Hindawi
Mobile Information Systems
Volume 2017, Article ID 2504604, 10 pages
https://doi.org/10.1155/2017/2504604

Research Article

Weighted-DESYNC and Its Application to End-to-End Throughput Fairness in Wireless Multihop Network

Ui-Seong Yu,[1] Ji-Young Jung,[1] Eutteum Kong,[1] HyungSeok Choi,[2] and Jung-Ryun Lee[1]

[1]*School of Electrical & Electronic Engineering, Chung-Ang University, Seoul, Republic of Korea*
[2]*Agency for Defense Development, Daejeon, Republic of Korea*

Correspondence should be addressed to Jung-Ryun Lee; jrlee@cau.ac.kr

Received 8 December 2016; Revised 21 March 2017; Accepted 29 March 2017; Published 13 April 2017

Academic Editor: Hideyuki Takahashi

The end-to-end throughput of a routing path in wireless multihop network is restricted by a bottleneck node that has the smallest bandwidth among the nodes on the routing path. In this study, we propose a method for resolving the bottleneck-node problem in multihop networks, which is based on multihop DESYNC (MH-DESYNC) algorithm that is a bioinspired resource allocation method developed for use in multihop environments and enables fair resource allocation among nearby (up to two hops) neighbors. Based on MH-DESYNC, we newly propose weighted-DESYNC (W-DESYNC) as a tool artificially to control the amount of resource allocated to the specific user and thus to achieve throughput fairness over a routing path. Proposed W-DESYNC employs the weight factor of a link to determine the amount of bandwidth allocated to a node. By letting the weight factor be the link quality of a routing path and making it the same across a routing path via Cucker-Smale flocking model, we can obtain throughput fairness over a routing path. The simulation results show that the proposed algorithm achieves throughput fairness over a routing path and can increase total end-to-end throughput in wireless multihop networks.

1. Introduction

Wireless multihop networks are used in various environments such as sensor networks, Internet of Things (IoT), wireless body area networks (WBAN), wireless ad hoc network (WANET), wireless mesh network (WMN), and vehicle ad hoc network (VANET). In such a large number of fields, various devices such as sensors, smart phones, and automobiles that support various types of wireless communication continue to increase in number, so it is essential to utilize resources more efficiently.

Because a wireless node can communicate directly with only the node within the transmission range, in order for a source-destination pair to communicate with each other over a long distance, nodes serving as routers for delivering messages are required. In such wireless multihop networks, bottleneck-node problems can occur when the source and destination nodes communicate via a multihop routing path. The bottleneck node means the node whose allocated bandwidth is the lowest across the routing path connecting the source and the destination. In a multihop environment, bottleneck nodes are the main cause of decrease of end-to-end throughput. This bottleneck node is mainly caused by sharing the limited amount of resources while crossing certain other paths in the multihop communication process [1].

For the purpose of organizational communication between nodes, a routing process that operates in a distributed way is essential for a node to efficiently communicate with other nodes. In general, the performance of a routing process such as an end-to-end throughput between source and destination pair is mainly restricted by a bottleneck node, which serves as a relay node on multiple routing paths or suffers from a low link quality and thus is allocated the smallest bandwidth (lowest data rate) among the nodes in the routing path [2]. As a result, these bottleneck nodes can decrease the end-to-end throughput of the routing path and increase the queuing delay between source and destination nodes. One simple method to solve the bottleneck-node problem is to replace the routing path with one of higher

performance, but this method has a limitation: there might not be a sufficient number of nodes in the network, meaning that multiuser diversity cannot be exploited in the given network environment. For this reason, researchers have tried to solve the bottleneck-node problem via scheduling, mostly by giving a high priority to bottleneck nodes for data processing [3].

