Journal of Southeast University (English Edition) Vol. 34, No. 2, pp. 156 - 165 June 2018 ISSN 1003—7985

A movement-assisted software-defined sensor network with NFV support

Yin Haohao Ding Cui Yan Feng Xia Weiwei Shen Lianfeng

(National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China)

Abstract: A flexible and controllable movement-assisted software-defined sensor network (MA-SDSN) based on the software-defined network (SDN) and network function virtualization (NFV) is proposed. First, a three-layer fundamental architecture is proposed to overcome the inherent distributed management and rigidity of the conventional wireless sensor networks. Furthermore, the platform for research and development of MA-SDSN is established, and the dumb node (DN), the software-defined node (SN) and the movement-assisted node (MN) are designed and implemented. Then, the southbound application programming interface (API) is designed to provide a series of frames for communication between controllers and sensor nodes. The northbound API is developed and demonstrated overall and in detail. The functions of the controller are presented including topology discovery, dynamic networking, packet processing, mobility management and virtualization. Followed by the MA-SDSN network model, a Markov chain-based movement-assisted weighted relocation (MMWR) topology control algorithm is proposed to redeploy the MNs based on the node status and weight. Simulation results and analysis indicate that the proposed algorithm based on the MA-SDSN extends network lifetime with a lower average power consumption.

Key words: software-defined sensor network; network function virtualization; movement-assisted; topology control

DOI: 10.3969/j.issn.1003-7985.2018.02.003

The wireless sensor network (WSN) is deemed to be ubiquitous and has been widely applied in the infrastructure of Internet of things (IoT). However, there are some inherent limitations deep-rooted in the architecture of the WSN. First of all, the conventional WSN is a distributed architecture, which needs a distributed management system. The decentralized management system is difficult to implement[1]. Secondly, the purpose-built WSN is too stubborn to adapt to protocol changes. Developers and innovators are unavailable to deploy their novel networks, and the ways of innovations in protocols or sensor network frameworks are blocked[2]. Thirdly, the rigidity problem of conventional networks is conspicuous[3] since the conventional WSNs can do little to repair the network or control the topology when changes of the nodal status occur.

Inspired by software-defined networks (SDNs) which uncouple conventional IP networks into a control plane and data plane, a new type of sensor network which merges the SDN into WSN has been built. The novel sensor network architecture called the software-defined sensor network (SDSN)[4] can be applied to topology control and coverage optimization of WSNs. Along with SDN, the network function virtualization (NFV) decouples the network functions from the dedicated hardware[5]. With the development of virtualization technologies, the network applying NFV can address the network ossification problem since the network functions can be implemented in software rather than in purpose-built hardware by the developer and providers.

As a matter of fact, topology control in conventional WSNs is always a challenging problem. How to balance the power and connectivity among sensor nodes is a critical task in topology control[6]. The introduction of the movement-assisted node can facilitate the topology control and hole healing of the WSN.

In this paper, we propose a flexible and controllable movement-assisted software-defined sensor network (MA-SDSN) with NFV support. In addition, we define the formats of flow packet and flow entry. The details such as topology discovery, packet processing, mobility management and virtualization are further discussed in the subsequent sections. After that, the network model of MA-SDSN is established. Under the network model, a Markov chain based movement-assisted weighted relocation (MMWR) topology control algorithm is proposed based on the mobility of the movement-assisted node and the centralized configuration of the controller. The simulation results indicate that the proposed algorithm extends network lifetime with a lower average power consumption.

1 Related Work

Over the years, there have been plenty of studies on improving existing sensor network frameworks or proposing novel-innovative network paradigms by academics

Received 2017-11-26, **Revised** 2018-02-23.

Biographies: Yin Haohao (1991—), male, graduate; Shen Lianfeng (corresponding author), male, professor, lfshen@seu.edu.cn.

Foundation item: The National Natural Science Foundations of China (No. 61471164, 61601122).

Citation: Yin Haohao, Ding Cui, Yan Feng, et al. A movement-assisted software-defined sensor network with NFV support[J]. Journal of Southeast University (English Edition), 2018, 34(2): 156 - 165. DOI: 10.3969/j.issn.1003-7985.2018.02.003.

and industry researchers. Luo et al. [1] analyzed the issues deep-rooted in the conventional WSN and proposed the southbound interface called sensor OpenFlow (SOF). Galluccio et al. [7] proposed the SDN-WISE by designing a stateful SDN solution for WSNs. The SDN-WISE makes programmable sensor nodes into finite state machines, thus enabling them to run operations that cannot be supported by stateless solutions. The architecture for SDN-based sensor networks[8] follows the idea that all the logic is still decided in the control plane while the data plane still handles the flow data without general knowledge of the network. Kobo et al. [9] showed us a survey on the challenge and design requirements of the software-defined WSN.

In Ref. [5], software-defined network function virtualization technologies are summarized, which provide programmable network connectivity between network functions to achieve optimized traffic engineering and steering. Coincidentally, Bizanis et al. [10] indicated that the network virtualization for WSN can serve multiple sensing applications concurrently and the network contains multiple-purpose sensor that are application-agnostic. Khan et al. [11] divided the virtualization of WSNs into node-level virtualization and network-level virtualization so that multiple applications can share the same WSN infrastructure.

There are also certain efficient movement-assisted sensor deployment solutions proposed to relocate the sensor nodes[12]. These sensor nodes were deployed in the region of interest (RoI) randomly or autonomously for coverage hold detection, healing and power control[13-14]. The research on topology control of the WSN has been proceeding rapidly since it commenced years ago. A taxonomy of topology control and a suite of open issues in WSNs are presented in Refs. [6, 15]. Using the graph theory, Ramanathan et al. [16] proposed CONNECT and BICONNECT algorithms which are the optimum centralized algorithms of static networks.

Nevertheless, topology control has never made the breakthrough in actual deployments. The conventional WSN has the limitations of upgrading, supervising, self-organizing and limited computing power. Due to these limitations, existing topology control algorithms based on decentralized WSN are limited to being distributed algorithms and are impractical to consider transmission power or residual energy of the sensor nodes without global network information.

