Telecommun Syst (2007) 35: 1–20
DOI 10.1007/s11235-007-9037-1

A hexagonal-tree TDMA-based QoS multicasting protocol for wireless mobile ad hoc networks

Yuh-Shyan Chen · Tsung-Hung Lin · Yun-Wei Lin

Published online: 9 September 2007

Abstract The mobile multimedia applications have recently generated much interest in wireless ad hoc networks with supporting the quality-of-service (QoS) communications. The QoS metric considered in this work is the reserved bandwidth, i.e., the time slot reservation. We approach this problem by assuming a common channel shared by all hosts under a TDMA (Time Division Multiple Access) channel model. In this paper, we propose a new TDMA-based QoS multicast routing protocol, namely hexagonal-tree QoS multicast protocol, for a wireless mobile ad hoc network. Existing QoS routing solutions have addressed this problem by assuming a stronger multi-antenna model or a less-strong CDMA-over-TDMA channel model. While more practical and less costly, using a TDMA model needs to face the challenge of radio interference problems. The simpler TDMA model offers the power-saving nature. In this paper, we propose a new multicast tree structure, namely a hexagonal-tree, to serve as the QoS multicasting tree, where the MAC sub-layer adopts the TDMA channel model. In this work, both the hidden-terminal and exposed-terminal problems are taken into consideration to possibly exploit the time-slot reuse capability. The hexagonal-based scheme offers a higher success rate for constructing the QoS multicast tree due to the use of the hexagonal-tree. A hexagonal-tree is a tree whose sub-path is a hexagonal-path. A hexagonal-path is a special two-path structure. This greatly improves the success rate by means of multi-path routing. Performance analysis results are discussed to demonstrate the achievement of efficient QoS multicasting.

Keywords QoS routing · Mobile ad hoc network (MANET) · Multi-path · Multicast · TDMA · Wireless communication

1 Introduction

A mobile ad hoc network (MANET) [4, 22] consists of wireless hosts that communicate with each other in the absence of a fixed network infrastructure. Due to factors such as radio power limitations, power consumption, and channel utilization, a mobile host might not be able to communicate directly with other hosts via a single-hop communication. In a MANET, host mobility can cause unpredictable topological variability; therefore the design of a MANET QoS routing protocol is more complicated because it requires strong fault-tolerant capability. Extensive research efforts have been devoted to the design of MANET routing protocols [8–10, 22], and all address the following issues: discovering a route from a source node to a destination, maintaining a route while it is being used, and delivering data packets to the destination host. However, these protocols, when searching for a route to the destination, are always

This work was supported by grants NSC-92-2213-E-194-022 and NSC-94-2213-E-194-030, from the National Science Council of the ROC.

Y.-S. Chen (✉)
Department of Computer Science and Information Engineering, National Taipei University, Taipei County, Taiwan, ROC
e-mail: yschen@csie.ntpu.edu.tw

T.-H. Lin
Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung County, Taiwan, ROC
e-mail: duke@ncut.edu.tw

Y.-W. Lin
Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan, ROC
e-mail: jyneda@giam.dynu.com

concerned with shortest-path routing and the availability of multiple routes in the MANET's dynamically changing environment. Connections with quality-of-service (QoS) requirements, such as multimedia with delay and bandwidth constraints, are less frequently addressed, especially when designing QoS multicast protocols for MANETs.

Recently, some researchers intensively studied the QoS issue in MANETs [5–7, 11–13, 15–19]. Initially, a quite ideal model assumes that the bandwidth of a link can be determined independently of its neighboring links [7]. This strong assumption may be realized by a costly multi-antenna model such that a host can send/receive using different antennas independently and simultaneously. Under such a model, a ticket-based QoS routing protocol was proposed in [7]. Recently, a less-strong CDMA-over-TDMA channel model was assumed in [15, 16] for developing QoS routing protocols in a MANET, where the use of a time slot on a link is only dependent on the status of its one-hop neighboring links. Observe that a code assignment protocol should be supported (this can be regarded as an independent problem, which can be found in [15, 16]). Based on such a model, Lin calculated the end-to-end path bandwidth to develop a DSDV-based QoS routing protocol [16] and an on-demand QoS routing protocol [15] for MANETs. More recently, one simpler TDMA model was further considered in [14] for developing a uni-path QoS routing protocol in a MANET. Their approaches show how to allocate time slots on each link of a path such that no two adjacent links share a common time slot.

This paper mainly presents a new TDMA-based QoS multicast routing protocol, namely the hexagonal-tree QoS multicast protocol, for a wireless mobile ad hoc network. Existing QoS routing solutions have addressed this problem by assuming a stronger multi-antenna model or a less-strong CDMA-over-TDMA channel model. This study attempts to build a new multicast tree structure, namely a hexagonal-tree, to serve as the QoS multicast tree, where the MAC sub-layer adopts the TDMA channel model [1]. In this work, both the hidden-terminal and exposed-terminal problems are taken into consideration in order to possibly exploit the time-slot reuse capability. This hexagonal-based scheme offers a higher success rate for constructing a QoS multicast tree due to use of the hexagonal-tree. A hexagonal-tree is a tree whose sub-path is a hexagonal-path. A hexagonal-path is a special two-path structure. This greatly improves the success rate by means of multi-path routing. Performance analysis results are discussed to demonstrate achievement of efficient QoS multicasting.

The rest of the paper is organized as follows. Section 2 presents basic ideas and challenges. Our protocol is developed in Sect. 3, and experimental results are discussed in Sect. 4. Section 5 concludes this paper.

2 Basic ideas and challenges

Existing TDMA-based routing protocols ignore the transmission activities of individual mobile hosts. Recall the hidden-terminal and exposed-terminal problems, which are well-known problems in the literature of radio-based communication. Consider the scenario in Fig. 1, where the bandwidth requirement is two time slots. Fig. 1(a) shows the hidden-terminal problem in which if A sends data to B on slots 1 and 2, then D is not allowed to send data to F on slots 1 and 2. On the contrary, Fig. 1(b) illustrates the exposed-terminal problem in which if A sends data to B on slots 1 and 2, then C is allowed to send data to E on slots 1 and 2. Observe that if the QoS routing protocol only considers the hidden-terminal problem, then no three adjacent links are allowed to share the same free time-slots. This limitation can be overcome if the exposed-terminal problem is also taken into consideration, such that it is possible that no two adjacent links share same time slots. This observation motivated our design to take the hidden- and exposed-terminal problems into consideration. The major study is to design of a QoS multi-path multicast routing protocol under the above considerations.

This work aims to develop a new multicast tree by exploiting the time-slot reuse capability. Before formally defining our multicast-tree structure, a network model is assumed. The MAC sub-layer in our model is implemented by using a TDMA channel model, for which each frame is divided into a control and a data phase, and each data phase of a frame is split into k time slots. Some terms are defined. Let $(h_1, h_2, \ldots, h_k)$ represent a path from nodes h_1 to h_k. A node, B, is said to be a *branch node* if more than one disjoint paths exist from a three-hop neighboring node, B', so that nodes B and B' are called a pair of *branch nodes*.

In this investigation, all nodes are assumed to have the fixed reception and carrier-sense ranges. We introduce the terms *hexagonal-block*, *hexagonal-twin*, *hexagonal-path*, *hexagonal-branch*, and *hexagonal-tree* for constructing our QoS multicast routing protocol.

Definition 1 (Hexagonal-block) Given a pair of two branch nodes, B and B', there are two disjoint three-hop paths (B, X, Y, B') and (B, X', Y', B') between these two branch nodes, where X and X' are not one-hop neighbors and Y and Y' are not one-hop neighbors. Paths (B, X, Y, B') and (B, X', Y', B') form a hexagonal-block, and let $\left[\begin{smallmatrix} B & X & Y \\ & X' & Y' \end{smallmatrix}\ B'\right]$ denote a hexagonal-block between B and B'.

For instance, a hexagonal-block $\left[\begin{smallmatrix} B & X & Y \\ & X' & Y' \end{smallmatrix}\ B'\right]$ is shown in Fig. 2; there are two disjoint three-hop paths (B, X, Y, B') and (B, X', Y', B') between a pair of branch nodes B and B'. In general, nodes X and X' can be a pair of neighbors, and nodes Y and Y' can be a pair of neighbors. However, the

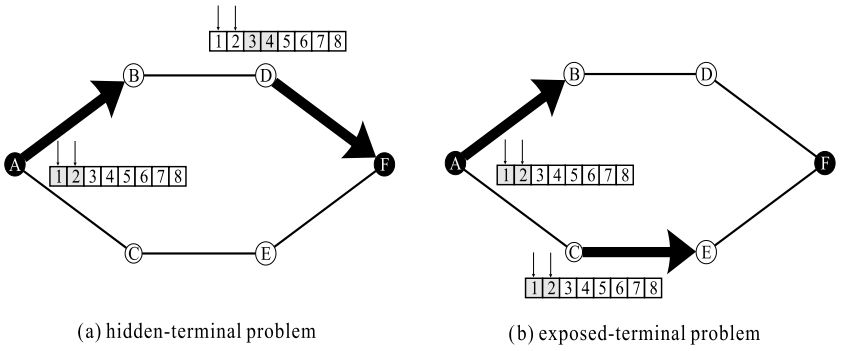


Fig. 1 Example of bandwidth calculations both considering the hidden-terminal and exposed-terminal problems

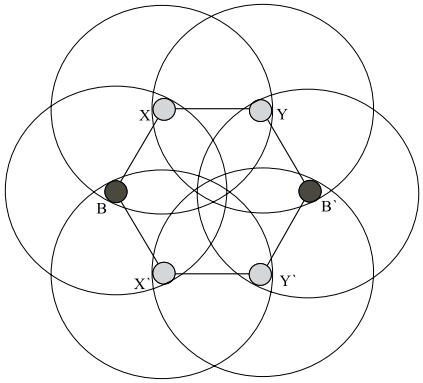


Fig. 2 Example of a hexagonal-block

hexagonal-block gives a strong condition that nodes X and X' cannot be one-hop neighbors and nodes Y and Y' cannot be one-hop neighbors, as illustrated in Fig. 2. This condition is important for the time-slot reservation scheme in order to exploit the time-slot reuse capability.

Using the hexagonal-block allows us to split the original data packet into two equal sub-packets, such that one sub-packet goes through (B, X, Y, B') and the other sub-packet goes through (B, X', Y', B'). The hexagonal-twin is now formally defined as follows. This hexagonal-twin is a fundamental component for constructing the *hexagonal-path* and *hexagonal-tree* structure.

Definition 2 (Hexagonal-twin) Given two hexagonal-blocks,

$$Z = \begin{bmatrix} A & N_1 & \mathbf{N}_3 & \mathbf{B} \\ & N_2 & N_4 & \end{bmatrix} \quad \text{and} \quad Z' = \begin{bmatrix} \mathbf{N}_3 & P_1 & P_2 & C \\ & \mathbf{B} & P_3 & \end{bmatrix},$$

let

$$Z^{Z'} = \begin{bmatrix} & & & P_1 & P_2 & \\ & N_1 & \mathbf{N}_3 & & & C \\ A & & & \mathbf{B} & P_3 & \\ & N_2 & N_4 & & & \end{bmatrix}$$

denote a hexagonal-twin if a link ($\mathbf{N}_3$, $\mathbf{B}$) is shared by Z and Z'.

For instance, a hexagonal-twin

$$Z^{Z'} = \begin{bmatrix} & & & F & H & \\ & B & D & & & J \\ A & & & G & I & \\ & C & E & & & \end{bmatrix}$$

is given in Fig. 3(c); there are two disjoint paths (A, B, D, F, H, J) and (A, C, E, G, I, J) between nodes A and J. Similarly, using the hexagonal-twin, one sub-packet travels along (A, B, D, F, H, J) and the other sub-packet travels along (A, C, E, G, I, J).

Definition 3 (Hexagonal-path) A path is denoted a hexagonal-path if one or more hexagonal-blocks or hexagonal-twins exist in the path.

During QoS route discovery, if a uni-path exists which satisfies a given QoS requirement, a uni-path will be identified. However, if a QoS uni-path does not exist, then the hexagonal-block and hexagonal-twin are used to offer a multi-path routing scheme, which aims to increase the success rate of identifying a QoS route. For instance as shown in Fig. 3(a), a uni-path with two time slots is constructed. If we cannot find a uni-path with two time slots, then a hexagonal-block with two time slots or a hexagonal-twin with two time slots is possibly identified, as shown in Figs. 3(b) and 3(c). Further, Fig. 4(a) shows a hexagonal-path with two non-adjacent hexagonal-blocks connected by a uni-path. Figure 4(b) illustrates a hexagonal-path constructed by two adjacent hexagonal-blocks. Figure 4(c) displays a hexagonal-path constructed by a hexagonal-twin. Note that all cases are instances of hexagonal-paths. We introduce the time-slot reuse capability of a hexagonal-block and hexagonal-twin as follows.

