-based radio access networks: issues and challenges," *IEEE Network*, vol. 30, no. 4, pp. 46–53, 2016.

[16] K. Bilal, O. Khalid, A. Erbad, and S. U. Khan, "Potentials, trends, and prospects in edge technologies: fog, cloudlet, mobile edge, and micro data centers," *Computer Networks*, vol. 130, pp. 94–120, 2018.

[17] X. Gao, X. Huang, S. Bian, Z. Shao, and Y. Yang, "PORA: predictive offloading and resource allocation in dynamic fog computing systems," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 72–87, 2020.

[18] Q. Wu, H. Ge, H. Liu, Q. Fan, Z. Li, and Z. Wang, "A task offloading scheme in vehicular fog and cloud computing system," *IEEE Access*, vol. 8, pp. 1173–1184, 2020.

[19] D. Zeng, L. Gu, S. Guo, Z. Cheng, and S. Yu, "Joint optimization of task scheduling and image placement in fog computing supported software-defined embedded system," *IEEE Transactions on Computers*, vol. 65, no. 12, pp. 3702–3712, 2016.

[20] L. F. Bittencourt, J. Diaz-Montes, R. Buyya, O. F. Rana, and M. Parashar, "Mobility-aware application scheduling in fog computing," *IEEE Cloud Computing*, vol. 4, no. 2, pp. 26–35, 2017.

[21] O. Osanaiye, S. Chen, Z. Yan, R. Lu, K.-K. R. Choo, and M. Dlodlo, "From cloud to fog computing: a review and a conceptual live VM migration framework," *IEEE Access*, vol. 5, pp. 8284–8300, 2017.

[22] M. Mukherjee, L. Shu, and D. Wang, "Survey of fog computing: fundamental, network applications, and research challenges," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 1826–1857, 2018.

[23] H. Elazhary, "Internet of things (IoT), mobile cloud, cloudlet, mobile IoT, IoT cloud, fog, mobile edge, and edge emerging computing paradigms: disambiguation and research directions," *Journal of Network and Computer Applications*, vol. 128, pp. 105–140, 2019.

[24] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[25] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: the communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[26] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590–3605, 2016.

[27] P. Mach and Z. Becvar, "Mobile edge computing: a survey on architecture and computation offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[28] T. Taleb, S. Dutta, A. Ksentini, M. Iqbal, and H. Flinck, "Mobile edge computing potential in making cities smarter," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 38–43, 2017.

[29] X. Xu, B. Shen, X. Yin et al., "Edge server quantification and placement for offloading social media services in industrial cognitive IoV," *IEEE Transactions on Industrial Informatics*, p. 1, 2020.

[30] A. Zhou, S. Wang, S. Wan, and L. Qi, "LMM: latency-aware micro-service mashup in mobile edge computing environment," *Neural Computing and Applications*, vol. 32, no. 19, pp. 15411–15425, 2020.

[31] Y. Cao, H. Song, O. Kaiwartya et al., "Mobile edge computing for big-data-enabled electric vehicle charging," *IEEE Communications Magazine*, vol. 56, no. 3, pp. 150–156, 2018.

[32] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: a survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, 2018.

[33] A. K. Malhi, S. Batra, and H. S. Pannu, "Security of vehicular ad-hoc networks: a comprehensive survey," *Computers & Security*, vol. 89, article 101664, 2020.

[34] W. Kiess and M. Mauve, "A survey on real-world implementations of mobile ad-hoc networks," *Ad Hoc Networks*, vol. 5, no. 3, pp. 324–339, 2007.

[35] L. Hanzo and R. Tafazolli, "A survey of QoS routing solutions for mobile ad hoc networks," *IEEE Communications Surveys & Tutorials*, vol. 9, no. 2, pp. 50–70, 2007.

[36] A. Ghaffari, "Hybrid opportunistic and position-based routing protocol in vehicular ad hoc networks," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 4, pp. 1593–1603, 2020.

[37] F. Mari, I. Melatti, E. Tronci, and A. Finzi, "A multi-hop advertising discovery and delivering protocol for multi administrative domain MANET," *Mobile Information Systems*, vol. 9, no. 3, pp. 261–280, 2013.

[38] P. Millán, C. Aliagas, C. Molina, R. Meseguer, S. F. Ochoa, and R. M. Santos, "Predicting topology propagation messages in mobile ad hoc networks: the value of history," *Sensors*, vol. 20, no. 1, p. 24, 2020.

[39] N. Bouchama, D. Aïssani, N. Djellab, and N. Nouali-Taboudjemat, "A critical review of quality of service models in mobile ad hoc networks," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 21, no. 1, pp. 49–70, 2019.