2 Overview of MA-SDSN with NFV Support

2.1 Foundations

As mentioned before, the SDN technology decouples the network control plane from the data plane. By using the software-defined architecture, topology control algorithms can be deployed and updated by the centralized controller in real time. The sensor OpenFlow is the southbound interface protocol. Since the centralized controller has global nodes information, it can provide optimal performance, deployment and management especially when the scale of WSN expands significantly.

As we have mentioned, NFV can decouple the network functions from the dedicated hardware. The commonly used network functions considered for NFV include network switching elements, mobile network devices and virtualized home devices. Due to diverse service demands, the infrastructure network is divided into different virtualized network functions (VNFs)[5]. Different network functions implemented in dedicated infrastructure networks such as computing/switch/storage resources migrate to the virtualization layer which runs functional software on commercial off-the-shelf (COTS) equipment.

The advent of SDN and NFV technologies facilitate the network architecture designs and network optimization, and the development of the unmanned aerial vehicle (UAV) facilitates the deployment of the movement-assisted node. The NFV is in charge of the virtualization of infrastructure, and the SDN is in charge of the virtualization of the network itself.

2.2 MA-SDSN with NFV support

In this section, we propose the movement-assisted software-defined sensor network (MA-SDSN) with NFV support. The layered architecture is shown in Fig. 1.

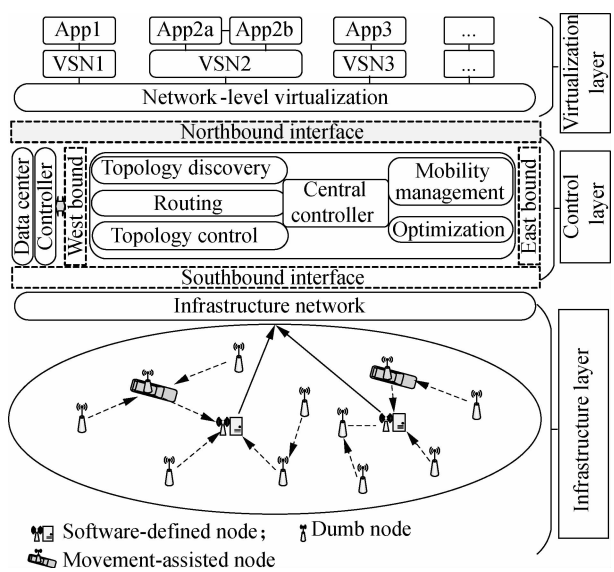

Fig. 1 The architecture of MA-SDSN with NFV support

The three-layer framework includes the infrastructure layer, control layer and virtualization layer. The infrastructure layer which is located in the underlying network consists of dumb nodes, movement-assisted nodes and software-defined nodes. The software-defined node is designed to play the roles of the local controller that makes local decisions such as topology discovery and in-network processing. Without loss of generality, the software-de-

fined node is also set as the sink node. All collected data will be converged to sink nodes. The movement-assisted node controlled by the central controller can be used for coverage hole detection, healing and topology optimization. The dumb node in the MA-SDSN is the ordinary sensor node which can only collect data or forward data according to the instructions of the controller.

Above the infrastructure layer is the control layer that connects to the infrastructure layer through the southbound interface. The central controller is responsible for some global and long-term decisions such as routing protocol and global configuration. A central controller may also have good knowledge about application requirements from the virtualization layer such as the discovery of the actual topology and the quality of the links.

The virtualization layer is an abstraction of the underlying wireless sensor network. The virtualization layer implements network-level virtualization into multi-applications. How to share multiple applications on the same WSN infrastructure is becoming a matter of general concern. The primary way that network-level virtualization can be implemented is to build a virtual sensor network (VSN)[11]. The VSNs divide the WSN into multiple virtual networks for different applications. Dynamic VSN assignments promote the resource efficiency and satisfy various requirements.

The southbound interface connecting to the infrastructure layer and the northbound interface connecting to the virtualization layer are designed for the architecture. Besides, the westbound interface and eastbound interface are presented for multi-controllers and large-scale data centers.

2.3 Underlying wireless sensor network

As illustrated in Fig. 1, the underlying infrastructure of the WSN is composed of the dumb node (DN), the movement-assisted node (MN) and the software-defined node (SN). These customized nodes are redefined for MA-SDSN and are deployed to constitute infrastructure networks collaboratively for various applications and optimization targets.

2.3.1 Dumb nodes

In the designed network, there are some sensor nodes that are considered to collect or forward data according to the flow entries in flow tables configured by the central controller. These kind of nodes are called dumb nodes (DN).

The structure of the DN is shown in Fig. 2. The functional role of the DN includes topology discovery, packet forwarding, data collection and configuration. It is worth mentioning that only when data is needed by an MA-SDSN application, the collection of data can be started, since extra data collection will reduce the network lifetime.

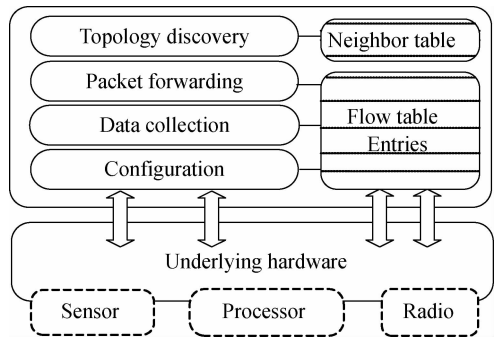

Fig. 2 The structure of the DN

2.3.2 Movement-assisted nodes

The design of the MN considers a scenario in a dangerous or hostile task environment. With the advantage of the mobile device, sensor nodes are able to move on their own way so that we can solve the WSN deployment problems. We are looking for an architecture that can improve overall network performance. Unlike motionless nodes or random mobile nodes which may cause unforeseeable network topological variation, the movement-assisted sensor node can be migrated to specified location by the instructions from the controller.