However, under the contention-based medium access protocol, typical scheduling methods to resolve the bottleneck-node problem require high complexity and additional procedures for prioritization for scheduling. Furthermore, in wireless multihop networks, scheduling-based solutions for the bottleneck-node problem show poor scalability because of rapid changes in the network topology. Thus, scheduling-based solutions for the bottleneck-node problem are not suitable considering the high computational complexity, network overhead, and low scalability.

2. Related Work

The unfair and insufficient resource allocation of a bottleneck node causes the throughput decrease and delay increase in the end-to-end packet delivery of a routing path. To resolve this problem, the fair use of throughput in multihop networks is emphasized [3]. Fair use of throughput in multihop networks can be implemented in three ways: (1) the selection of the routing path to avoid the bottleneck node and thus guarantee throughput fairness in the routing process, (2) the fair allocation of bandwidths among the nodes on the routing path via the multiple-access control (MAC) protocol, and (3) the use of cross-layer optimization for both MAC and routing protocols to overcome the fundamental limits of these protocols.

2.1. Related Works: Routing Protocol. In an effort to solve the bottleneck-node problem via routing procedures, the concept of the parallel path has been proposed. In [4], multiple disjoint routing paths for a source-destination pair were provided through the unit cell division and signaling message exchange and constant per-node transmission rate is maintained. In [5], the authors obtained the minimum hop bandwidth requirements and network optimization and extended the straightforward precomputation scheme to solve the bottleneck for QoS, by using a straightforward precomputation scheme of standard Bellman-Ford algorithm. In [6], an optimal path was selected using distance-vector routing techniques with consideration of QoS, and the estimated available bandwidth of the path was set to the maximum bandwidth. In an environment where the network nodes have mobility and thus the network topology dynamically changes, these efforts to solve bottleneck-node problem via routing protocols have limits, as the prompt adaptation and preservation of fair resource allocation are challenging because of signaling overheads and the redundant delay time required to reestablish the routing path via techniques such as route reply (RREP) or route request (RREQ) messages.

2.2. Related Works: MAC Protocol. In multihop ad hoc networks, end-to-end bandwidth and delay are highly dependent on the network topology [7]. Therefore, the MAC layer which controls the resource allocation of a node can be a method for throughput fairness among nodes on the same routing path. Fair resource allocation among users is provided by controlling the frame length of a user for adjusting the amount of data transferred in a scheduling-based MAC protocol (such as the time-division multiple-access (TDMA) protocol) or the contention window size of a user for the control of the user-based transmission opportunity in a contention-based MAC protocol (such as CSMA/CA) [8]. In [9], a method that ensures fairness using flow control and a virtual queue-scheduling scheme is suggested [9]. In [10], a fixed-length bitmap vector is used in each packet header for exchanging slot timing information between immediate neighbors and those separated by up to two hops. Each node iteratively moves its slot near the middle of the slots of its neighbors until an allocation pattern becomes stable and fair allocation of resources is obtained for all the nodes in the neighborhood.

2.3. Related Works: Cross-Layer Approach. Although each communication layer has its specific functionality such as congestion control at the transport layer, routing at the network layer, and scheduling/power control at the MAC/physical (PHY) layer, it is expected that interactions between the various layers are necessary for achieving the optimal performance [11]. In [12], distributed multiuser scheduling was proposed, which is a carrier sense multiple access with collision avoidance-based technique considering both the PHY and MAC layers. This method calculates the signal-to-noise ratio (SNR) of each node and determines the contention window size according to the average SNR. In the distributed queuing collision avoidance scheme proposed in [13], a virtual priority function is used for queuing scheduling to ensure fairness and collision avoidance. The dual-based method is implemented as follows. At the transport layer, end-to-end sessions adjust their rates in a distributed manner to attain proportionally fair session rates given specific link rates. At the link layer, the link-attempt probabilities are adjusted with local information, so that the bandwidth bottlenecks are alleviated and the aggregate utilities can be further increased [14].