[40] D.-G. Zhang, P.-Z. Zhao, Y.-y. Cui, L. Chen, T. Zhang, and H. Wu, "A new method of mobile ad hoc network routing based on greed forwarding improvement strategy," *IEEE Access*, vol. 7, pp. 158514–158524, 2019.

[41] S. A. Sharifi and S. M. Babamir, "The clustering algorithm for efficient energy management in mobile ad-hoc networks," *Computer Networks*, vol. 166, article 106983, 2020.

[42] Z. Jipeng, L. Liangwen, and H. Tan, "Traffic-predictive QoS on-demand routing for multi-channel mobile ad hoc networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2018, no. 1, 2018.

[43] S. D. Ubarhande, D. D. Doye, and P. S. Nalwade, "A secure path selection scheme for mobile ad hoc network," *Wireless Personal Communications*, vol. 97, no. 2, pp. 2087–2096, 2017.

[44] M. R. Khosravi, H. Basri, and H. Rostami, "Efficient routing for dense UWSNs with high-speed mobile nodes using spherical divisions," *The Journal of Supercomputing*, vol. 74, no. 2, pp. 696–716, 2018.

[45] A. Pal and A. Jolfaei, "On the lifetime of asynchronous software-defined wireless sensor networks," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6069–6077, 2020.

[46] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic networking: data forwarding in disconnected mobile ad hoc networks," *IEEE Communications Magazine*, vol. 44, no. 11, pp. 134–141, 2006.

[47] M. Conti and S. Giordano, "Mobile ad hoc networking: milestones, challenges, and new research directions," *IEEE Communications Magazine*, vol. 52, no. 1, pp. 85–96, 2014.

[48] Z. Zhang, "Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges," *IEEE Communications Surveys & Tutorials*, vol. 8, no. 1, pp. 24–37, 2006.

[49] Y. Li, S. Haiying, C. Kang, and L. Guoxin, "MobileCopy: Improving data availability and file search efficiency in delay tolerant networks against correlated node failure," *IEEE*

Transactions on Mobile Computing, vol. 20, no. 1, pp. 188–203, 2021.

[50] "Disruption tolerant networking (DTN)," 2008, http://www.darpa.mil/sto/solicitations/DTN/index.html.

[51] K. Fall and S. Farrell, "DTN: an architectural retrospective," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 5, pp. 828–836, 2008.

[52] T. Spyropoulos, R. N. B. Rais, T. Turletti, K. Obraczka, and A. Vasilakos, "Routing for disruption tolerant networks: taxonomy and design," *Wireless Networks*, vol. 16, no. 8, pp. 2349–2370, 2010.

[53] S. M. Tornell, C. T. Calafate, J.-C. Cano, and P. Manzoni, "DTN protocols for vehicular networks: an application oriented overview," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 868–887, 2015.

[54] M. Quwaider and S. Biswas, "DTN routing in body sensor networks with dynamic postural partitioning," *Ad Hoc Networks*, vol. 8, no. 8, pp. 824–841, 2010.

[55] N. Banerjee, M. D. Corner, and B. N. Levine, "Design and field experimentation of an energy-efficient architecture for DTN throwboxes," *IEEE/ACM Transactions on Networking*, vol. 18, no. 2, pp. 554–567, 2010.

[56] M. Seligman, K. Fall, and P. Mundur, "Storage routing for DTN congestion control," *Wireless Communications and Mobile Computing*, vol. 7, no. 10, pp. 1183–1196, 2007.

[57] J. Whitbeck and V. Conan, "HYMAD: hybrid DTN-MANET routing for dense and highly dynamic wireless networks," *Computer Communications*, vol. 33, no. 13, pp. 1483–1492, 2010.

[58] Y. Mao, C. Zhou, Y. Ling, and J. Lloret, "An optimized probabilistic delay tolerant network (DTN) routing protocol based on scheduling mechanism for Internet of things (IoT)," *Sensors*, vol. 19, no. 2, p. 243, 2019.

[59] S. Wan, X. Xu, T. Wang, and Z. Gu, "An intelligent video analysis method for abnormal event detection in intelligent transportation systems," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–9, 2020.

[60] M. García-Valls, A. Dubey, and V. Botti, "Introducing the new paradigm of social dispersed computing: applications, technologies and challenges," *Journal of Systems Architecture*, vol. 91, pp. 83–102, 2018.

[61] D. Hu and B. Krishnamachari, "Throughput optimized scheduler for dispersed computing systems," in *2019 7th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*, pp. 76–84, Newark, CA, USA, May 2019.

[62] K. Fujikawa, H. Harai, M. Ohmori, and M. Ohta, "Quickly converging renumbering in network with hierarchical link-state routing protocol," *IEICE Transactions on Information and Systems*, vol. E99.D, no. 6, pp. 1553–1562, 2016.