Lemma 1 *A time slot t can be used by a host X to send data to another host Y without causing a collision if all of the following conditions are satisfied*:

1. *Slot t is not yet scheduled to send or receive data in either X or Y.*

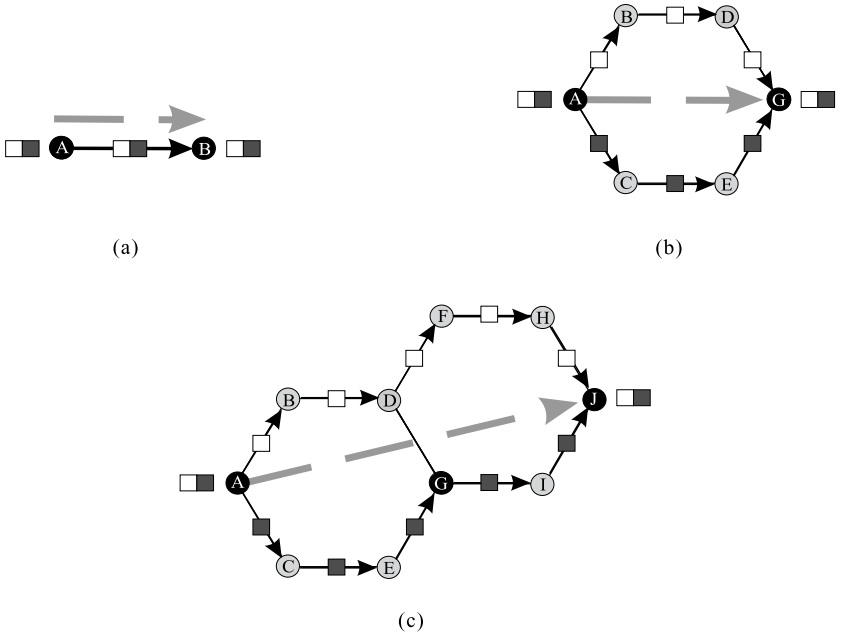


Fig. 3 Example of (**a**) a uni-path, (**b**) a hexagonal-block, and (**c**) a hexagonal-twin

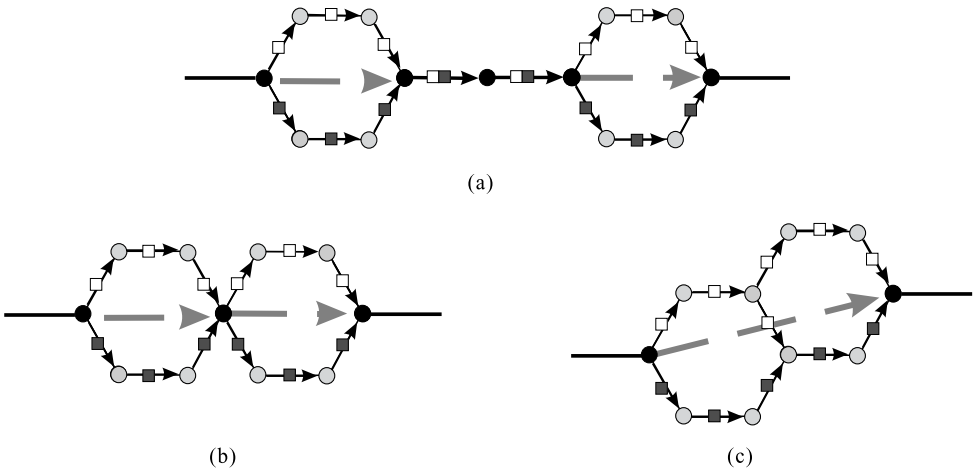


Fig. 4 Examples of hexagonal-paths

2. *For any one-hop neighbor Z of X, slot t is not scheduled to receive data in Z. Slot t can be scheduled* (*or reused*) *to send data in Z, but cannot send data to any of the other one-hop neighbors of X* (*which conflicts with condition* 1).
3. *For any one-hop neighbor Z of Y, slot t is not scheduled to send data in Z. Slot t can be scheduled* (*or reused*) *to receive data in Z, but cannot receive data from any of the other one-hop neighbors of Y* (*which conflicts with condition* 1).

Given the hexagonal-twin

$$\begin{bmatrix} & & G & H & \\ & B & D & & J \\ A & & F & I & \\ & C & E & & \end{bmatrix}$$

as illustrated Fig. 5, the time-slot reuse capability of all links of a hexagonal-twin are listed.

1. Time slot t scheduled in $\overrightarrow{AB}$ can be reused in $\overrightarrow{CE}$, $\overrightarrow{GH}$, and $\overrightarrow{IJ}$ (see Fig. 5(a)).
2. Time slot t scheduled in $\overrightarrow{BD}$ can be reused in $\overrightarrow{AC}$ and $\overrightarrow{JH}$ (see Fig. 5(b)).
3. Time slot t scheduled in $\overrightarrow{DG}$ can be reused in $\overrightarrow{FI}$ (see Fig. 5(c)).
4. Time slot t scheduled in $\overrightarrow{EF}$ can be reused in $\overrightarrow{HJ}$ (see Fig. 5(d)).

To prepare for the construction of a *hexagonal-tree* structure, a hexagonal-branch is formally defined based on the hexagonal-twin structure.

Definition 4 (Hexagonal-branch) Given three hexagonal-blocks,

$$Z = \begin{bmatrix} A & N_1 & N_3 & B \\ & N_2 & N_4 & \end{bmatrix}, \quad Z' = \begin{bmatrix} N_3 & P_1 & P_2 & C \\ & B & P_3 & \end{bmatrix},$$

and

$$Z'' = \begin{bmatrix} N_4 & B & P_3 & D \\ & P_4 & P_5 & \end{bmatrix},$$

let

$$Z \begin{matrix} Z' \\ Z'' \end{matrix} = \begin{bmatrix} & N_1 & \mathbf{N}_3 & P_1 & P_2 & \\ A & & & \mathbf{B} & \mathbf{P}_3 & C \\ & N_2 & \mathbf{N}_4 & P_4 & P_5 & D \end{bmatrix}$$

denote a hexagonal-branch which is constructed by Z, Z', and Z'', for which the three links of ($\mathbf{N}_3$, $\mathbf{B}$), ($\mathbf{N}_4$, $\mathbf{B}$) and ($\mathbf{B}$, $\mathbf{P}_3$) are shared by Z, Z', and Z''.

Given the hexagonal-branch

$$\begin{bmatrix} & B & D & G & H & \\ A & & & F & I & J \\ & C & E & K & L & M \end{bmatrix}$$

illustrated in Fig. 6, the time-slot reuse capability of all links of a hexagonal-branch are listed.

1. Time slot t scheduled in $\overrightarrow{AB}$ can be reused in $\overrightarrow{CE}$, $\overrightarrow{GH}$, $\overrightarrow{IJ}$ and $\overrightarrow{LK}$ (see Fig. 6(a), which is the same as Fig. 5(a)).
2. Time slot t scheduled in $\overrightarrow{AC}$ can be reused in $\overrightarrow{BD}$, $\overrightarrow{JH}$, $\overrightarrow{IM}$ and $\overrightarrow{KL}$ (see Fig. 6(b), which is the same as Fig. 5(b)).
3. Time slot t scheduled in $\overrightarrow{DG}$ can be reused in $\overrightarrow{FI}$ and $\overrightarrow{EK}$ (see Fig. 6(c), which is the same as Fig. 5(c)).
4. Time slot t scheduled in $\overrightarrow{HJ}$ can be reused in $\overrightarrow{FI}$, $\overrightarrow{LM}$, $\overrightarrow{DB}$ and $\overrightarrow{EC}$ (see Fig. 6(d), which differs from Fig. 5(d)).

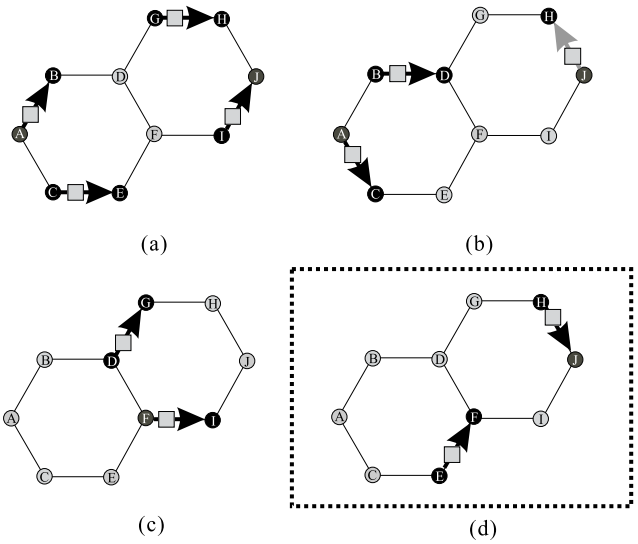


Fig. 5 Time-slot reuse capability of a hexagonal-twin

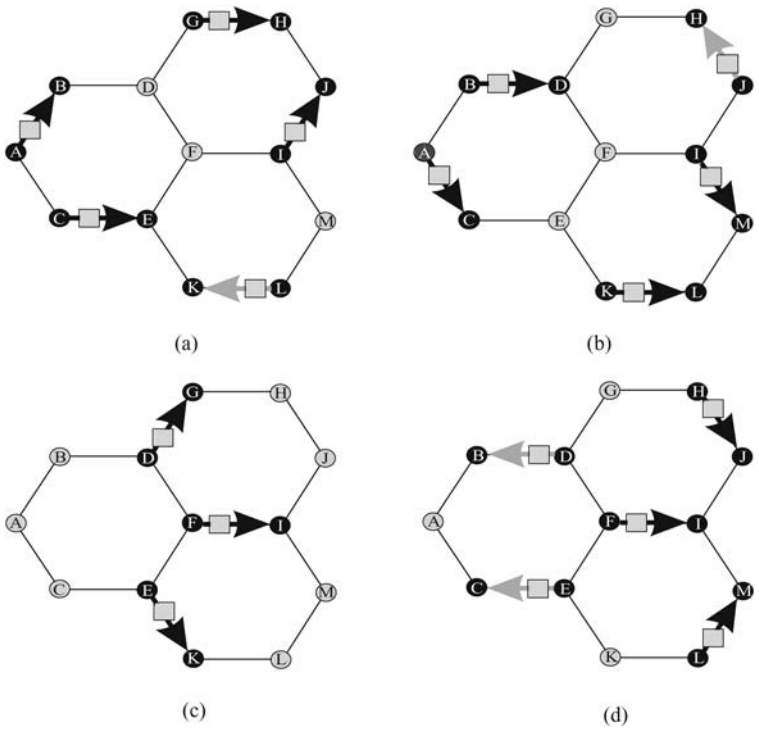
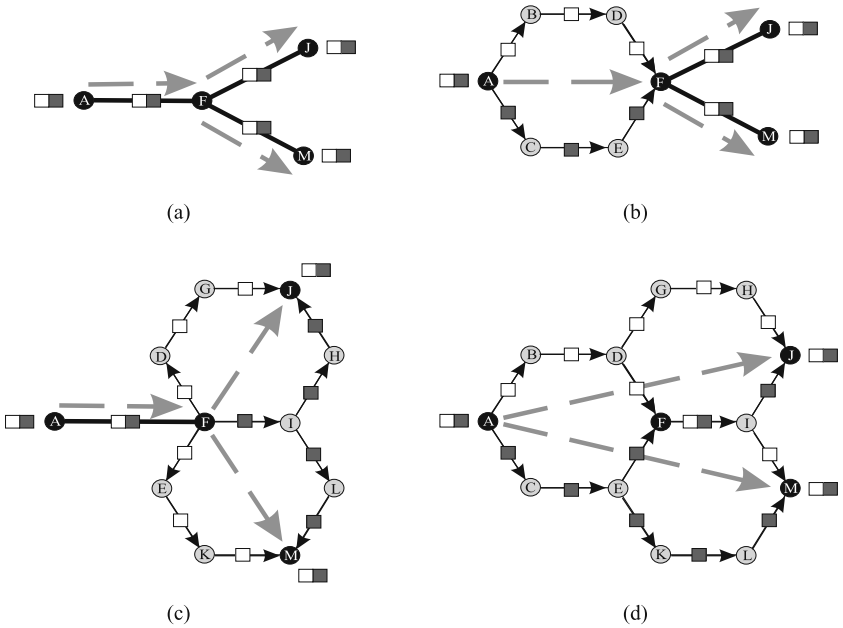


Fig. 6 Time-slot reuse capability of a hexagonal-branch

Fig. 7 Four cases of hexagonal-branches



As shown in Fig. 7(d), the hexagonal-branch

$$Z\begin{matrix}Z'\\Z''\end{matrix}=\begin{bmatrix} & & & G & H & \\ & B & D & & & J \\ A & & & F & I & \\ & C & E & & & M \\ & & & K & L & \end{bmatrix}$$

is constructed by two hexagonal-twins

$$Z\,{}^{Z'}=\begin{bmatrix} & & & G & H & \\ & B & D & & & J \\ A & & & F & I & \\ & C & E & & & \end{bmatrix}$$

and

$$Z_{\,Z''}=\begin{bmatrix} & B & D & & & \\ A & & & F & I & \\ & C & E & & & M \\ & & & K & L & \end{bmatrix}.$$

Observe that an original packet is split into two sub-packets. From host A, the two sub-packets can send data to host J by paths (A, B, D, G, H, J) and (A, C, E, F, I, J). Similarly, the two sub-packets can send data to host M by paths (A, B, D, F, I, M) and (A, C, E, K, L, M). This achieves the purpose of delivering the original message to two distinct hosts. This is the fundamental operation of the tree structure. The hexagonal-branch

$$Z\begin{matrix}Z'\\Z''\end{matrix}=\begin{bmatrix} & & & F & H & \\ & B & D & & & J \\ A & & & G & I & \\ & C & E & & & M \\ & & & K & L & \end{bmatrix}$$

is given in Fig. 7(d), where

$$Z\,{}^{Z'}=\begin{bmatrix} & & & F & H & \\ & B & D & & & J \\ A & & & G & I & \\ & C & E & & & \end{bmatrix}$$

and

$$Z_{\,Z''}=\begin{bmatrix} & B & D & & & \\ A & & & G & I & \\ & C & E & & & M \\ & & & K & L & \end{bmatrix}.$$

Observe that an original packet is split into two sub-packets. From host A, the two sub-packets can traverse to host J by paths (A, B, D, F, H, J) and (A, C, E, G, I, J). Similarly, the two sub-packet can traverse to host M by paths (A, B, D, G, I, M) and (A, C, E, K, L, M). This achieves the purpose of delivering the original message to two distinct hosts. This is the fundamental branch operation of a tree structure.