As shown in Fig. 3, the MN executes the missions such as topology discovery or relocation depending on the flow entries in flow tables configured by the controller. In order to migrate the MN to the designated location, the inertial navigation system (INS) is the optimal technology. Without loss of generality, the movement-assisted nodes can be created by embedding sensors on mobile platforms, such as robots and UAV.

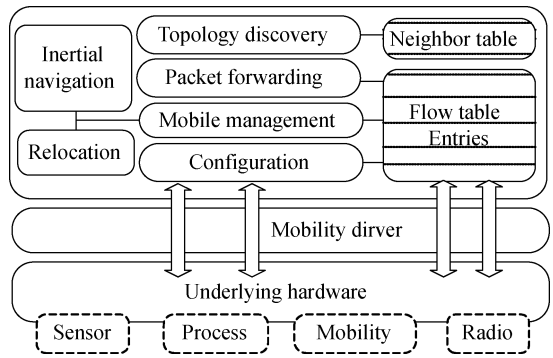

Fig. 3 The structure of the MN

2.3.3 Software-defined nodes

We put forward a flexible programming model for the SN with NFV support which is illustrated in Fig. 4. The designed model is layered in three layers: the physical layer, the control layer and the virtualization layer. Node-level virtualization executed in the virtualization layer is implemented into an embedded virtual machine (VM) so that a wireless sensor node can be applied in a multi-purpose scene. Different functions are implemented upon sensor hardware. The WSN node-level virtualization

supports over-the-air (OTA) programming and allows multiple applications to run multi-tasks concurrently in a single sensor node. The OTA programming is enabled to update tasks executions according to decisions from the central controller.

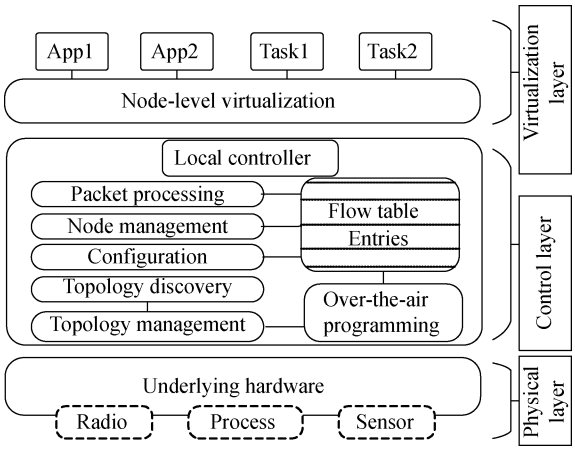

Fig. 4 The structure of the SN

The most important idea in the proposed underlying WSN is that the local controller is introduced into the SN. More autonomy is given to the SN which plays the role of the local controller as well as the sink node. The local controller can launch the topology discovery processing at the beginning and converge topology information to the central controller. Thanks to the local controller, communication traffic between the control plane and the data plane in the WSN is decreased, as well as the energy consumption.

3 Design and Implementation of the MA-SDSN with NFV Support

Both centralized and decentralized control can be deployed in the underlying WSNs based on the MA-SDSN architecture with NFV support. The combination of the central controller and local controller decreases communication traffic between the control plane and the data plane. Furthermore, it can improve the efficiency of the network operation and quality of service (QoS) by increasing the response rate of the whole network. In this architecture, network application and optimization algorithms (such as the topology control algorithm) can be deployed by the controller which can provide global network information and centralized management in real time. In this section, the details of the framework will be described by defining packet formats and illustrating the protocol process, and a platform for research and development is implemented based on the designed MA-SDSN. The detailed designs of topology discovery, packet processing, mobility management and virtualization make the architecture feasible and legible.

3.1 Sensor nodes

The structures of the three kinds of nodes in the underlying WSN are introduced in Section 2.3. In this section, the movement-assisted node controlled by the controller is designed and realized as well as the dumb node, and the software-defined node. These kinds of functional sensor nodes are designed to adapt to different needs and application scenarios.

In the MA-SDSN, routing and topology control algorithms are no longer executed in the dumb node which merely collects or forwards packets according to flow entries in the flow table. The implementation of the dumb node is based on the zigbee CC2530 module. The movement-assisted node is made up of the wireless sensor module, the power module and the mobility driver module. The designed movement-assisted node can be relocated by the central controller for specific tasks, including data acquisition and topology control.

As mentioned before, the programmable software-defined node is introduced into the MA-SDSN. The software-defined node is made up of the wireless sensor module, the power module and the embedded processor module which is based on the ARM nano Pi-M3 platform. As the local controller, the software-defined node will cover part of control functions of the central controller including topology discovery, packet processing and data collection.

3.2 MA-SDSN controller

The central controller is the core of the whole MA-SDSN. It runs the main part of network management, control or optimization. The connection between the central controller and the software-defined node can be established by the Ethernet port, COM port or TCP/IP socket. The controller is responsible for the whole network operations and management. In this paper, we realize the graphical user interface (GUI) of the MA-SDSN controller based on the Linux operating system. The network features include topology discovery, dynamic networking, packet processing and mobility management and so on.

3.2.1 Topology discovery

As illustrated in Fig. 4, the process of topology discovery (TD) is launched by the local controller which is a software-defined node as well as a sink node. Then, local neighbor information is delivered to the central controller by the sink node. We take into account the neighbor information and next hop information towards the sink node of each sensor node[7]. The neighbor information includes addresses, type and the residual energy of neighbor nodes, and the value of the received signal strength indicator (RSSI) of the received message. First, all the sink nodes will send a topology discovery packet using the maximum transmitting power when broadcasting simultaneously. The frame format of TD is defined, as shown in

Fig. 5, and the explanation of the fields is shown in Tab. 1. The frame format of the neighbor information is defined, as shown in Fig. 6, and the explanation of the fields is shown in Tab. 2. Then the operations which handle the TD packet by the sensor node are expounded as follows.