In this paper, we propose a resource allocation scheme called W-DESYNC based on multihop DESYNC in wireless network, as a tool artificially to control the amount of resource allocated to the specific user and thus to achieve throughput fairness over a routing path. In W-DESYNC, the amount of resource allocated to each node is decided not by the number of neighbor nodes but by the weight factor of the node. In order to make the weight factors of nodes on the same routing path the same, we apply Cucker-Smale flocking model to the weight factors of node where the weight factor of a node is set to SNR and therefore achieves the throughput fairness over a routing path. In this way, the proposed W-DESYNC method enables the nodes on the routing path to have the same throughput and the bottleneck problem can be solved to improve the network performance.

The configuration of the paper is as follows. Section 3 discusses the previous MH-DESYNC. In Section 4, we

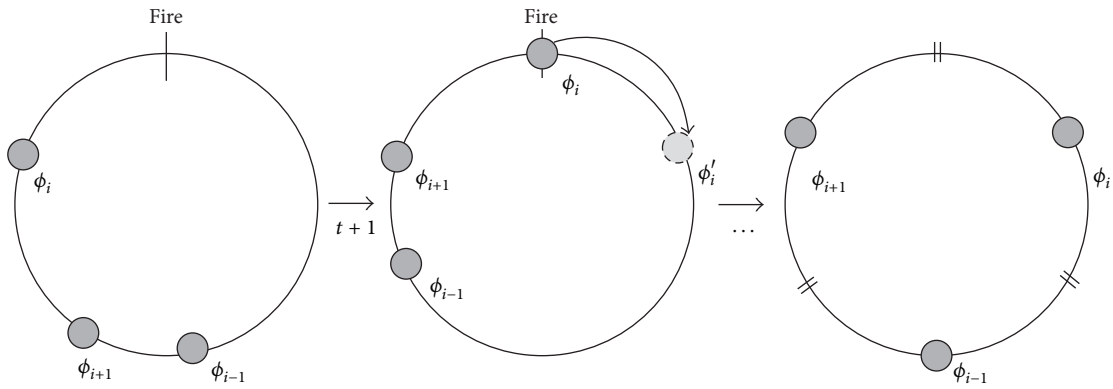


Figure 1: Phase update in DESYNC.

explain the proposed W-DESYNC method, which will act as a tool to assign priorities on the links of a routing path, and the detailed algorithm to achieve throughput fairness over a routing path by applying Cucker-Smale flocking model for the purpose of making the weight factors of the links the same across a routing path. Section 5 evaluates the performances of the proposed scheme and Section 6 concludes.

3. Previous DESYNC and MH-DESYNC Methods

In this section, we review bioinspired algorithms that have been proposed, with the purpose of providing a solution for distributed resource allocation in multihop networks. Bioinspired methods are derived by modeling the operational principles of living organisms in nature through observation and expressing these principles in mathematical formulae. Bioinspired algorithms have autonomous properties, such as self-organization, self-learning, and self-management, rather than being externally controlled by a centralized coordinator. Such characteristics are highly suitable for application to distributed processing and device-to-device communication, which are strongly required functionalities in wireless multihop networks.

3.1. DESYNC. We review the DESYNC method, which is based on the inverse concept of firefly synchronization [15–17]. Firefly synchronization imitates a phenomenon where fireflies with different glittering phases glitter simultaneously through interaction among them. Conversely, the DESYNC method shows desynchronization between glittering phases of fireflies by trying to make the glittering phase of a node as far as possible from the glittering phases of other fireflies. As a result, the glittering phases of all fireflies become evenly spaced in a distributed way.