The purpose of the tree branch is to send the same data packet to two distinct nodes while satisfying the given bandwidth requirement. Each node can continually build a

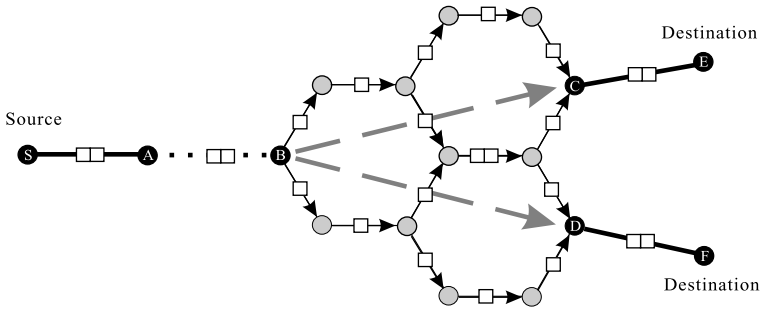


Fig. 8 Example of a hexagonal-tree

sub-tree to cover all possible destination nodes. All possible cases of tree branches are illustrated in Fig. 7, and a tree branch occurs between node A and nodes J and M. Figure 7(a) shows that node A initially sends a packet to nodes J and M using three uni-paths $\overrightarrow{AG}$, $\overrightarrow{GJ}$, and $\overrightarrow{GM}$, with two time-slots. If this case fails, three other cases are then constructed to improve the success rate of constructing a QoS multicast tree. Figure 7(b) illustrates a hexagonal-block $\begin{bmatrix} A & \begin{matrix} B & D \\ C & E \end{matrix} & G \end{bmatrix}$, and two uni-paths $\overrightarrow{GJ}$ and $\overrightarrow{GM}$ using two time-slots. Figure 7(c) displays a uni-path $\overrightarrow{AG}$ and two hexagonal-blocks which uses two time-slots. Figure 7(c) shows a hexagonal-branch which uses two time-slots. In addition, Fig. 7(d) shows the worst case of constructing a hexagonal-branch. Observe that, there is one path which requires the allocation of the total bandwidth demand (time-slots) instead of the halves used on other links. An instance is shown in link $\overrightarrow{FI}$. It still can reduces the time-slot requirement in links $\overrightarrow{IJ}$ and $\overrightarrow{IM}$. This condition only occurs in three hexagonal-blocks are simultaneously used for a hexagonal-branch. With the hexagonal-path and hexagonal-branch, the hexagonal-tree is formally defined as follows.

Definition 5 (Hexagonal-tree) A tree is said to be a hexagonal-tree if one or more hexagonal-path exist in the tree.

A possible hexagonal-tree, from source host S to destinations E and F, with uni-paths is illustrated in Fig. 8; note that a hexagonal-branch appears between node B and nodes C and D. Observe that, the uni-path which satisfies the QoS requirement is identified with a higher priority during construction of the multicast tree. In the following, the construction of a hexagonal-tree is presented.

3 Hexagonal-tree TDMA-based QoS multicast routing protocol

We first give an overview of our proposed protocol, a hexagonal-tree TDMA-based QoS multicast protocol, in a MANET. The proposed protocol mainly constructs a hexagonal-tree to perform the on-demand QoS multicast routing operation. The designed protocol is achieved by the *hexagonal-branch/twin identification* and *hexagonal-tree construction* phases. The *hexagonal-branch/twin identification* phase identifies the hexagonal-branch/twin in a MANET. The *hexagonal-tree construction* phase constructs the hexagonal-tree by merging hexagonal-paths from a source to all destinations.

3.1 Phase 1: hexagonal-branch/twin identification

3.1.1 Hexagonal-branch identification

This phase searches for a hexagonal-branch under a given bandwidth requirement. A hexagonal-branch,

$$\begin{bmatrix} & & & G & H & \\ & B & D & & & J \\ A & & & F & I & \\ & C & E & & & M \\ & & & K & L & \end{bmatrix},$$

is constructed by the three hexagonal-blocks of

$$\begin{bmatrix} A & \begin{matrix} B & D \\ C & E \end{matrix} & F \end{bmatrix}, \quad \begin{bmatrix} D & \begin{matrix} G & H \\ F & I \end{matrix} & J \end{bmatrix},$$

and $\begin{bmatrix} E & \begin{matrix} F & I \\ K & L \end{matrix} & M \end{bmatrix}$.

Node F contains the following data structure if we attempt to successfully identify a hexagonal-branch. Observe that each node maintains local information on all one/two/three-hop neighboring nodes:

- $node_x$. Let $node_x \in \{A, B, C, D, E, F, G, H, I, J, K, L, M\}$ and all one- or two-hop neighbors of $\{A, B, C, D, E, F, G, H, I, J, K, L, M\}$.
- A list of sending activities of $node_x$. Node $node_x$ records on which time slots it has sending activities. If $[y, l_1, \ldots, l_k]$ denotes $node_x$ sending data to node y on time slots $\{l_1, \ldots, l_k\}$, then a list of $[y, l_1, \ldots, l_k]$ is maintained.

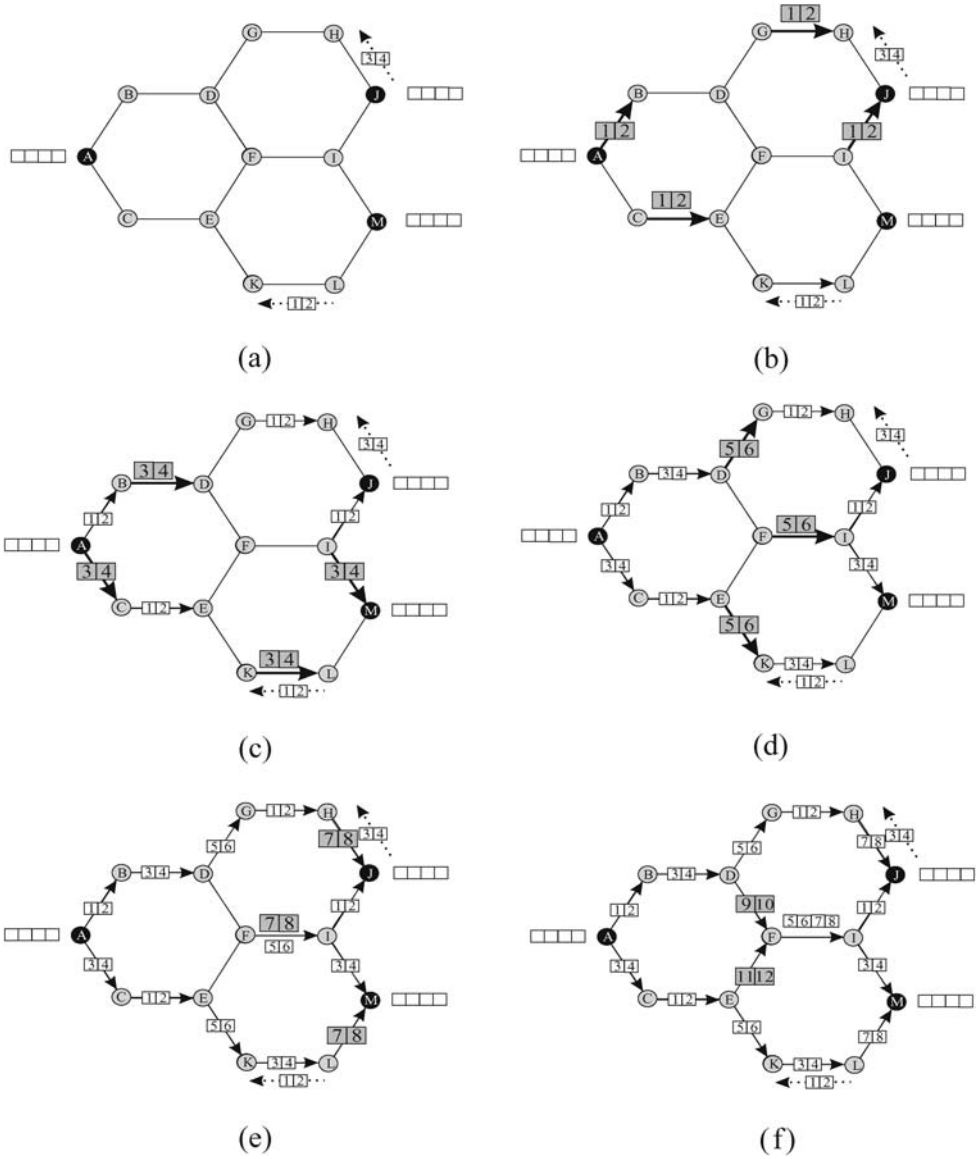


Fig. 9 Time-slot reservation for a hexagonal-branch

- A list of receiving activities of $node_x$. Node $node_x$ records from which time slots it has receiving activities. If $[y, l_1, \ldots, l_k]$ denotes $node_x$ receiving data from node y on time slots $\{l_1, \ldots, l_k\}$, then a list of $[y, l_1, \ldots, l_k]$ is maintained.

Observe that this section only discusses how to successfully identify a hexagonal-branch to exploit the time-slot reuse capability. Observe that all records are collected into node F, which indicates that the time slot reservation of paths from A to J and A to M are determined by node F. This is the main overhead of our proposed protocol. From the simulation results in Sect. 4, the improved success rate of identifying a QoS on-demand multicast tree can successfully cover this extra overhead. Let Free_time_slot ($\overrightarrow{\alpha\beta}$) denote time slots which can be used to send data from α to β, Used_time_slot ($\overrightarrow{\alpha\beta}$) denote time slots which have been used to send data from α to β. For instance as illustrated in Fig. 9(a), Used_time_slot($\overrightarrow{LK}$) = {1, 2}.

Given a hexagonal-branch,

$$\begin{bmatrix} & & & G & H & \\ & B & D & & & J \\ A & & & F & I & \\ & C & E & & & M \\ & & & K & L & \end{bmatrix},$$

as shown in Fig. 9(a), $\overrightarrow{AB}$ and $\overrightarrow{AC}$ cannot share the same time slots, since our protocol is a multi-path (two-path). Based on Fig. 6(a), we do know that Free_time_slot ($\overrightarrow{AB}$) can be the same as Free_time_slot ($\overrightarrow{CE}$), Free_time_slot ($\overrightarrow{GH}$), Free_time_slot ($\overrightarrow{IJ}$), and Free_time_slot ($\overrightarrow{LK}$), but time slots scheduled in $\overrightarrow{AB}$ cannot be used in $\overrightarrow{FD}$. Similarly, from Fig. 6(b), Free_time_slot ($\overrightarrow{AC}$) can be