Fig. 5 Frame of topology discovery

Tab. 1 Explanation of topology discovery frame

Field	Width/bit	Description
HE	16	Header of frame
FT	8	Frame type
SA	32	Source address
DA	32	Destination address
NT	2	Node type(00: SN, 01: MN; 10: DN; 11: Unknow)
RE	8	Residual energy
LOC	32	Location
NHTS	16	Next hop to sink
HO	8	Hops
TA	16	Tail of frame

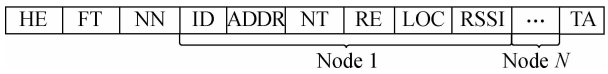

Fig. 6 Frame of neighbor information

1) For every sensor node that has received a TD packet, it extracts its neighbor address, neighbor type and nodal residual energy as well as the value of RSSI, and updates them as a piece of new neighbor information. If the neighbor address has existed in current neighbor information, just update the value of the RSSI and residual energy.

2) Extract the next hop towards the sink node and hops from the sink node, if the current packet has a lower value of hops, then update the next hop towards the sink node and hops with the value of the packet plus one.

3) For any sensor node which has received a TD packet but has not broadcasted yet, a random waiting time is to avoid conflicts with other broadcasting packets.

4) Encapsulate the sensor node address and other fields into a new TD packet and send it by broadcasting.

Tab. 2 Explanation of neighbor information frame

Field	Width/bit	Description
NN	8	Number of neighbors
ID	16	Node ID
ADDR	16	Short address
RSSI	16	Received signal strength indicator

3.2.2 Dynamic networking

Once a new node is added into the network, the controller will take into account the new node when planning the topology. The procedure of real-time dynamic networking is implemented based on the MST algorithm.

Dynamic networking of the MA-SDSN controller achieves the goal of software definition. First, the process of networking is finished in the controller but not in the underlying network. Secondly, after completing the topology planning, the controller will generate flow tables for data forwarding. The control functions of the controller are precisely realized through the flow table.

3.2.3 Packet processing

The flow packets transmitted in the network is designed to be processed according to flow entries in the flow table. If the node cannot find corresponding entries, the request frame of the flow table will be sent. The format of the request frame of the flow table is illustrated in Fig. 7, and the format of the response frame of the flow table is defined in Fig. 8. The explanation of the fields is shown in Tab. 3. The flow packets share the same fields with match field in flow entry for matching items. The action field in the proposed network supports two kinds of actions: forward and drop. The packet processing describes the procedure of how a flow packet generated by a sensor node is processed. The flow diagram of packet processing is shown in Fig. 9. The data request frames and data response frames between sensor nodes and controllers can be used to collect topology information and sensor data. The frames are shown in Fig. 10 and Fig. 11. The explanation of the fields is shown in Tab. 4 and Tab. 5.

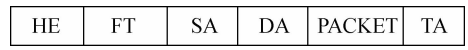

Fig. 7 Request frame of flow table

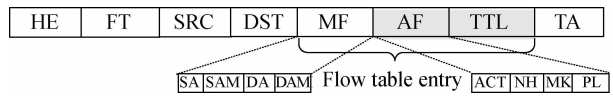

Fig. 8 Response frame of flow table

Tab. 3 Explanation of flow table response frame

Field		Width/bit	Description
SRC		32	Header of packet
DST		32	Frame type
MF	SA	32	Source address
	SAM	32	Source address mask
	DA	32	Destination address
	DAM	32	Destination address mask
AF	ACT	1	Action(1: forward, 0: drop)
	NH	32	Next hop address
	MK	32	Address mask
	PL	16	Transmitting power
TTL		16	Time to live

Considering the packet processing of any sensor node, the destination address is extracted when a flow packet is received by the radio to judge if the node is at the corresponding destination. If it is, certain operations will be handled according to the frame type. Otherwise, the operation of the lookup table is executed to decide how the corresponding packet can be dealt with when a flow entry in the flow table is matched. If there is no matched flow entry, a sub-process is launched to request a new flow entry to the controller. In addition to this, each flow entry

in the flow table has a TTL field that specifies the lifetime of the entry. After a lifetime, these entries will be deleted to improve the efficiency of matching.

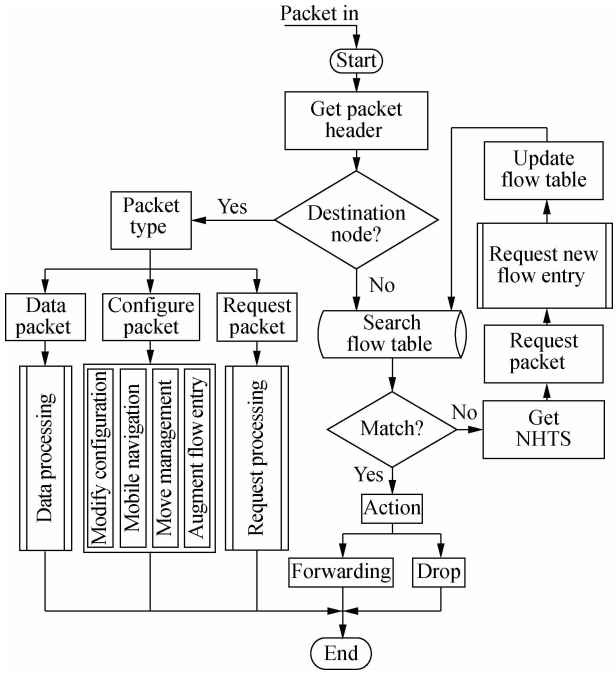

Fig. 9 Packet processing

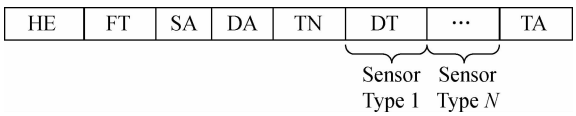

Fig. 10 Sensor data request frame

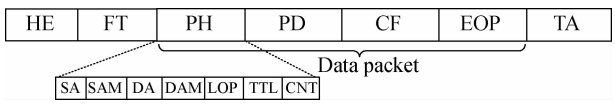

Fig. 11 Data response frame

Tab. 4 Explanation of sensor data request frame

Field	Width/bit	Description
TN	8	Sensor type number
DT	8	Sensor data type

Tab. 5 Explanation of data response frame

Field		Width/bit	Description
PH	SA	32	Source address
	SAM	32	Source address mask
	DA	32	Destination address
	DAM	32	Destination address mask
	LOP	32	Length of packet
	TTL	16	Time to live
	CNT	8	Number of hops
PD		N	Data body
CF		32	Check field
EOP		8	End of packet

3.2.4 Mobility management

The movement-assisted node which can be migrated to a specified location is dissimilar to motionless sink nodes or random mobile nodes which may bring about an unforeseeable change of the network topology. Most movement-assisted nodes are deployed in the target area randomly or autonomously only for coverage hole detection and healing. The central controller has global topology information including nodal state, link state and residual energy.