Suppose that there are N nodes in a fully connected network. Each node performs a firing periodically with a period T. The phase of node i at time t is denoted as $\phi_i(t) \in [0, 1]$, where the phases 0 and 1 are identical and $0 \le i \le N-1$. When node i reaches the node of its cycle ($\phi_i(t) = 1$), it fires to indicate the termination of its cycle to the other nodes. After firing, the node resets its phase to $\phi_i(t^+) = 0$. Node i records the times of the following two firing events: the one that precedes its own firing (previous firing $\phi_{i+1}(t)$) and the one that occurs just afterwards (next firing $\phi_{i-1}(t)$). Node i calculates the midpoint of its two reference phases as $\phi_{\text{mid}} = (1/2)[\phi_{i+1}(t) + \phi_{i-1}(t)]$ and jumps towards it as follows:

$$\phi_i'(t) = (1-\alpha)\,\phi_i(t) + \alpha\phi_{\text{mid}}(t), \tag{1}$$

where $\alpha \in [0, 1]$ is a parameter that scales how far node i moves from its current phase towards the desired midpoint. Figure 1 shows the phase update procedure in DESYNC.

3.2. Application of DESYNC to Resource Allocation in TDMA. The operational principle of the DESYNC can be applied to TDMA-based networks to obtain distributed and fair resource allocation among all the nodes in a network with the assumption that all nodes are fully connected. In the so-called DESYNC-TDMA, the amount and the range of TDMA time slots allocated for the data transfer of the node i at time $t+1$ are determined by two reference phases of node i at time t, $\phi_{i-1}(t)$ and $\phi_{i+1}(t)$. The left-mid phase and right-mid phase of node i at time t are defined as follows.

$$\begin{aligned}\phi_{i,\text{left-mid}}(t) &= \frac{1}{2}\left[\phi_i(t) + \phi_{i-1}(t)\right],\\ \phi_{i,\text{right-mid}}(t) &= \frac{1}{2}\left[\phi_i(t) + \phi_{i+1}(t)\right].\end{aligned} \tag{2}$$

Based on the above left- and right-mid phases, node i occupies the TDMA time slots beginning at $\phi_{i,\text{left-mid}}(t)$ and ending at $\phi_{i,\text{right-mid}}(t)$. In this way, all of the nodes occupy the nonoverlapping time slots that cover a period T evenly. Figure 2 shows the updating process of firing phase and the TDMA slot allocation procedure using left- and right-mid phases of node i.

3.3. Multihop DESYNC (MH-DESYNC). Because the DESYNC-TDMA method assumes a fully connected network topology, it causes a hidden-node problem and thus cannot be directly applied to multihop networks. Therefore, the MH-DESYNC method was proposed, with the purpose of facilitating collision-free and fair resource allocation among not only one-hop neighbors but also two-hop neighbors in wireless multihop networks [18].

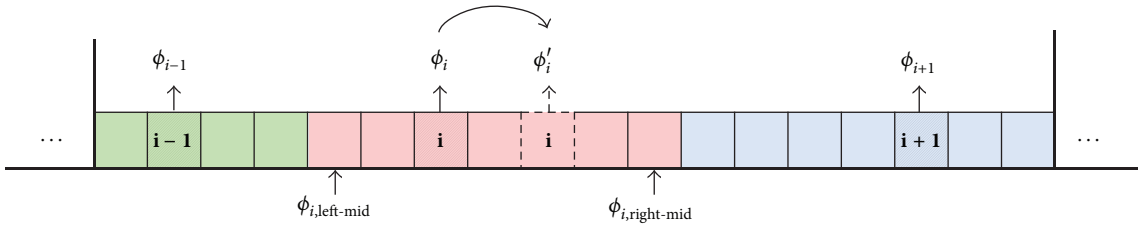


Figure 2: TDMA time-slot allocation of node i in DESYNC-TDMA.