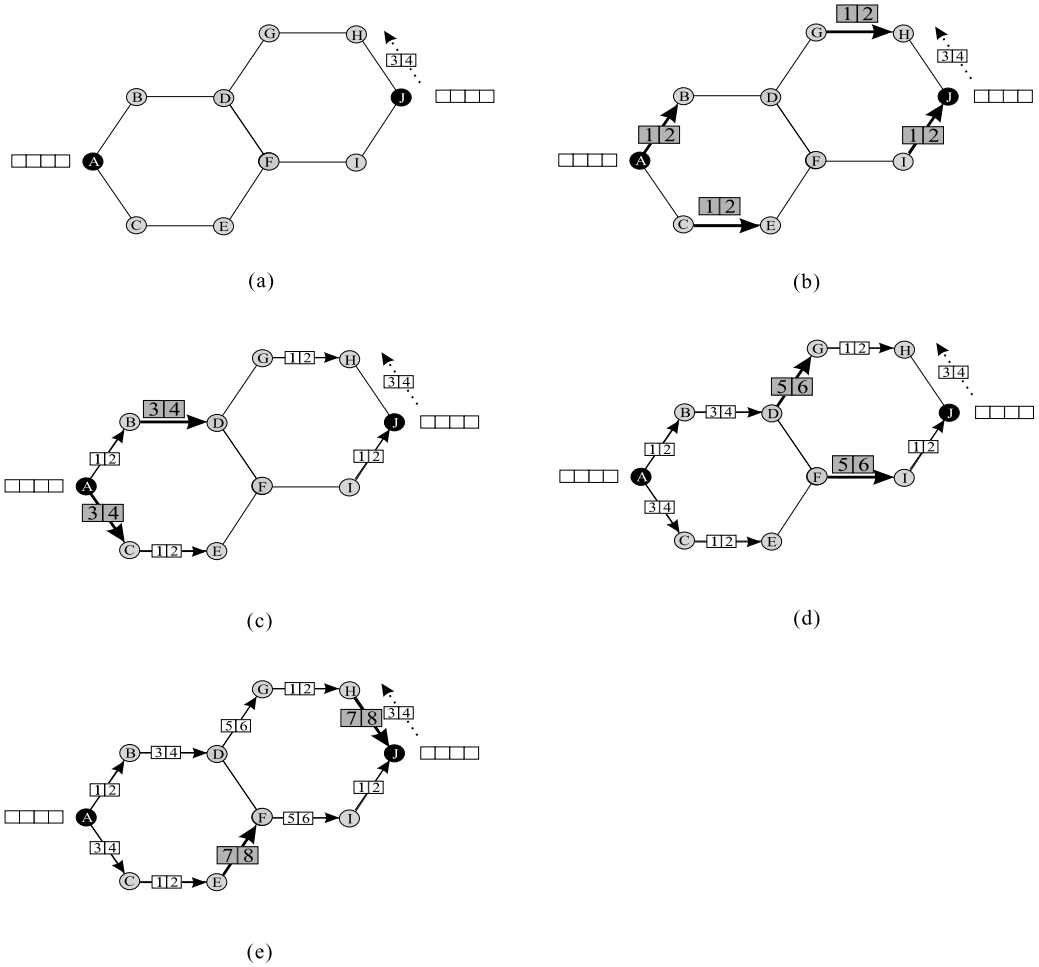


Fig. 10 Time slot reservation for a hexagonal-twin

the same as Free_time_slot ($\overrightarrow{BD}$), Free_time_slot ($\overrightarrow{IM}$), Free_time_slot ($\overrightarrow{KL}$), and Free_time_slot ($\overrightarrow{JH}$), but time slots scheduled in $\overrightarrow{AC}$ cannot be used in $\overrightarrow{FE}$, etc.

With a bandwidth requirement of γ, we perform the following reservation operations: (1) We try to allocate $\gamma/2$ time slots to $\overrightarrow{AB}$, $\overrightarrow{CE}$, $\overrightarrow{GH}$, and $\overrightarrow{IJ}$ (see Fig. 6(a)). (2) We then try to allocate $\gamma/2$ time slots to $\overrightarrow{AC}$, $\overrightarrow{BD}$, $\overrightarrow{IM}$, and $\overrightarrow{KL}$ (see Fig. 6(b)). (3) We then attempt to allocate $\gamma/2$ time slots to $\overrightarrow{HJ}$, $\overrightarrow{FI}$, and $\overrightarrow{LM}$ (see Fig. 6(c)). (4) We then attempt to allocate $\gamma/2$ time slots to $\overrightarrow{DG}$, $\overrightarrow{FI}$, and $\overrightarrow{EK}$ (see Fig. 6(d)). (5) We finally try to allocate $\gamma/2$ time slots to $\overrightarrow{DF}$ and $\overrightarrow{EF}$. If any one of these allocations fails, then the time slot reservation for a hexagonal-branch fails. Assuming that the bandwidth requirement is γ, the formal reservation procedure is given:

A1. Repeatedly reserve $\gamma/2-\alpha$ identical time slots and α distinct time slots on $\overrightarrow{AB}$, $\overrightarrow{CE}$, $\overrightarrow{GH}$, and $\overrightarrow{IJ}$, for $\alpha=0$ to $\gamma/2$, until the reservation is successful, and update the possible nodes' records of sending and receiving activities. Otherwise, if the reservation fails, then the reservation for a hexagonal-branch fails, and exit the procedure (see Fig. 6(a)).

A2. Repeatedly reserve $\gamma/2-\alpha$ identical time slots and α distinct time slots on $\overrightarrow{AC}$, $\overrightarrow{BD}$, $\overrightarrow{IM}$, and $\overrightarrow{KL}$, for $\alpha=0$ to $\gamma/2$, until the reservation is successful, and update the possible nodes' records of sending and receiving activities. Otherwise, if the reservation fails, then the reservation for a hexagonal-branch fails, and exit the procedure (see Fig. 6(b)).

A3. Repeatedly reserve $\gamma/2-\alpha$ identical time slots and α distinct time slots on $\overrightarrow{DG}$, $\overrightarrow{FI}$, and $\overrightarrow{EK}$, for $\alpha=0$ to $\gamma/2$, until the reservation is successful, and update the possible nodes' records of sending and receiving activities. Otherwise, if the reservation fails, then the reservation for a hexagonal-branch fails, and exit the procedure (see Fig. 6(c)).

A4. Repeatedly reserve $\gamma/2-\alpha$ identical time slots and α distinct time slots on $\overrightarrow{HJ}$, $\overrightarrow{FI}$, and $\overrightarrow{LM}$, for $\alpha=0$ to $\gamma/2$, until the reservation is successful, and update the possible nodes' records of sending and receiving activities. Otherwise, if the reservation fails, then the reser-

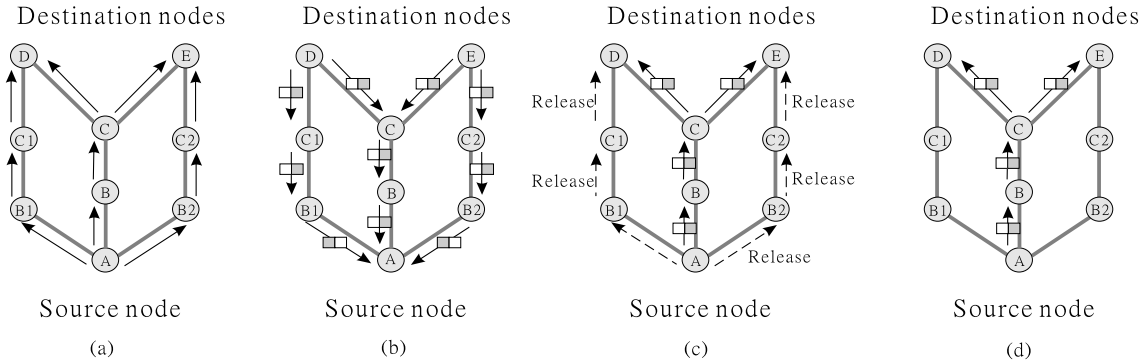


Fig. 11 Merging steps of the hexagonal-tree

vation for a hexagonal-branch fails, and exit the procedure (see Fig. 6(d)).

A5. Repeatedly reserve $\gamma/2-\alpha$ identical time slots and α distinct time slots on $\overrightarrow{DF}$ and $\overrightarrow{EF}$, for $\alpha=0$ to $\gamma/2$, until the reservation is successful, and update the possible nodes' records of sending and receiving activities. Otherwise, if the reservation fails, then the reservation for a hexagonal-branch fails, and exit the procedure.

For instance as shown in Fig. 9(a), let Used_time_slot($\overrightarrow{JH}$) = {3, 4} and Used_time_slot($\overrightarrow{LK}$) = {1, 2} before constructing the hexagonal-branch. Let bandwidth requirement γ be four slots. Figure 9(b) shows that slots {1, 2} are successfully allocated to $\overrightarrow{AB}$, $\overrightarrow{CE}$, $\overrightarrow{GH}$, and $\overrightarrow{IJ}$ for step A1. Figure 9(c) shows that slots {3, 4} are successfully allocated to $\overrightarrow{AC}$, $\overrightarrow{BD}$, $\overrightarrow{IM}$, and $\overrightarrow{KL}$ for step A2. Figure 9(d) shows that slots {5, 6} are successfully allocated to $\overrightarrow{DG}$, $\overrightarrow{FI}$, and $\overrightarrow{EK}$ for step A3. Figure 9(e) illustrates that slots {7, 8} are successfully allocated to $\overrightarrow{HJ}$, $\overrightarrow{FI}$, and $\overrightarrow{LM}$ for step A4. Figure 9(f) illustrates that slots {9, 10} and {11, 12} are respectively allocated to $\overrightarrow{DF}$ and $\overrightarrow{EF}$ for step A5. Finally, a hexagonal-branch with four time slots is successfully constructed.

3.1.2 Hexagonal-twin construction

This phase searches for a hexagonal-twin if the given bandwidth requirement is γ. A hexagonal-twin, $Z^{Z'}$ or $Z_{Z''}$, is constructed from a hexagonal-branch, $Z_{Z''}^{Z'}$, without the hexagonal-block Z' or Z''. The operation is similar to the identification of the hexagonal-branch. We omit the details herein. For instance as shown in Fig. 10(a), let Used_time_slot($\overrightarrow{JH}$) = {3, 4} before constructing the hexagonal-twin. Let the bandwidth requirement γ be four slots. Figure 10(b) shows that slots {1, 2} are successfully allocated to $\overrightarrow{AB}$, $\overrightarrow{CE}$, $\overrightarrow{GH}$, and $\overrightarrow{IJ}$ (the same as step A1, and see Fig. 5(a)). Figure 10(c) shows that slots {3, 4} are successfully allocated to $\overrightarrow{AC}$ and $\overrightarrow{BD}$ (the same as step A2, and see Fig. 5(b)). Figure 10(d) shows that slots {5, 6} are successfully allocated to $\overrightarrow{DG}$ and $\overrightarrow{FI}$ (the same as step A3, and see Fig. 5(c)). Figure 10(e) illustrates that slots {7, 8} are successfully allocated to $\overrightarrow{EF}$ and $\overrightarrow{HJ}$ (see Fig. 5(d)). Finally, a hexagonal-twin with four time slots is successfully constructed.

3.2 Phase 2: hexagonal-tree structure

The hexagonal-tree construction phase is divided into three operations:

1. *The hexagonal-path discovery operation*. Based on the identified hexagonal-branches and hexagonal-twins, many hexagonal-paths from a source to a given set of destinations are constructed (see Fig. 11(a)).
2. *The hexagonal-path reply operation*. The destination receives the route-request packet from a source, and replies with a route-reply packet to the source (see Fig. 11(b)).
3. *The hexagonal-tree construction operation*. Multiple hexagonal-paths are received from all destinations, and the hexagonal-tree is finally established at the source (see Fig. 11(c)). Finally, the source sends a data packet according to the determined hexagonal-tree as shown in Fig. 11(d).

3.2.1 The hexagonal-path discovery operation

After constructing hexagonal-branches and hexagonal-twins, we now describe how to construct the hexagonal-path. Let $[\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_i](b)$ denote a hexagonal-path, where α_i is a uni-path $\overrightarrow{B_i B_i'}$, a hexagonal-block $\begin{bmatrix} B_i & {\scriptstyle X_i}\ {\scriptstyle X_i'} & {\scriptstyle Y_i}\ {\scriptstyle Y_i'} & B_i' \end{bmatrix}$, or

$$\begin{bmatrix} & & & \widehat{X}_i & \widehat{Y}_i & \\ & X_i & Y_i & & & B_i' \\ B_i & & & \widehat{X}_i' & \widehat{Y}_i' & \\ & X_i' & Y_i' & & & \end{bmatrix}$$

or

$$\begin{bmatrix} & X_i & Y_i & & & \\ B_i & & & \widehat{X}_i & \widehat{Y}_i & \\ & X_i' & Y_i' & & & B_i' \\ & & & \widehat{X}_i' & \widehat{Y}_i' & \end{bmatrix},$$

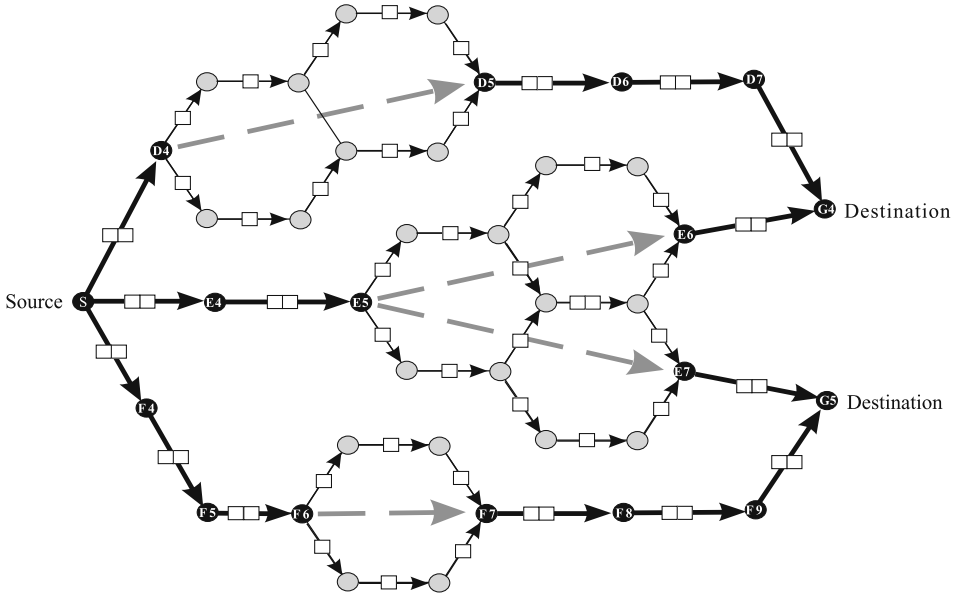


Fig. 12 Hexagonal-tree discovery

where b represents the hexagonal-path bandwidth. The hexagonal-path discovery algorithm is given:

B1. The source node $B_{i=1}$ or node $B_{i>1}$ floods a hexagonal-path request packet into a MANET to check if there is a uni-path α_i from node B_i which satisfies the required bandwidth b, then a hexagonal sub-path $\alpha_i = \overrightarrow{B_i B_i'}$ is identified, where $i \geq 1$.