The mobility management and deployment of the movement-assisted nodes are implemented by local controllers and the central controller through flow packets. The relocation frame of the movement-assisted node is shown in Fig. 12. Accurate inertial navigation systems (but not implemented) in the movement-assisted node can provide digital course information to controllers by transmitting flow packets. The movement-assisted node can also be used to optimize the global topology. Under the new architecture, practical algorithms can be proposed to relocate these nodes for bringing down power consumption and prolonging the lifetime of the underlying network.

Fig. 12 Relocation frame of the movement-assisted node

3.3 Virtualization

In the proposed MA-SDSN architecture, both node-level virtualization and network-level virtualization are implemented to meet a variety of needs and to increase the utilization efficiency of hardware resources. Node-level virtualization can be realized in SNs based on VMs or operating systems running on them. On the one hand, one software-defined node can handle multiple tasks in real time simultaneously. On the other hand, the central controller which virtualizes the infrastructure on a network-level allocates various tasks to multiple software-defined nodes collaboratively. The network-level virtualization is implemented by abstracting multi-virtual sensor networks (VSNs) which serve specified applications from the underlying network infrastructure.

We design the frames illustrated in Figs. 5 to 8 and Figs. 10 to 11 in the implementation of MA-SDSN southbound API. As for the northbound APIs, in the implementation of the controller, a series of interface functions are defined and divided into four categories. The northbound APIs consist of network initialization, network control, data processing and mobility management APIs.

4 Movement-Assisted Topology Control Algorithm in MA-SDSN

4.1 Network model

Without loss of generality, we denote the MA-SDSN as a weighted connected undirected graph $G = (V_{dn} \cup V_{mn} \cup V_{sn}, E)$, in which V_{dn} is the set of DNs, V_{mn} is the set of MNs, and V_{sn} is the set of SNs. If d_{uv} denotes the dis-

tance between node u and v, then $E\{(u,v): d_{uv} \leqslant r_{\max}, u, v \in V_{dn} \cup V_{mn} \cup V_{sn}\}$ denotes the original edges connecting all nodes within the maximum transmission range $r_{\max}$. If the MNs can communicate directly with any node within range $r_{\max}$, then we can define a MN's reachable neighborhood node set.

Definition 1 For a certain MN k, the reachable neighborhood node set is defined as $V_N^*(k) = (N^k, L^k)$, in which $N^k = \{v \mid d_{kv} \leqslant r_{\max}, v \in V_{dn} \cup V_{sn}, k \in V_{mn}\}$ is the set of node identifies, and $L^k: N^k \rightarrow (\mathbf{R}^+, \mathbf{R}^+)$ is the corresponding coordinates of reachable neighborhood nodes in the RoI area. For a circular area with the MN k as the center, the maximum transmission range $r_{\max}$ as radius, we define the vector form of the reachable neighborhood node set.

Definition 2 The vector form of the reachable neighborhood node set is defined as $V_N^*(k) = \{r_{kv} \mid |r_{kv}| < r_{\max}, v \in V_N^*(k)\}$, r_{kv} represents the vector from center k to node v.

Definition 3 The scheduling parameters (SP) matrices of sensor nodes in the n-th state are defined as

$$S_v^n = (E_v^n, p_v^n, g_v^n, E_0, P_{\max}, g_{\max}, \alpha, \beta, \gamma, \eta), v \in V_{dn}) \quad (1)$$

where E_v^n, p_v^n, g_v^n represent, respectively, the residual energy, transmission power and node degree of sensor node v in the n-th state. α, β, γ and η are the non-negative weight factors of scheduling parameters. E_0 is the original energy of the sensor node, and $g_{\max}$ is the maximum node degree among all nodes in the established topology.

4.2 Movement-assisted node relocation algorithm

First of all, we propose a nodal weight function to evaluate the status of a set of nodes. Then, the relocation matrices are constructed to direct the relocation of MNs.

4.2.1 Node weight function

Nodal weight reflects the node status at present such as transmission power, residual energy and node degree depicted in Definition 3. In all of these varibles, node power p_v is the first considered factor since the minimal possible power is the primary purpose of topology control. The residual energy $E_{r(v)}$ of the sensor node influences the lifetime of the network, and the shorter residual node energy implicitly indicates the less survival time of the whole network. For a similar reason, larger node degree g_v implies more redundancy and throughput. From the above, we define a node weight function based on the S defined in Definition 3.

$$\omega_v^n = \eta\left(\frac{p_v^n}{P_{\max}}\right)^{\alpha}\left(1 - \frac{E_{r(v)}^n}{E_0}\right)^{\beta}\left(\frac{g_v^n}{g_{\max}}\right)^{\gamma} \qquad v \in V_N^*(k) \quad (2)$$

Considering the circular area with the MN k as the center and the maximum transmission range $r_{\max}$ as the radius, all the reachable neighborhood nodes in $V_N^*(k)$ which have the node weights determined by Eq. (2) will be taken into account when relocating the MNs.

Definition 4 The circular area is divided into four quadrants, $R_{\mathrm{I}}, R_{\mathrm{II}}, R_{\mathrm{III}}, R_{\mathrm{IV}}$. Then, when v is located in R_i, we have

$$W(R_i) = \sum_v \omega_v^n \qquad v \in V_N^*(k), i \in \{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\} \quad (3)$$

$$W(V_N^*(k)) = \sum_i W(R_i) = \sum_v \omega_v^n, v \in V_N^*(k) \qquad i \in \{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\} \quad (4)$$

Eq. (3) formulate the sum of node weight in $R_{\mathrm{I}}, R_{\mathrm{II}}, R_{\mathrm{III}}, R_{\mathrm{IV}}$, respectively, and Eq. (4) represents the weight sum of all the nodes in $V_N^*(k)$.