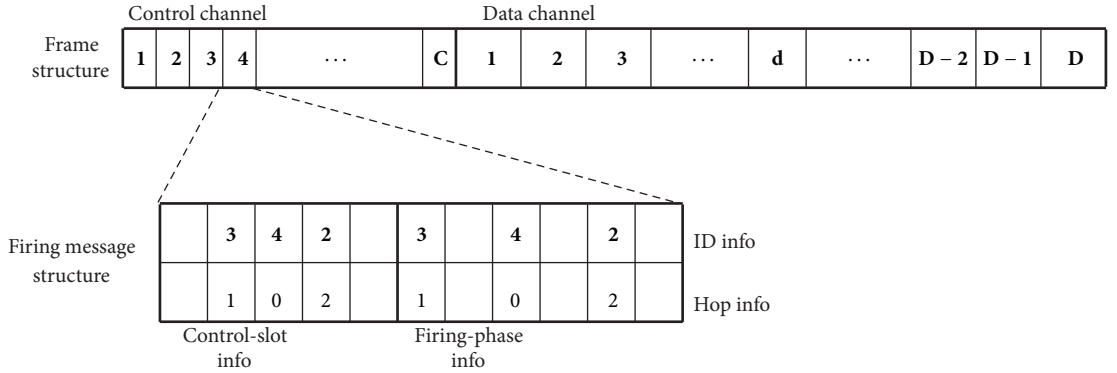


Figure 3: Physical and logical firing structure.

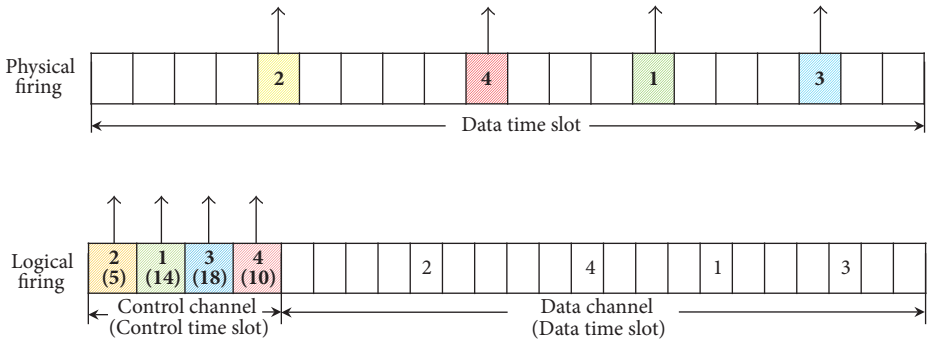


Figure 4: Frame structure and firing message in MH-DESYNC.

Compared with the DESYNC-TDMA method, the main difference of the MH-DESYNC method is virtual firing. In the DESYNC-TDMA method, the firing of a node is identified by the broadcasting of the firing phase of the node to its neighboring nodes via a firing signal. The firing of a node is a series of short interrupt messages that are transmitted in a given data time slot, and the firing signal is referred to as a physical firing signal, as shown in Figure 3. In contrast, with virtual firing, each node records both its own firing phase and its one-hop neighboring nodes firing phases in its own firing message, which is generated in every frame and is sent to its one-hop neighboring nodes via the control time slot of the frame. It is comparatively noted that the physical firing signal in the DESYNC-TDMA method notifies the phase of a node implicitly, by the location of the data time slot where a series of interrupt messages are sent. The virtual firing of the MH-DESYNC method is facilitated by a new frame structure and the firing-message structure, which are shown in Figure 4. In addition, the detailed operational procedures of the MH-DESYNC method for control time-slot allocation, firing-phase allocation and updating, and data time-slot allocation are provided. Regarding control time-slot allocation, the initial control time-slot allocation procedure for a newly entering node and the detailed method for control time-slot collision detection and resolution are provided. For firing-phase allocation and updating, the initial firing-phase allocation and updating procedures for nodes that succeed in cooccupying a control time slot and the firing-phase updating procedure are explained. Regarding data time-slot allocation, the methods for detecting and resolving firing-phase collision and allocating an actual data time slot to each node are provided.