B2. If no uni-path exists, then node B_i further checks to see if one or more hexagonal-block exist from node B_i. If a hexagonal-block exists which satisfies the required bandwidth b, then a hexagonal sub-path $\alpha_i = \begin{bmatrix} B_i & \begin{matrix} X_i & Y_i \\ X_i' & Y_i' \end{matrix} & B_i' \end{bmatrix}$ is constructed.

B3. If no hexagonal-block exists, then node B_i further checks to see if one or more hexagonal-branches exist from node B_i. If a hexagonal-branch exists which satisfies the required bandwidth b, then

$$\alpha_i = \begin{bmatrix} & & & \widehat{X}_i & \widehat{Y}_i & \\ & X_i & Y_i & & & B_i' \\ B_i & & & \widehat{X}_i' & \widehat{Y}_i' & \\ & X_i' & Y_i' & & & \end{bmatrix}$$

or

$$\begin{bmatrix} & X_i & Y_i & & & \\ B_i & & & \widehat{X}_i & \widehat{Y}_i & \\ & X_i' & Y_i' & & & B_i' \\ & & & \widehat{X}_i' & \widehat{Y}_i' & \end{bmatrix}$$

is constructed.

B4. If step B3 fails, then the searching operation of a hexagonal-path is stopped and exit the procedure. If B_i' is not the destination node, then add hexagonal sub-path α_i into the current hexagonal-path $[\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_{i-1}](b)$ and let $B_{i+1} = B_i'$, node B_{i+1} recursively performs steps B1, B2, and B3 until a hexagonal-path $[\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_k](b)$ is identified, where B_k' is the destination node.

For instance, two hexagonal-paths from source S to two different destination nodes $D1$ and $D2$ are given in Fig. 12.

3.2.2 *The hexagonal-path reply operation*

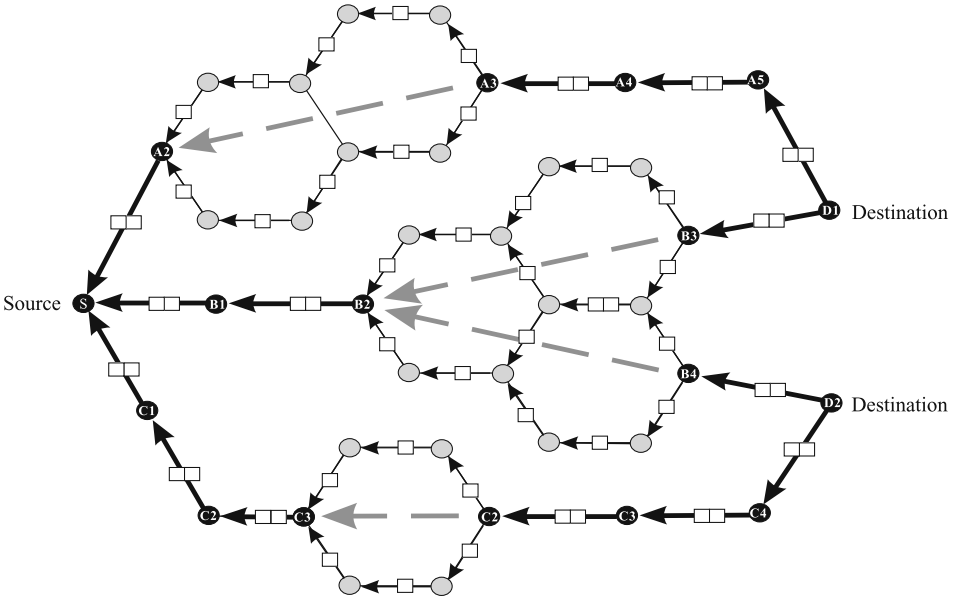
Each destination node replies to all possible hexagonal-paths $[\alpha_i, \alpha_{i-1}, \alpha_{i-2}, \ldots, \alpha_1]$ to the source node. For instance, some hexagonal-paths reply from destination nodes $D1$ and $D2$ to source S, as illustrated in Fig. 13. Many hexagonal-paths will be collected at the source node, and then a constructing hexagonal-tree operation is eventually performed at the source node as follows.

3.2.3 *The hexagonal-tree construction operation*

This study mainly constructs a hexagonal tree which is modified from the spiral-fat-tree on-demand multicast (SOM) protocol [8]. Observe that, the construction of the multicast tree can use the results of [2, 6, 20–22]. In this work, all spiral-paths of a spiral-fat-tree [8] are replaced by hexagonal-paths for the purpose of providing the QoS multicast protocol.

Given that two hexagonal-paths, $[\alpha_1, \alpha_2, \alpha_p, \alpha_{p+1}, \ldots, \alpha_k](b)$, and $[\alpha_1, \alpha_2, \alpha_p, \alpha'_{p+1}, \ldots, \alpha'_k](b)$, have the same hexagonal sub-path $[\alpha_1, \alpha_2, \ldots, \alpha_p](b)$, we denote $\cap([\alpha_1, \alpha_2, \alpha_p, \alpha_{p+1}, \ldots, \alpha_k](b), [\alpha_1, \alpha_2, \alpha_p, \alpha'_{p+1}, \ldots, \alpha'_k](b)) = [\alpha_1, \alpha_2, \ldots, \alpha_p](b)$. Further, let $[\alpha_1, \alpha_2, \ldots, \alpha_p](b, s)$ denote s hexagonal-paths with the same $[\alpha_1, \alpha_2, \ldots, \alpha_p](b)$. We also denote $|[\alpha_1, \alpha_2, \ldots, \alpha_p](b)| = p$ to be the shared hexagonal-path length. Consider s and s' hexagonal-paths

Fig. 13 Hexagonal-tree reply



from all reply hexagonal-paths, we have the merging criterion to construct the hexagonal-tree according to values of p, s, and b:

C1 If the s reply hexagonal-paths have maximum values of p, s, and b, then these s hexagonal-paths are merged together with the highest priority.

C2 If there are s and s' reply hexagonal-paths with the same values of p and s, then s hexagonal-paths with the greater value of b are merged together.

C3 If there are s and s' reply hexagonal-paths with the same values of b and s, then s hexagonal-paths with the greater value of p are merged together.

C4 If there are s and s' reply hexagonal-paths with the same values of b and p, then s hexagonal-paths with the greater value of s are merged together.

C5 If there are s and s' reply hexagonal-paths with the same value of p, then s hexagonal-paths with the greater value of b are merged together.

C6 If there are s and s' reply hexagonal-paths with the same value of s, then s hexagonal-paths with the greater value of b are merged together.

C7 If there are s and s' reply hexagonal-paths with the same value of b, then s hexagonal-paths with the greater value of s are merged together.

C8 If there are s and s' reply hexagonal-paths with the different value of p, b, and s, then s hexagonal-paths with the greater value of b are merged together.

For example, Figs. 14(a) and 14(b) are instances of the case C3, Figs. 14(b) and 14(c) are instances of case C7, Figs. 14(c) and 14(d) are instances of case C2, and Figs. 14(b) and 14(d) are instances of case C6.

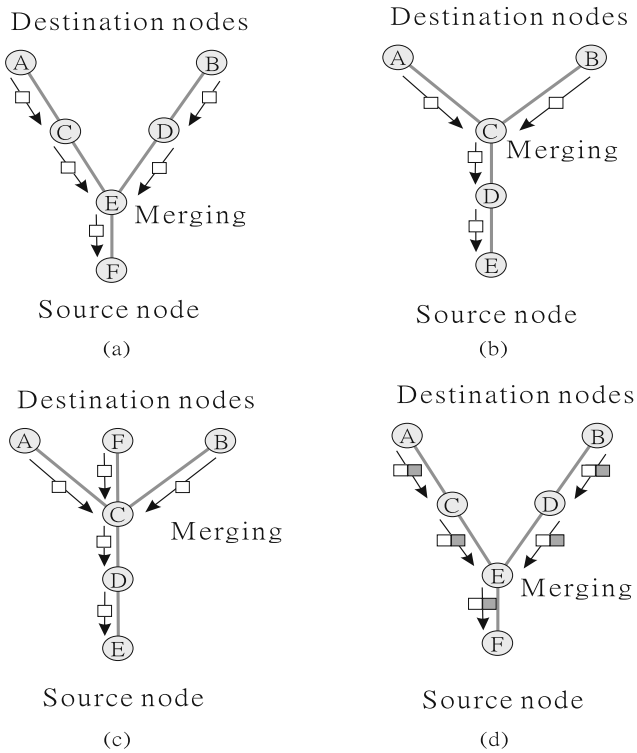


Fig. 14 Merging examples

3.3 Phase 3: hexagonal-tree maintenance

Given a hexagonal path $[\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_i](b)$ denote a hexagonal-path of a constructed hexagonal-tree, where α_i is a uni-path $\overrightarrow{B_i B_i'}$, or a hexagonal-block

$$\begin{bmatrix} B_i & X_i & Y_i & B_i' \\ & X_i' & Y_i' & \end{bmatrix},$$

$$\text{or} \quad \begin{bmatrix} & & & \widehat{X}_i & \widehat{Y}_i & \\ & X_i & Y_i & & & B'_i \\ B_i & & & \widehat{X}'_i & \widehat{Y}'_i & \\ & X'_i & Y'_i & & & \end{bmatrix}$$

$$\text{or} \quad \begin{bmatrix} & X_i & Y_i & & & \\ B_i & & & \widehat{X}_i & \widehat{Y}_i & \\ & X'_i & Y'_i & & & B'_i \\ & & & \widehat{X}'_i & \widehat{Y}'_i & \end{bmatrix}.$$

Without loss of generality, let $\begin{bmatrix} B_i & \begin{smallmatrix} X_i & Y_i \\ X'_i & Y'_i \end{smallmatrix} & B'_i \end{bmatrix}$ be the ith hexagonal-block, and

$$\begin{bmatrix} & & & \widehat{X}_i & \widehat{Y}_i & \\ & X_i & Y_i & & & B'_i \\ B_i & & & \widehat{X}'_i & \widehat{Y}'_i & \\ & X'_i & Y'_i & & & \end{bmatrix}$$

$$\text{or} \quad \begin{bmatrix} & X_i & Y_i & & & \\ B_i & & & \widehat{X}_i & \widehat{Y}_i & \\ & X'_i & Y'_i & & & B'_i \\ & & & \widehat{X}'_i & \widehat{Y}'_i & \end{bmatrix}$$

be the ith hexagonal-twin. If sub-path $[B_i\, X'_i\, Y'_i\, B'_i]$ in $\begin{bmatrix} B_i & \begin{smallmatrix} X_i & Y_i \\ X'_i & Y'_i \end{smallmatrix} & B'_i \end{bmatrix}$ fails, then a backup path $[B_i\, X''_i\, Y''_i\, B'_i]$ in $\begin{bmatrix} B_i & \begin{smallmatrix} X_i & Y_i \\ X''_i & Y''_i \end{smallmatrix} & B'_i \end{bmatrix}$ is constructed to replace with the failed sub-path $[B_i\, X'_i\, Y'_i\, B'_i]$. If sub-path $[B_i\, X'_i\, Y'_i\, \widehat{X}'_i]$ or $[Y_i\, \widehat{X}'_i\, \widehat{Y}'_i\, B'_i]$ fails in

$$\begin{bmatrix} & & & \widehat{X}_i & \widehat{Y}_i & \\ & X_i & Y_i & & & B'_i \\ B_i & & & \widehat{X}'_i & \widehat{Y}'_i & \\ & X'_i & Y'_i & & & \end{bmatrix},$$

then a backup path $[B_i\, X''_i\, Y''_i\, \widehat{X}'_i]$ or $[Y_i\, \widehat{X}''_i\, \widehat{Y}''_i\, B'_i]$ is constructed to replace with the failed sub-path $[B_i\, X'_i\, Y'_i\, \widehat{X}'_i]$ or $[Y_i\, \widehat{X}'_i\, \widehat{Y}'_i\, B'_i]$. In addition, if sub-path $[B_i\, X'_i\, Y'_i\, \widehat{X}_i]$ or $[Y'_i\, \widehat{X}'_i\, \widehat{Y}'_i\, B'_i]$ fails in

$$\begin{bmatrix} & X_i & Y_i & & & \\ B_i & & & \widehat{X}_i & \widehat{Y}_i & \\ & X'_i & Y'_i & & & B'_i \\ & & & \widehat{X}'_i & \widehat{Y}'_i & \end{bmatrix},$$

then we try to search for a backup path $[B_i\, X''_i\, Y''_i\, \widehat{X}_i]$ or $[Y'_i\, \widehat{X}''_i\, \widehat{Y}''_i\, B'_i]$ to replace with the failed sub-path $[B_i\, X'_i\, Y'_i\, \widehat{X}_i]$ or $[Y'_i\, \widehat{X}'_i\, \widehat{Y}'_i\, B'_i]$.