4.2.2 Relocation vectors

Definition 5 Assume that $L_{k,o}^{(n)}$ is the location of the MN k at the n-th state. Then, the relocation space within the four quadrants $R_{\mathrm{I}}, R_{\mathrm{II}}, R_{\mathrm{III}}, R_{\mathrm{IV}}$ is defined as

$$\mathrm{LS}_k^{(n)} = [\boldsymbol{L}_{k,\mathrm{I}}^{(n)} \quad \boldsymbol{L}_{k,\mathrm{II}}^{(n)} \quad \boldsymbol{L}_{k,\mathrm{III}}^{(n)} \quad \boldsymbol{L}_{k,\mathrm{IV}}^{(n)}] \quad (5)$$

where $\mathrm{LS}_k^{(n)}$ is the relocation space of MNs to accommodate relocation vectors which utilize the vector forms of reachable neighborhood node set defined in Definition 2, and they are written as

$$\boldsymbol{L}_{k,\mathrm{I}}^{(n)} = \frac{\sum \omega_v^n |r_{kv}|}{N_{\mathrm{I}}} \frac{\sum \omega_v^n r_{kv}}{|\sum \omega_v^n r_{kv}|} \qquad r_{kv} \in V_N^*(k) \cap R_{\mathrm{I}} \quad (6)$$

$$\boldsymbol{L}_{k,\mathrm{II}}^{(n)} = \frac{\sum \omega_v^n |r_{kv}|}{N_{\mathrm{II}}} \frac{\sum \omega_v^n r_{kv}}{|\sum \omega_v^n r_{kv}|} \qquad r_{kv} \in V_N^*(k) \cap R_{\mathrm{II}} \quad (7)$$

$$\boldsymbol{L}_{k,\mathrm{III}}^{(n)} = \frac{\sum \omega_v^n |r_{kv}|}{N_{\mathrm{III}}} \frac{\sum \omega_v^n r_{kv}}{|\sum \omega_v^n r_{kv}|} \qquad r_{kv} \in V_N^*(k) \cap R_{\mathrm{III}} \quad (8)$$

$$\boldsymbol{L}_{k,\mathrm{IV}}^{(n)} = \frac{\sum \omega_v^n |r_{kv}|}{N_{\mathrm{IV}}} \frac{\sum \omega_v^n r_{kv}}{|\sum \omega_v^n r_{kv}|} \qquad r_{kv} \in V_N^*(k) \cap R_{\mathrm{IV}} \quad (9)$$

where $N_{\mathrm{I}}, N_{\mathrm{II}}, N_{\mathrm{III}}, N_{\mathrm{IV}}$ are the number of static nodes within the four quadrants $R_{\mathrm{I}}, R_{\mathrm{II}}, R_{\mathrm{III}}, R_{\mathrm{IV}}$. The scalar forms $\boldsymbol{L}_{k,i}^{(n)}, i \in \{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}$ is the location where the vector forms $\boldsymbol{L}_{k,i}^{(n)}, i \in \{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}$ are pointing to.

4.2.3 Relocation probability matrices

Definition 6 We define the relocation probability matrices (RPM) of the MN k in the n-th state as

$$\mathrm{RPM}_k^{(n)} = (p(\mathrm{I}) \quad p(\mathrm{II}) \quad p(\mathrm{III}) \quad p(\mathrm{IV}))_k^{(n)} \quad (10)$$

Furthermore, the relocation probability matrices $\mathrm{RPM}_k^{(n)}$ can be expanded into the migration probability be-

tween the current location $\boldsymbol{L}_k^{(n)}$ and relocation $\boldsymbol{L}_k^{(n+1)}$, formulated as

$$p_k(i) = P_k(\boldsymbol{L}_k^{(n+1)} = \boldsymbol{L}_{k,i}^{(n)} \mid \boldsymbol{L}_k^{(n)} = \boldsymbol{L}_{k,O}^{(n)}) = \frac{W(R_i)}{W(V_N^*(k))}$$
$$i \in \{\text{Ⅰ}, \text{Ⅱ}, \text{Ⅲ}, \text{Ⅳ}\} \quad (11)$$

Owing to that relocation $\boldsymbol{L}_k^{(n+1)}$ only depends on current location $\boldsymbol{L}_k^{(n)}$ and current node weight ω_v^n, while it is irrelevant to its previous status, the discrete location status $\boldsymbol{L}_k^{(n)}$ meet the expectation of the Markov chain.

Eq. (10) is the one-step transfer probability matrix of the Markov chain. The MMWR algorithm relies on status parameters of sensor nodes and weight factors in the weight function to construct relocation vectors, and to relocate the MNs based on the Markov chain RPM[17]. We construct the original topology using the centralized minimum spanning tree (MST) algorithm here by optimizing running time of constructing the whole network topology. The MST algorithm may achieve great performance in constructing the network topology.

4.3 Performance evaluation

In this section, the performance of the proposed algorithm will be evaluated by simulations, compared with other algorithms such as the CONNECT algorithm and the MST algorithm without MNs.

4.3.1 Simulation settings

In order to construct the original topology, some default parameters[18] are configured as shown in Tab. 6. Considering a MA-SDSN where all nodes are deployed randomly within the rectangle scope of 1 000 m × 1 000 m, g_{max} is the maximum degree that can be calculated by the generated topology.

Tab. 6 Simulation parameters

Parameters	Value	Description
N_{dn}	100	Number of DN
N_{mn}	3	Number of MN
N_{sn}	3	Number of SN
P_{max}/dBm	4.5	Maximum transmitting power
r_{max}/m	245	Maximum transmitting range
E_0/mJ	300	Initial residual energy
T/s	30	Topology update period

The metrics in this paper are emulated to evaluate the algorithms include: network topologies generated by different algorithms, average node degree, average transmission power, as well as the network lifetime.