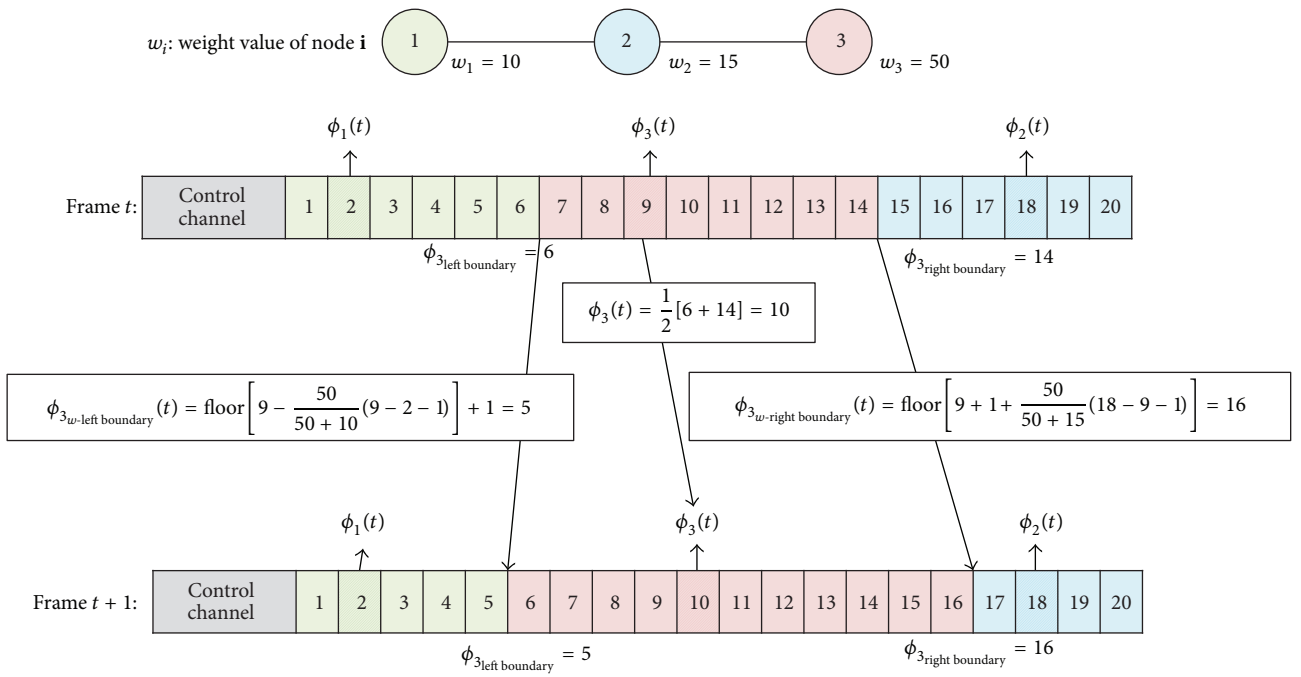


Figure 5: Firing-phase update data and time-slot allocation through firing in the W-DESYNC method.

4. Proposed Algorithm for Throughput Fairness over a Routing Path

4.1. Proposed W-DESYNC Method. The existing MH-DESYNC method can carry out fair resource allocation under wireless multihop network in a distributed way [18]. Specifically, in MH-DESYNC, the amount of resource allocated to node i is $1/(N1(i) + N2(i) + 1)$, where $N1(i)$ and $N2(i)$ are the numbers of one-hop and two-hop neighbor nodes of node i, respectively. Here, it is noted that the amount of resource allocated to a node under MH-DESYNC is determined passively; that is, a node cannot determine the amount of resource allocated to itself but it is just determined by the network environment. As we mentioned in Section 1, this unfair resource allocation naturally involves the so-called bottleneck node and causes a max–min problem on the end-to-end throughput of a routing path.