[63] C.-S. Yang, R. Pedarsani, and A. S. Avestimehr, "Communication-aware scheduling of serial tasks for dispersed computing," *IEEE/ACM Transactions on Networking*, vol. 27, no. 4, pp. 1330–1343, 2019.

[64] A. Knezevic, Q. Nguyen, J. A. Tran et al., "CIRCE-A runtime scheduler for DAG-based dispersed computing," in *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*, pp. 12–14, San Jose, CA, USA, October 2017.

[65] Q. Nguyen, P. Ghosh, and B. Krishnamachari, "End-to-end network performance monitoring for dispersed computing," in *2018 International Conference on Computing, Networking and Communications (ICNC)*, pp. 707–771, Maui, HI, USA, March 2018.

[66] P. Ghosh, Q. Nguyen, and B. Krishnamachari, "Container orchestration for dispersed computing," in *Proceedings of the 5th International Workshop on Container Technologies and Container Clouds - WOC '19*, pp. 19–24, Davis, CA, USA, December 2019.

[67] J. Spillner and A. Schill, "Towards dispersed cloud computing," in *May 2014 in 2014 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, pp. 170–174, Odessa, Ukraine, May 2014.

[68] K. S. Meena and T. Vasanthi, "Reliability design for a MANET with cluster-head gateway routing protocol," *Communications in Statistics - Theory and Methods*, vol. 45, no. 13, pp. 3904–3918, 2016.

[69] Z. Lv, H. Song, P. Basanta-Val, A. Steed, and M. Jo, "Next-generation big data analytics: state of the art, challenges, and future research topics," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 1891–1899, 2017.

[70] S. K. Goudos, P. D. Diamantoulakis, and G. K. Karagiannidis, "Multi-objective optimization in 5G wireless networks with massive MIMO," *IEEE Communications Letters*, vol. 22, no. 11, pp. 2346–2349, 2018.

[71] S. Wan, R. Gu, T. Umer, K. Salah, and X. Xu, "Toward offloading internet of vehicles applications in 5G networks," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–9, 2020.

[72] D. Anveshini, "LANMAR routing protocol to support real-time communications in MANETs using soft computing technique," in *Data Engineering and Communication Technology*, vol. 1079 of Advances in Intelligent Systems and Computing, pp. 231–243, 2020.

[73] H. Mohamed, M. H. Lee, M. Sarahintu, S. Salleh, and B. Sanugi, "Application of Taguchi's design of experiment in performance analysis of destination sequence distance vector (DSDV) routing protocol in mobile ad hoc networks," *Sains Malaysiana*, vol. 38, no. 3, pp. 423–428, 2009.

[74] D. Z. Rodríguez, R. L. Rosa, and P. H. M. de Lima, "New cache system-based power-aware algorithm in MANET," in *2010 Fifth International Conference on Digital Telecommunications*, pp. 86–91, Athens, Greece, June 2010.

[75] J. Toutouh, J. Garcia-Nieto, and E. Alba, "Intelligent OLSR routing protocol optimization for VANETs," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 4, pp. 1884–1894, 2012.

[76] R. Bai and M. Singhal, "DOA: DSR over AODV routing for mobile ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 10, pp. 1403–1416, 2006.

[77] R. B. Anand and P. Kumar, "Process automation of simulation using Toolkit/Tool Command Language (TK/TCL) scripting," *IOP Conference Series: Materials Science and Engineering*, vol. 42, 2018.

[78] L. Li, T.-T. Goh, and D. Jin, "How textual quality of online reviews affect classification performance: a case of deep learning sentiment analysis," *Neural Computing and Applications*, vol. 32, no. 9, pp. 4387–4415, 2020.

[79] C. Chen, Y. Zhang, M. R. Khosravi, Q. Pei, and S. Wan, "An intelligent platooning algorithm for sustainable transportation systems in smart cities," *IEEE Sensors Journal*, 2020.

[80] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking:

past, present, and future of programmable networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014.

[81] C. Jayapal, S. Jayavel, and V. P. Sumathi, "Enhanced service discovery protocol for MANET by effective cache management," *Wireless Personal Communications*, vol. 103, no. 2, pp. 1517–1533, 2018.

[82] T. T. Nguyen, V. N. Ha, L. B. Le, and R. Schober, "Joint data compression and computation offloading in hierarchical fog-cloud systems," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 293–309, 2020.

[83] T. Q. Thinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in mobile edge computing: task allocation and computational frequency scaling," *IEEE Transactions on Communications*, vol. 65, no. 8, pp. 3571–3584, 2017.