4.3.2 Simulation results

In the first simulation, we generate optimized network topologies consisting of 100 DNs and 3 MNs from the original topology (see Fig. 13(a)) by the maximum transmission power P_{max}. We compare the CONNECT algorithm[16] with the centralized MST algorithm. The derived topologies are illustrated in Figs. 13(b) and (c). These topologies which generate in dissimilar ways look similar at first glance, while in details, the topology generated by the MST has less redundancy and a lower average node degree which can be seen in Fig. 14.

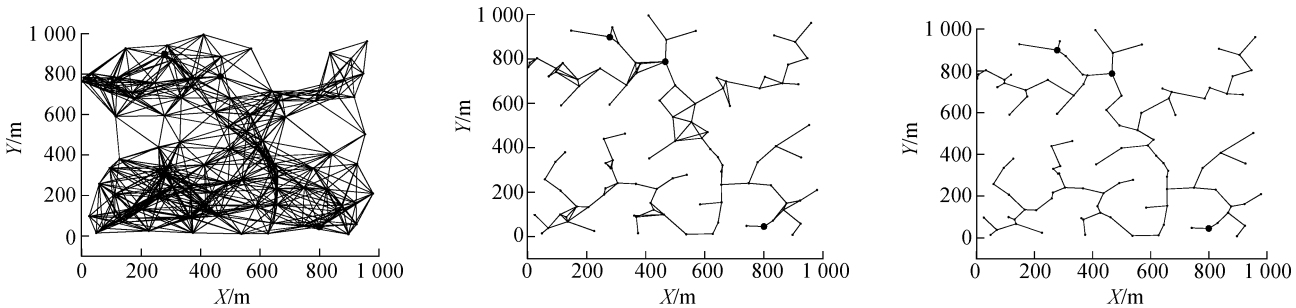

Fig. 13 Topologies derived from CONNECT and MST. (a) Original topology; (b) By CONNECT; (c) By MST

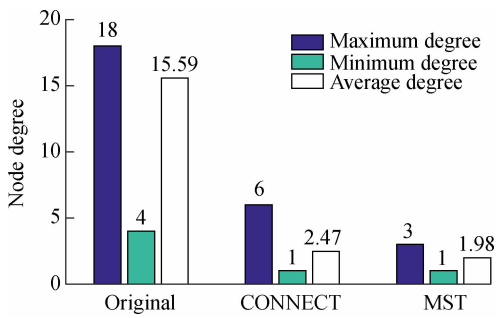

Fig. 14 Comparison of the node degree by different algorithms

It is clear that the MST algorithm has a lower average node degree compared to the CONNECT algorithm, which means less average power consumption and interference. Also, the maximum node degree in the MST algorithm is 3.

In the second simulation after establishing the optimal topology, another critical issue is to select proper values of weight factors α, β, γ and η to achieve a predicted performance. Based on the topology derived from the MST algorithm, we evaluate average power consumption p_{av} variation with α, β, γ and η using the MMWR algorithm by the control variable method. The reasonable scope of the four weight factors can be obtained as

$$0.1 \leqslant \alpha \leqslant 0.3, 0.3 \leqslant \beta \leqslant 0.7, 0.2 \leqslant \gamma \leqslant 1, \ 1 \leqslant \eta \leqslant 3.5$$

In the following simulation, we evaluate the proposed MMWR algorithm with the weight factors $\alpha = 0.2, \beta = 0.5, \gamma = 0.4, \eta = 2$, and $\alpha = 0.1, \beta = 0.5, \gamma = 0.4, \eta = 3$.

Then, we emulate the average power consumption by comparing the number of active nodes of the whole network with time elapse. The simulation results are depicted in Fig. 15. We compare the network lifetime by the MMWR with that of the conventional WSN, which has the same number of original nodes as the preceding simulation.

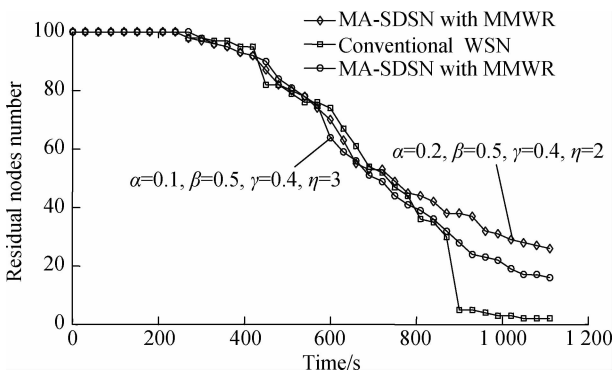

Fig. 15 The comparison of network lifetime of MA-SDSN with MMWR and conventional WSN

As can be seen from Fig. 15, the lifetime of the whole network can be prolonged and extended effectively with the collaboration of MNs. The MMWR is verified as being more robust than the conventional WSN without MNs.

5 Conclusion

We propose a network called MA-SDSN with NFV support, which can overcome the inherent limitations deep-rooted in the conventional WSNs such as distributed management, inflexibility and inefficiency. The SDN and NFV solutions have been given to realize network management and increase the utilization efficiency of hardware resources, respectively. Furthermore, a platform for research and development is built based on the designed MA-SDSN architecture. The movement-assisted node which can be controlled by the controller, dumb node, and software-defined are designed and realized. The functions of the MA-SDSN controller are designed in detail including topology discovery, dynamic networking, packet processing, mobility management and virtualization. The southbound API is designed to provide a series of frames for communication between the underlying network and centralized controller. The northbound API is developed and demonstrated overall and in detail. Finally, the network model of the MA-SDSN is established, and on that basis we propose the MMWR topology control algorithm to relocate the MNs. The simulation results indicate that the proposed algorithm extends network lifetime with a lower average power consumption. However, the algorithm has not been evaluated in the real MA-SDSN network environment since the node position and residual energy are difficult to obtain in real networks. Nevertheless, such a flexible and controllable movement-assisted software-defined sensor network will have a more promising future.