In this section, the proposed W-DESYNC method is described as a tool to provide throughput fairness over a routing path. While the left boundary (right boundary) of node i is determined by the *midpoint* of the firing phases of nodes $i - 1$ ($i + 1$) and i in MH-DESYNC, the proposed W-DESYNC method dynamically determines both the forward mid and backward mid of a node, considering the amount of bandwidth necessary for each node. The relative priority for the amount of bandwidth of a node is expressed by a weight factor. Let the weight factor of node i be w_i. Then, the forward mid and backward mid for each node i are calculated as follows.

$$\arg\min_{\forall i \in N,\,(i \neq k)} \left(\phi_{\text{left boundary}} = \text{ceil}\left[\phi_i(t) - \frac{w_i}{w_{i-1} + w_i} \left(\phi_i(t) - \phi_{i-1}(t) - 1 \right) \right] + 1 \right),$$

$$\arg\min_{\forall i \in N,\,(i \neq k)} \left(\phi_{\text{right boundary}} = \text{ceil}\left[\phi_i(t) + 1 + \frac{w_i}{w_{i-1} + w_i} \left(\phi_i(t) - \phi_{i-1}(t) - 1 \right) \right] \right). \tag{3}$$

Here, it is noted that the order of the firing phases can be changed under the W-DESYNC algorithm, whereas it cannot be changed once it is decided under the DESYNC and MH-DESYNC algorithms. Thus, the W-DESYNC method requires each node i to update the indices of its forward and backward nodes, by calculating the left boundary and right boundary through (3) for all neighbors of node i at every frame. The resources allocated to node i at frame t are given by

$$\left[\phi_{i_{\text{left boundary}}(t)}, \phi_{i_{\text{right boundary}}(t)} \right]. \tag{4}$$

After the resource allocation is finished through the aforementioned process, the firing phase is updated, as follows:

$$\phi_i'(t) = \frac{1}{2} \left[\phi_{i_{\text{left boundary}}(t)} + \phi_{i_{\text{right boundary}}(t)} \right]. \tag{5}$$

Figure 5 shows the operational procedure of the W-DESYNC method when there are nodes 1, 2, and 3 with $w_1 = 10$, $w_2 = 15$, and $w_3 = 50$.

4.2. Application of Cucker-Smale Flocking Model to W-DESYNC for Throughput Fairness over a Routing Path. The W-DESYNC method is designed to dynamically control the amount of resources allocated to each node by using a weight factor. Here, we explain the method to facilitate throughput fairness among the nodes on the given routing path by using

Table 1: Comparison of Cucker-Smale flocking model and the proposed algorithm.

	Operational rule	Parameter	Flocking object
Cucker-Smale flocking	$v_i(t+1) - v_i(t) = \frac{\lambda}{N}\sum_{j=1}^{N}\psi(\lvert x_j(t) - x_i(t)\rvert)(v_j(t) - v_i(t))$	Position (x), velocity (v)	Direction, velocity
The proposed algorithm	$w_i(n+1) - w_i(n) = \frac{1}{\lVert N_1(i)\rVert}\sum_{j\in N_1,\ j\neq i}(w_j(n) - w_i(n))$	Link quality (w_i)	Link throughput

the Cucker-Smale flocking model. In this flocking model, each autonomous entity controls its velocity and moving direction through interaction with neighboring entities. The CS flocking model is described as follows. Suppose that there are N autonomous entities. The position and velocity vector of the ith entity in $\mathbb{R}^3$ at time $t \in \mathbb{N}$ are denoted as $x_i(t)$ and $v_i(t)$, respectively. The CS flocking model is given as

$$\frac{dx_i}{dt}(t) = v_i(t), \tag{6}$$

$$v_i(t+1) - v_i(t) = \frac{\lambda}{N}\sum_{j=1}^{N}\psi\left(\left|x_j(t) - x_i(t)\right|\right)\left(v_j(t) - v_i(t)\right), \tag{7}$$

for $i = 0, \ldots, N$, and $t > 0$, where λ and $\psi(\cdot)$ are the coupling strength and the function denoting the communication range that quantifies the way the entities influence each other, respectively. The communication-range function $\psi(\cdot)$ is a nonnegative function denoting the distance between entities [19], and some possible $\psi(\cdot)$s are given by