3.4 The algorithm complexity

Before discussing the time complexity of constructing the hexagonal tree, the time cost and complexity of a hexagonal-block, a hexagonal-twin, and a hexagonal-path are given as follows.

Lemma 2 *The time cost of constructing a hexagonal-block is* $T_{\text{block}} = 4T_\alpha$.

Proof Given a hexagonal-block $\begin{bmatrix} A & \begin{smallmatrix} B & D \\ C & E \end{smallmatrix} & F \end{bmatrix}$, and let T_α denote the time cost of sending a packet between a link $\overrightarrow{XY}$ in the hexagonal-block. Node A broadcasts the hexagonal-block request packet to nodes B, C simultaneously within T_α. Then, nodes B and C simultaneously send hexagonal-block request packets to nodes D and E, respectively within T_α. To present the collision, nodes D and E transmit hexagonal-block request packets to node F in different time slots. Let T_{block} denote the time cost of a hexagonal-block. Therefore, the time cost of constructing a hexagonal-block is $T_{\text{block}} = 4T_\alpha$. □

Lemma 3 *The time cost of constructing a hexagonal-twin is* $T_{\text{twin}} = 6T_\alpha$.

Proof For a hexagonal-twin

$$\begin{bmatrix} & & & G & H & \\ & B & D & & & J \\ A & & & F & I & \\ & C & E & & & \end{bmatrix},$$

which is constructed by two hexagonal-blocks $\begin{bmatrix} A & \begin{smallmatrix} B & D \\ C & E \end{smallmatrix} & F \end{bmatrix}$ and $\begin{bmatrix} D & \begin{smallmatrix} G & H \\ F & I \end{smallmatrix} & J \end{bmatrix}$. Based on result from Lemma 2, nodes G and F received the hexagonal-twin request packet from node D at time $3T_\alpha$, then forward the packet to node H, while node H received the packet at time $4T_\alpha$. To avoid the collision, node J received packets from nodes H and I at $5T_\alpha$ and $6T_\alpha$, respectively. Let T_{twin} denote the time cost of a hexagonal-twin. Therefore, the time cost of constructing a hexagonal-twin is $T_{\text{twin}} = 6T_\alpha$. □

Lemma 4 *The time complexity of constructing a hexagonal-path is* $T_{\text{path}} = O(5mT_\alpha)$ *if* $m = p + q$ *and* $N_{\text{path}} = (5p + 1) + (9q + 1)$, *where* N_{path} *is he total number of nodes in a hexagonal-path.*

Proof For the worst case, we assumed that a hexagonal-path consists of m segments, where these m segments contain p hexagonal-blocks and q hexagonal-twins, $m = p + q$, as shown in Fig. 18(a). This is because that the time cost of a hexagonal-block and a hexagonal-twin are larger than that of a uni-path. Let N_{path} denote the node number of a hexagonal-path, where $N_{\text{path}} = (5p + 1) + (9q + 1)$, 5 and 9 are node number of a hexagonal-block and a hexagonal-

twin, respectively. Therefore, there are $m+1$ various cases as follows:

$$\begin{cases} \textit{case } 1\text{: } m \textit{ hexagonal-blocks and } 0 \textit{ hexagonal-twin} \\ \textit{case } 2\text{: } m-1 \textit{ hexagonal-blocks and } 1 \textit{ hexagonal-twin} \\ \vdots \\ \textit{case } m\text{: } 1 \textit{ hexagonal-blocks and } m-1 \textit{ hexagonal-twin} \\ \textit{case } m+1\text{: } 0 \textit{ hexagonal-blocks and } m \textit{ hexagonal-twin} \end{cases}$$

Let T_{path} denote the time cost of hexagonal-path, which is calculated as follows.

$$\begin{aligned} T_{\text{path}} &= \frac{\frac{(0+m)(m+1)}{2} T_{\text{block}} + \frac{(m+0)(m+1)}{2} T_{\text{twin}}}{m+1} \\ &= \frac{\frac{(0+m)(m+1)}{2} \times 4T_\alpha + \frac{(m+0)(m+1)}{2} \times 6T_\alpha}{m+1} = 5mT_\alpha, \end{aligned}$$

where $T_{\text{block}} = 4T_\alpha$, $T_{\text{twin}} = 6T_\alpha$. Therefore, the average time complexity of constructing a hexagonal-path is $T_{\text{path}} = O(5mT_\alpha)$. □

Lemma 5 *The time complexity of constructing a hexagonal-tree is* $O(2k_1(7+5m)T_\alpha) \le T_{\text{tree}} \le O(7T_\alpha k_2 + 5mT_\alpha(k_2-1))$, *where* $k_1 = \log_2(\frac{N_{\text{tree}}}{N_{\text{path}}+13} + 1)$ *and* $k_2 = \frac{N_{\text{tree}}+N_{\text{path}}}{N_{\text{path}}+13}$.

Proof Based on Lemma 3, and consider a hexagonal-branch

$$\begin{bmatrix} & & & G & H & \\ & B & D & & & J \\ A & & & F & I & \\ & C & E & & & \mathbf{M} \\ & & & \mathbf{K} & \mathbf{L} & \end{bmatrix},$$

which is constructed by hexagonal-blocks

$$\begin{bmatrix} A & B & D & F \\ & C & E & \end{bmatrix}, \quad \begin{bmatrix} D & G & H & J \\ & F & I & \end{bmatrix},$$

and $\begin{bmatrix} E & F & I & M \\ & K & L & \end{bmatrix}$.

It is easily to develop the time cost of constructing a hexagonal-branch is $T_{\text{branch}} = 7T_\alpha$, by adding nodes K, L, and M.

To construct a hexagonal-tree, the hexagonal-tree is a complete binary tree or a skew tree for the best and worst cases, respectively. Let N_{tree} be the total number of nodes of the tree and the node number in a hexagonal-branch is 13. In a hexagonal-tree, we assumed that any hexagonal-path between two hexagonal-branches has at most m segments, where N_{path} is the node number of the hexagonal-path. For the best case, a k_1-level complete binary tree can be constructed, where $k_1 = \log_2(\frac{N_{\text{tree}}}{N_{\text{path}}+13} + 1)$ and $N_{\text{tree}} = 13 \times (2^{k_1} - 1) + (2^{k_1} - 1) \times N_{\text{path}}$. Example is given in Fig. 18(b). Let T_{tree} denote the time cost of hexagonal-branch and hexagonal-tree. The time cost of a k_1-level complete binary hexagonal-tree is

$$2k_1 \times (T_{\text{branch}} + T_{\text{path}}) = 2k_1 \times (7+5m)T_\alpha \le T_{\text{tree}}.$$

For the worst case, the hexagonal tree is a k_2-level skew tree as shown in Fig. 18(c), where $k_2 = \frac{N_{\text{tree}}+N_{\text{path}}}{N_{\text{path}}+13}$ and $N_{\text{tree}} = 13 \times k_2 + (k_2 - 1) \times N_{\text{path}}$. The time cost of the hexagonal-tree is

$$\begin{aligned} T_{\text{tree}} &\le k_2 \times T_{\text{branch}} + (k_2-1) \times T_{\text{path}} \\ &= 7T_\alpha k_2 + 5mT_\alpha(k_2-1). \end{aligned}$$

Therefore, the time complexity of constructing a hexagonal-tree is $O(2k_1(7+5m)T_\alpha) \le T_{\text{tree}} \le O(7T_\alpha k_2 + 5mT_\alpha(k_2-1))$. □

4 Experimental results

To examine the effectiveness of our approach, two well-known multicast routing protocols, AODV [6] and ODMRP [2], are mainly compared with our approach. Observe that AODV and ODMRP do not offer the QoS capability. In addition, Tseng designed a TDMA-based QoS uni-path routing protocol in [14]. To make a fair comparison, we offer the QoS-extension AODV [6] and ODMRP [2] such that each path in AODV [6] and ODMRP [2] protocols adopts Tseng's TDMA-based QoS uni-path routing [14], where MAC sublayer is adopted the TDMA channel model. AODV [6] and ODMRP [2] are well known multicast routing protocols. Therefore, AODV(+Tseng) and ODMRP(+Tseng) are QoS multicast routing protocol, where each time slot reservation of link in AODV [6] and ODMRP [2] adopts Tseng's TDMA-based QoS uni-path routing result [14]. All these protocols are mainly implemented using the NCTUns 2.0 simulator and emulator [3]. In this simulation, these two integrated results are denoted as AODV(+Tseng) and ODMRP(+Tseng), respectively. The scenario simulates a 1000 × 1000 m^2 area. In our simulation, every result is the average of 1000 runs for the same setup environment. To express the confidence level of our simulator, the simulation parameters in the simulator are given as follows:

- The number of mobile hosts ranges from 20 to 50.
- The number of time slots of the data frame is assumed to be 16 slots.
- The bandwidth requirements are two, four, six and eight time slots.
- The mobility ranges from 10 to 40 km/h.
- The message length ranges from 1 to 4 Mbits.
- The radio transmission range is 200 m.

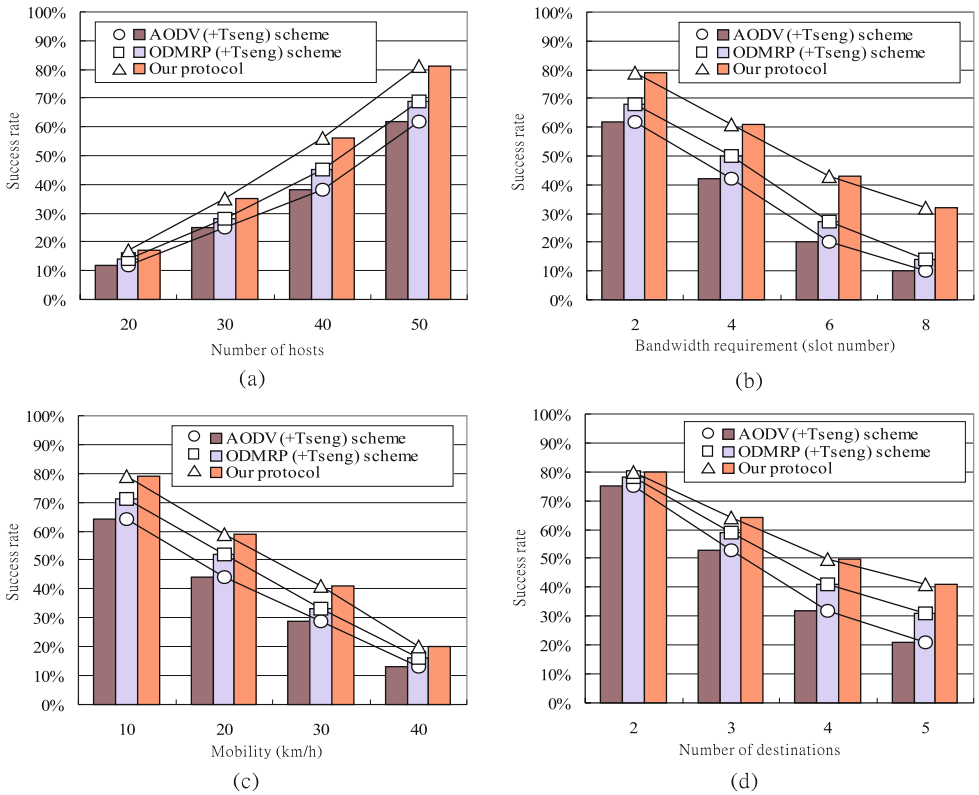


Fig. 15 Performance of success rate vs. effects of (**a**) the number of hosts, (**b**) bandwidth requirement, (**c**) mobility, and (**d**) number of destinations

- The transmission rate is 5 Mbit/s.
- The duration time of each time slot of a time frame is assumed to be 5 ms, and the duration time of a control slot is 0.1 ms.
- The source and destination are selected randomly.
- A packet is dropped if the packet stays in a node in excess of the maximal queuing delay time, which is set to four frame lengths (328 ms).
- Once a QoS request is successful, time slots are reserved for all subsequent packets. The reservation is released when either the data transmission process is finished or the link is broken.

The performance metrics to be observed are:

- *Success rate* (SR). The value of successful QoS route requests divided by the total number of QoS route requests from source to destination.
- *Overhead* (OH). The number of transmitted packets, including the control and data packets.
- *Latency* (LT). The interval from the time the multicast is initiated to the time the last host receives its multicasting.