References

[1] Luo T, Tan H P, Quek T Q S. Sensor OpenFlow: Enabling software-defined wireless sensor networks[J]. *IEEE Communications Letters*, 2012, **16**(11): 1896 – 1899. DOI: 10.1109/lcomm.2012.092812.121712.

[2] Mckeown N, Anderson T, Balakrishnan H, et al. OpenFlow: Enabling innovation in campus networks[J]. *ACM Sigcomm Computer Communication Review*, 2008, **38**(2): 69 – 74.

[3] Bera S, Misra S, Roy S K, et al. Soft-WSN: Software-defined WSN management system for IoT applications[J]. *IEEE Systems Journal*, 2016: 1 – 8. DOI: 10.1109/jsyst.2016.2615761.

[4] Tang M, Yan F, Deng S, et al. Coverage optimization algorithms based on voronoi diagram in software-defined sensor networks[C]//*IEEE International Conference on Wireless Communications and Signal Processing*. Yangzhou, China, 2016. DOI: 10.1109/wcsp.2016.7752658.

[5] Blenk A, Basta A, Reisslein M, et al. Survey on network virtualization hypervisors for software defined networking [J]. *IEEE Communications Surveys and Tutorials*, 2016, **18**(1): 655 – 685. DOI: 10.1109/comst.2015.2489183.

[6] Li X, Cai J, Zhang H. Topology control for guaranteed connectivity provisioning in heterogeneous sensor networks [J]. *IEEE Sensors Journal*, 2016, **16**(12): 5060 – 5071. DOI: 10.1109/jsen.2016.2549543.

[7] Galluccio L, Milardo S, Morabito G, et al. SDN-WISE: Design, prototyping and experimentation of a stateful SDN solution for wireless sensor networks[C]//2015 *IEEE Conference on Computer Communications (INFOCOM)*. Hong Kong, China, 2015: 15385390. DOI: 10.1109/infocom.2015.7218418.

[8] Friedman R, Sainz D. An architecture for SDN based sensor networks[C]// *Proceedings of the* 18*th International Conference on Distributed Computing and Networking*. Hyderabad, India, 2017. DOI: 10.1145/3007748.3007758.

[9] Kobo H I, Mahfouz A M, Hancke G P. A survey on software-defined wireless sensor networks: challenges and design requirements[J]. *IEEE Access*, 2017, **5**: 1872 – 1899. DOI: 10.1109/access.2017.2666200.

[10] Bizanis N, Kuipers F A. SDN and virtualization solutions for the Internet of things: A survey[J]. *IEEE Access*, 2016, **4**: 5591 – 5606. DOI: 10.1109/access.2016.2607786.

[11] Khan I, Belqasmi F, Glitho R, et al. Wireless sensor network virtualization: A survey[J]. *IEEE Communications Surveys & Tutorials*, 2015, **18**(1): 553 – 576. DOI: 10.1109/comst.2015.2412971.

[12] Senouci M R, Mellouk A, Asnoune K, et al. Movement-assisted sensor deployment algorithms: A survey and taxonomy[J]. *IEEE Communications Surveys and Tutorials*,

2015, **17**(4): 2493 - 2510. DOI: 10.1109/comst.2015.2407954.

[13] Fletcher G, Li X, Nayak A, et al. Randomized robot-assisted relocation of sensors for coverage repair in wireless sensor networks[C]//2010 *IEEE* 72*nd Vehicular Technology Conference Fall*. Ottawa, ON, Canada, 2010. DOI: 10.1109/VETECF.2010.5594513.

[14] Senouci M R, Mellouk A, Assnoune K. Localized movement-assisted sensor deployment algorithm for holedetection and healing[J]. *IEEE Transactions on Parallel and Distributed Systems*, 2014, **25**(5): 1267 - 1277. DOI: 10.1109/tpds.2013.137.

[15] Li M, Li Z, Vasilakos A V. A survey on topology control in wireless sensor networks: Taxonomy, comparative study, and open issues[J]. *Proceedings of the IEEE*, 2013, **101**(12): 2538 - 2557. DOI: 10.1109/jproc.2013.2257631.

[16] Ramanathan R, Rosales-Hain R. Topology control of multihop wireless networks using transmit power adjustment[C]//*IEEE INFOCOM* 2000. Aviv, Israel, 2000: 404 - 413.

[17] Yin H H, Ding C, Yan F, et al. A robot-assisted topology control algorithm in software-defined sensor networks [C]//2017 9*th International Conference on Wireless Communications and Signal Processing (WCSP)*. Nanjing, China, 2017. DOI: 10.1109/wcsp.2017.8171086.

[18] Soleimani M, Bhuiyan M M, MacGregor M H, et al. RF channel modeling and multi-hop routing for wireless sensor networks located on oil rigs[J]. *IET Wireless Sensor Systems*, 2016, **6**(5): 173 - 179. DOI: 10.1049/iet-wss.2015.0096.

一种支持网络功能虚拟化的移动辅助软件定义传感网络

尹浩浩　丁　翠　燕　锋　夏玮玮　沈连丰

(东南大学移动通信国家重点实验室,南京 210096)

摘要:基于软件定义网络(SDN)和网络功能虚拟化(NFV)技术提出了一种灵活可控的移动节点辅助软件定义传感网络(MA-SDSN). 首先,提出了 3 层网络架构来克服传统无线传感器网络(WSN)固有的分布式管理和网络僵化问题. 然后,搭建了 MA-SDSN 网络研究开发平台,实现了哑结点、软件定义节点和移动辅助节点,设计了南向接口从而为控制器和传感器节点之间提供一系列通信帧,开发实现了北向接口,对网络控制器的具体功能(包括拓扑发现、动态组网、帧处理、移动性管理和虚拟化等)进行了详细阐述. 在提出的 MA-SDSN 网络模型的基础上,设计了基于移动节点和节点权重的马尔可夫链重定位(MMWR)拓扑控制算法. 仿真和分析表明,提出的 MMWR 算法降低了平均节点能耗从而有效扩展了网络生命期.

关键词:软件定义传感网络;网络功能虚拟化;移动辅助;拓扑控制

中图分类号:TN915