It is worth mentioning that an efficient QoS routing protocol is achieved by with a high success rate, high throughput, and low latency. In the following, we illustrate the performance of success rate (SR), overhead (OH), and latency (LT) from several perspectives.

4.1 Success rate (SR)

The performance metric SR means that the ratio of successful QoS route requests to the total number of QoS route requests. Figure 15 shows the performance results of success rate (SR) vs. (a) the number of mobile hosts, (b) network bandwidth, (c) mobility, and (d) number of destinations. We have observed that our approach has the better success rate under the situations of high number of hosts, the low bandwidth requirement, the low mobility, and the less number of destination nodes. The performance of SR is discussed as follows.

1(a) *Effects of the number of hosts*. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations) = ((20, 30, 40, 50), 2, 10 km/h, 2). First, to compare the change rate of curve lines along different parameters with the other schemes, AODV(+Tseng) and ODMRP(+Tseng), the change rate of our protocol is increasing speedily than that of the other schemes, as shown in Fig. 15(a). This means that our protocol achieves more success rate by using less cost than that of the other schemes. This result also indicates that the greater the number of hosts is, the higher the success rate is obtained. Second, the bar charts of Fig. 15(a) show that our protocol achieves a better success rate than that of the other schemes under various numbers of hosts.

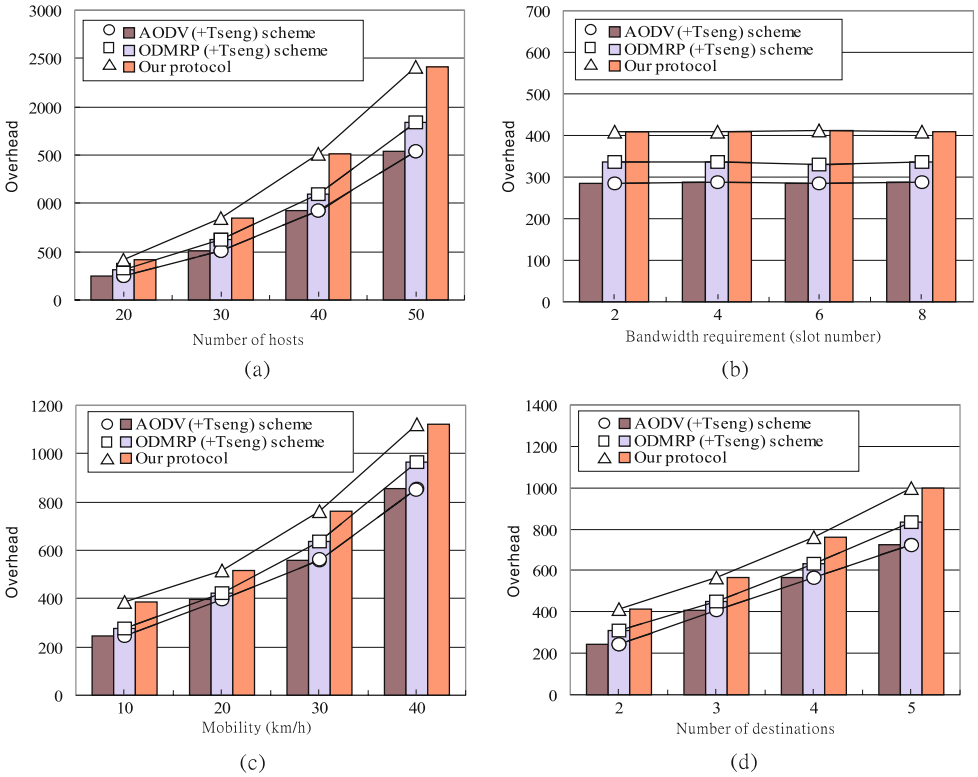


Fig. 16 Performance of overhead vs. effect of (**a**) the number of hosts, (**b**) bandwidth requirement, (**c**) mobility, and (**d**) number of destinations

1(b) *Effects of bandwidth requirement*. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations) = (20, (2, 4, 6, 8), 10 km/h, 2). To compare the change rate of curve lines along different parameters with the other schemes, our protocol is decreasing slowly than that of the other schemes, as shown in Fig. 15(b). This means that our method achieves more success rate by even using more bandwidth than that of the other schemes. This result also indicates that the higher the bandwidth requirement is, the lower the success rate is obtained. The bar charts of Fig. 15(b) show that our protocol achieves better success rates than that of the other schemes under various bandwidth requirements.

1(c) *Effects of mobility*. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations) = (20, 2, (10, 20, 30, 40) km/h, 2). The curve lines of Fig. 15(c) show that the higher the mobility is, the lower the success rate will be. The bar charts of Fig. 15(c) show that our protocol achieves better success rates than that of the other schemes under various mobilities.

1(d) *Effects of the number of destinations*. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations) = (20, 2, 10 km/h, (2, 3, 4, 5)). To compare the change rate of curve lines along different parameters with the other schemes, our method is decreasing slowly than that of the other schemes, as shown in Fig. 15(d). This means that our protocol achieves more success rate by even existing more number of destinations than that of the other schemes. This result also indicates that the greater the number of destinations is, the lower the success rate is obtained. The bar charts of Fig. 15(d) show that our protocol achieves better success rates than that of the other schemes under various numbers of destinations.

The above results indicate that our protocol achieves a better success rate than those of the other schemes under various situations.

4.2 Overhead (OH)

The performance metric OH contains the number of transmitted control packets and data packets. Figure 16 illustrates the performance results of overhead vs. (a) the number of mobile hosts, (b) network bandwidth, (c) mobility, and (d) number of destinations. We have observed that our approach has the higher OH under the different situations of the number of hosts, the bandwidth requirement, the mobility, and the number of destination nodes. Our protocol wins the higher success rate and requires more control packets

than that of the other schemes in each transmission. The reason is that extra control packets need to construct the multi-path in our approach. The performance of OH is discussed as follows.

2(a) *Effects of the number of hosts*. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations) = ((20, 30, 40, 50), 2, 20 km/h, 2). The curve lines of Fig. 16(a) show that the greater the number of hosts is, the higher the OH will be. The reason is the more the number of hosts used, the more the transmitted paths and control packets needed. The bar chars of Fig. 16(a) show that our protocol achieves a higher OH than that of the other schemes under various numbers of hosts.

2(b) *Effects of the bandwidth requirement*. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations) = (20, (2, 4, 6, 8), 20 km/h, 2). The curve lines of Fig. 16(b) show that the OH is almost constant independently of the bandwidth requirement. The bar chars of Fig. 16(b) show that our protocol achieves a higher OH than that of the other schemes under various bandwidth requirements.

2(c) *Effects of mobility*. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations) = (20, 2, (10, 20, 30, 40) km/h, 2). The curve lines of Fig. 16(c) show that the higher the mobility is, the higher the number of connected hosts will be. The more the number of hosts used, the more the transmitted paths and control packets needed. The bar chars of Fig. 16(c) show that our protocol achieves a higher OH than that of the other schemes under various mobilities.

2(d) *Effects of number of destinations*. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations) = (20, 2, 20 km/h, (2, 3, 4, 5)). The curve lines of Fig. 16(d) show that the greater the number of destinations is, the higher the OH will be. The reason is the more the destinations added, the more the control packets needed. The bar chars of Fig. 16(d) show that our protocol achieves a higher OH than that of the other schemes under various numbers of destinations.

The above results indicate that our protocol achieves a higher OH than those of the other schemes under various situations. In particular, our protocol requires a little overhead value than that of the other schemes in this simulation. This is because that extra control packet of our approach is needed to construct the multi-path. However, our approach uses additional control packets to achieve the purpose of higher success rate, higher throughput, and lower latency.

4.3 Latency (LT)

The performance metric LT means that the interval from the time the multicast is initiated to the time the last host receives its multicasting. Figure 17 illustrates the performance results of latency (LT) vs. (a) the number of mobile hosts, (b) network bandwidth, (c) mobility, (d) number of destinations, and (e) message length. We have observed that our approach has the low latency under the different situations of the less number of hosts, the high bandwidth requirement, the low mobility, the greater number of destination nodes, and the greater number of message length. The performance of latency is discussed as follows.

3(a) *Effects of the number of hosts*. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations) = ((20, 30, 40, 50), 2, 20 km/h, 2). The curve lines of Fig. 17(a) show that the greater the number of hosts is, the higher the latency will be. The bar chars of Fig. 17(a) show that our protocol achieves a better (lower) latency than that of the other schemes under various numbers of hosts.

3(b) *Effects of the bandwidth requirement*. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations) = (20, (2, 4, 6, 8), 20 km/h, 2). The curve lines of Fig. 17(b) show that the higher the bandwidth requirement is, the lower the latency will be. The bar chars of Fig. 17(b) show that our protocol achieves a better (lower) latency than that of the other schemes under various bandwidth requirements.

3(c) *Effects of mobility*. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations) = (20, 2, (10, 20, 30, 40) km/h, 2). The curve lines of Fig. 17(c) show that the higher the mobility is, the higher the latency will be. The bar chars of Fig. 17(c) show that our protocol achieves a better (lower) latency than that of the other schemes under various mobilities.

3(d) *Effects of the number of destinations*. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations) = (20, 2, 20 km/h, (2, 3, 4, 5)). The curve lines of Fig. 17(d) show that the greater the number of destinations is, the higher the latency will be. The bar chars of Fig. 17(d) show that our protocol achieves a better (lower) latency than that of the other schemes under various numbers of destinations.

3(e) *Effects of message length*. To observe the influences of message length, we add a situation simulation to this metric. In this simulation, the tuple of parameters (number of mobile hosts, network bandwidth, mobility, number of destinations, message length) =

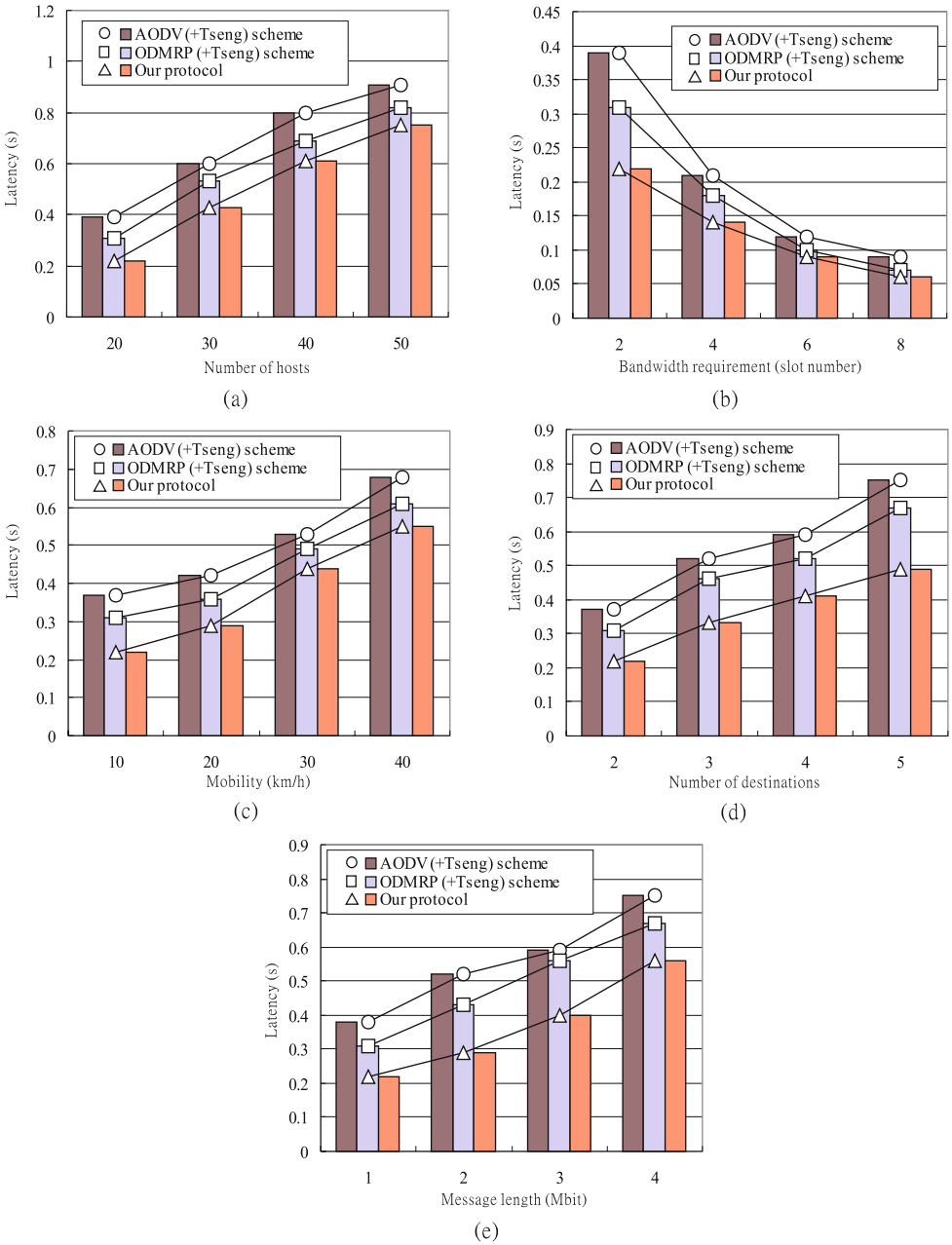


Fig. 17 Performance of latency vs. the effect of (**a**) the number of hosts, (**b**) bandwidth requirement, (**c**) mobility, (**d**) number of destinations, and (**e**) message length

(20, 2, 20 km/h, 2, (1, 2, 3, 4) Mbits). The curve lines of Fig. 17(e) show that the greater the value of message length is, the higher the latency will be. The bar chars of Fig. 17(e) show that our protocol achieves a better (lower) latency than that of the other schemes under various value of message length.

The above results indicate that our protocol achieves a better (lower) latency than those of the other schemes under various situations. As a summary, all above metrics indicate that our protocol achieves higher success rate, higher throughput, and lower latency than those of the other schemes.

5 Conclusions

This paper presents a new TDMA-based QoS multicast routing protocol for a wireless mobile ad hoc network. This study builds a new multicast tree structure, namely a hexagonal-tree, to serve as the QoS multicasting tree. To the power-saving issue, the MAC sub-layer in this paper adopts the simple TDMA channel model. Both of the hidden-terminal and exposed-terminal problems are taken into consideration in order to possibly exploit the time-slot reuse capability during construction of the hexagonal-tree. In this paper, the proposed hexagonal-based scheme indeed

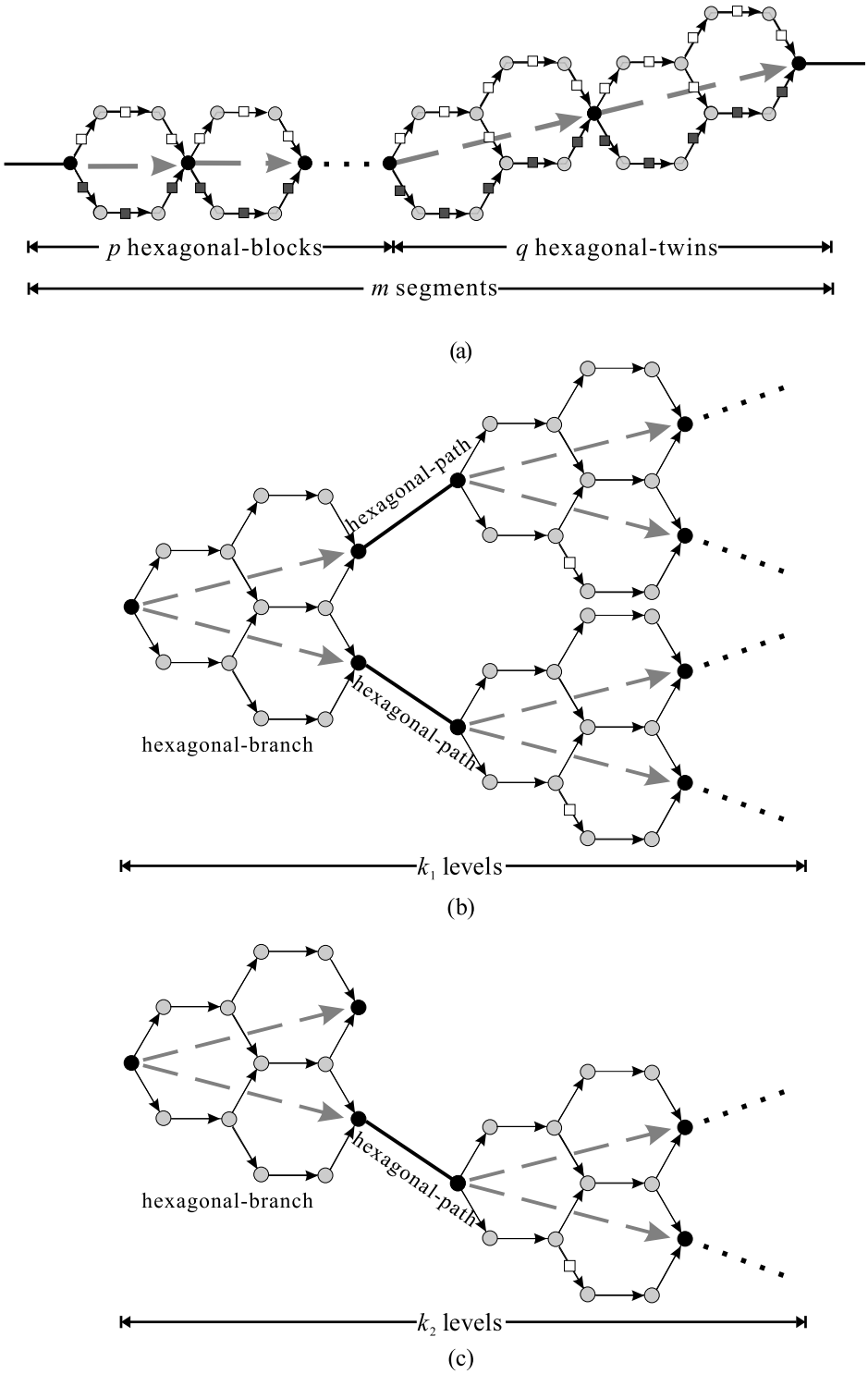


Fig. 18 Example of calculating the time complexity of (**a**) a hexagonal-path, (**b**) a complete hexagonal-tree, and (**c**) a skew hexagonal-tree

offers higher success rates for constructing the QoS multicast tree. This improves the success rate by means of two-path routing. Performance analysis results demonstrate the achievement of efficient QoS multicasting. In addition, the efficiency of wireless sensor networks may gain from this proposed concept, i.e., wireless sensor network can be an additional use case.

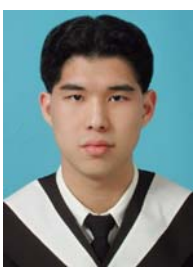
References

1. IEEE Std 802.11-1997. (1997). Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. Institute of Electrical and Electronics Engineer.
2. Lee, S.-J., Su, W., & Gerla, M. (2000). On-Demand Multicast Routing Protocol (ODMRP) for ad hoc networks. IETF manet (draft-ietf-manet-odmrp-02.txt).

3. Wang, S.-Y. NCTUns 2.0 network simulator and emulator. Network and System Laboratory, Department of Computer Science, National Chiao Tung University (NCTU), Taiwan. http://nsl10.csie.nctu.edu.tw/.
4. Johnson, D. B., & Maltz, D. A. (1996). Dynamic source routing in ad hoc wireless networks. In T. Imielinski & H. Korth (Eds.), *Mobile Computing* (Chapter 5, pp. 81–153).
5. Chen, S., & Nahrstedt, K. (1999). Distributed quality-of-service routing in ad hoc networks. *IEEE Journal on Selected Areas in Communications*, *17*(8), 1488–1505.
6. Royer, E.-M., & Perkins, C.-E. (1999). Ad hoc on-demand distance vector routing. In *Proceedings of the second IEEE workshop on mobile computer system and application* (pp. 1488–1505), August 1999.
7. Chen, S., Nahrstedt, K., & Shavitt, Y. (2000). A QoS-aware multicast routing protocol. In *Proceedings of IEEE nineteenth annual joint conference of the IEEE computer and communications societies (INFOCOM)* (pp. 1594–1603).
8. Chen, Y.-S., Chen, T.-S., & Huang, C.-J. (2002). SOM: spiral-fat-tree-based on-demand multicast protocol in a wireless ad hoc network. *Computer Communications*, *25*(17), 1684–1695.
9. Chen, Y.-S., & Lai, K.-C. (2001). MESH: multi-eye spiral-hopping protocol in a wireless ad hoc network. *IEICE Transactions on Communications*, *E84-B*(8), 2237–2248.
10. Chen, Y.-S., Tseng, Y.-C., Sheu, J.-P., & Kuo, P.-H. (2004). An on-demand, link-state, multi-path QoS routing in a wireless mobile ad-hoc network. *Computer Communications*, *27*(1), 27–40.
11. Ho, Y.-K., & Liu, R.-S. (2000). On-demand QoS-based routing protocol for ad hoc mobile wireless networks. In *Proceedings of the fifth IEEE symposium on computers and communications (ISCC)* (pp. 560–565).
12. Hu, Y.-C., & Johnson, D. (2000). Caching strategies in on-demand routing protocols for wireless ad hoc networks. In *Proceedings of the sixth annual IEEE/ACM international conference on mobile computing and networking (MOBICOM)* (pp. 231–242).
13. Liao, W.-H., Tseng, Y.-C., Sheu, J.-P., & Wang, S.-L. (2002). A multi-path QoS routing protocol in a wireless mobile ad hoc network. *Telecommunication Systems*, *19*(3–4), 329–347.
14. Liao, W.-H., Tseng, Y.-C., & Shih, K.-P. (2002). A TDMA-based bandwidth reservation protocol for QoS routing in a wireless mobile ad hoc network. In *Proceedings of IEEE international conference on communications (ICC)* (pp. 3186–3190).
15. Lin, C.-R. (2001). On-demand QoS routing in multihop mobile networks. In *Proceedings of IEEE nineteenth annual joint conference of the IEEE computer and communications societies (INFOCOM)* (pp. 1735–1744), April 2001.
16. Lin, C.-R., & Liu, J.-S. (1999). QoS routing in ad hoc wireless networks. *IEEE Journal on Selected Areas in Communications*, *17*(8), 1426–1438.
17. Lin, H.-C., & Fung, P.-C. (2000). Finding available bandwidth in multihop mobile wireless networks. In *Proceedings of the 51st IEEE vehicular technology conference (VTC)* (pp. 15–18), May 2000.
18. Talukdar, A.-K., & Badrinath, B. R. (1998). Rate adaptation schemes in networks with mobile hosts. In *Proceedings of the fourth annual ACM/IEEE international conference on mobile computing and networking (MOBICOM)* (pp. 169–180).
19. Tseng, Y.-C., Sheu, J.-P., & Lin, C.-Y. (2000). A new multi-channel MAC protocol with on-demand channel assignment for mobile ad hoc networks. In *Proceedings of international symposium on parallel architectures, algorithms and networks (I-SPAN)* (pp. 232–237).
20. Murthy, S., & Garcia-Luna-Aceves, J. J. (1996). An efficient routing protocol for wireless networks. *ACM Mobile Networks and Applications Journal*, *1*(2), 183–197.
21. Deering, S. E., & Cheriton, D. R. (1990). Multicast routing in datagram internetworks and extended LANs. *ACM Transactions on Computer Systems*, *8*(2), 85–110.
22. Toh, C.-K. (1996). Associativity-based routing for ad hoc mobile networks. *Wireless Personal Communications Review*, *2*(4), 1–36.

Yuh-Shyan Chen received the B.S. degree in computer science from Tamkang University, Taiwan, Republic of China, in June 1988 and the M.S. and Ph.D. degrees in Computer Science and Information Engineering from the National Central University, Taiwan, Republic of China, in June 1991 and January 1996, respectively. He joined the faculty of Department of Computer Science and Information Engineering at Chung-Hua University, Taiwan, Republic of China, as an associate professor in February 1996. He joined the Department of Computer Science and Information Engineering, National Chung Cheng University in August 2002. Since 2006, he has been a Professor at the Department of Computer Science and Information Engineering, *National Taipei University*, Taiwan. Dr. Chen served as Co-Editors-in-Chief of *International Journal of Ad Hoc and Ubiquitous Computing* (IJAHUC), Editorial Board Member of Telecommunication System Journal. He also served as Guest Editor of Telecommunication Systems, special issue on "Wireless Sensor Networks" (2004), and Guest Editor of Journal of Internet Technology, special issue on "Wireless Internet Applications and Systems" (2002) and special issue on "Wireless Ad Hoc Network and Sensor Networks" (2004). His paper wins the 2001 IEEE 15th ICOIN-15 Best Paper Award. Dr. Chen was a recipient of the 2005 Young Scholar Research Award given by National Chung Cheng University to four young faculty members, 2005. His recent research topics include mobile ad-hoc network, wireless sensor network, mobile learning system, and 4G system. Dr. Chen is a member of the IEEE Computer Society, IEICE Society, and Phi Tau Phi Society.

Tsung-Hung Lin is an Assistant Professor of Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung County, Taiwan, Republic of China Taiwan, Republic of China. He received the B.S. degree in computer science from Tamkang University, Taiwan, Republic of China, and the M.S. and Ph.D. degrees in Computer Science and Information Engineering from the National Central University, Taiwan, Republic of China. His research interests include wireless communication and mobile computing, the next-generation mobile network and system, wireless sensor network, 4G system, and optical computing.



Yun-Wei Lin received the B.S. degree in computer and information science from the Aletheia University, Taiwan, Republic of China, in June 2003 and the M.S. degree in computer science and information engineering from National Chung Cheng University, Taiwan, Republic of China, in July 2005. His research interests include mobile ad hoc network and wireless sensor network.