$$\begin{aligned}\psi_1\left(\left|x_j - x_i\right|\right) &= 1,\\ \psi_2\left(\left|x_j - x_i\right|\right) &= 1_{|x_j - x_i| \le r},\\ \psi_3\left(\left|x_j - x_i\right|\right) &= \frac{1}{\left(1 + \left|x_j - x_i\right|^2\right)^{\beta}},\end{aligned} \tag{8}$$

where r and β have positive values. According to (7), each entity adds a weighted average for the differences between the entity and its neighboring entities each time. By executing (7) repeatedly in a distributed manner, the collective behavior of each entity, that is, flocking, is obtained. Time-asymptotic flocking phenomena are expressed to satisfy the following equations:

$$\begin{aligned}\lim_{t\to\infty}\left|v_i(t) - v_j(t)\right| &= 0 \quad \text{for } i \neq j,\\ \sup_{0\le t<\infty}\left|x_i(t) - x_j(t)\right| &< \infty \quad \text{for } i \neq j.\end{aligned} \tag{9}$$

In [19], Cucker and Smale showed that if the communication has long-range interaction, global unconditional flocking occurs, which means that the velocities of all agents converge to the same asymptotic velocity.

For the purpose of making the weight factors of all nodes on the routing path the same, we employ this Cucker-Smale flocking model. Using this model, we could make the weight factor of the link the same across a routing path and thus obtain the throughput fairness over a routing path. The detailed procedure is described as follows.

Let $\mathrm{BW}_i(n)$, $\mathrm{SNR}_{i,j}(n)$, and $d_{i,j}(n)$ be the bandwidth of node i allocated at frame n, the received SNR at node i from node j at frame n, and the throughput of node i at frame n, respectively. Then we have

$$d_{i,j}(n) = \mathrm{BW}_i(n)\log_2\left(1 + \mathrm{SNR}_{i,j}(n)\right). \tag{10}$$

Here, the SNR information of a node is shared with its one-hop neighbor nodes of the routing path (1) by using control messages such as RREQs and RREPs occurring in the routing path setup process, (2) by piggybacking data transmission, or (3) by overhearing the messages sent by the one-hop neighbor nodes. We use $d_{i,i+1}(n)$ as the weight factor of node i at frame n for throughput fairness; that is, $w_i(n) = d_{i,i+1}(n)$. Thus, the throughput fairness of the nodes in a routing path is solved by synchronizing $w_i(n)$ for all i.

Let $N_1(i)$ and $w_i(n)$ be the set of one-hop neighbor nodes of node i that are located in the routing path passing through node i and the weight factor of node i at frame n, respectively. Then, we apply the C-S flocking model to the weight factor of node i by updating the weight factor of node i as follows:

$$w_i(n+1) - w_i(n) = \frac{1}{\left\lVert N_1(i)\right\rVert}\sum_{j\in N_1,\ j\neq i}\left(w_j(n) - w_i(n)\right), \tag{11}$$

where $\lVert\cdot\rVert$ represents the cardinality of the set. The iterative and distributive execution of (11) results in convergence of w_i.

Table 1 compares original Cucker-Smale flocking model and the proposed algorithm with respect to its operational rule, parameter, and flocking object. It is noted that the main target to synchronize in the Cucker-Smale flocking model is the directions and velocities of flying birds (or autonomous objects) while that is link throughput of a routing path (expressed by the weight factor) in the proposed algorithm.

4.3. Application to Throughput Fairness over Multiple Routing Paths. Application of the CS flocking model to the weight factor of a node on a given routing path yields the throughput fairness of the routing path. However, some nodes may belong to more than one path, for example, node 2 shown in Figure 6. Because (11) is assumed to be applied to a node on a single routing path, the resource allocation procedure for a node belonging to more than two routing paths must be defined. For this purpose, we simply assume that a node

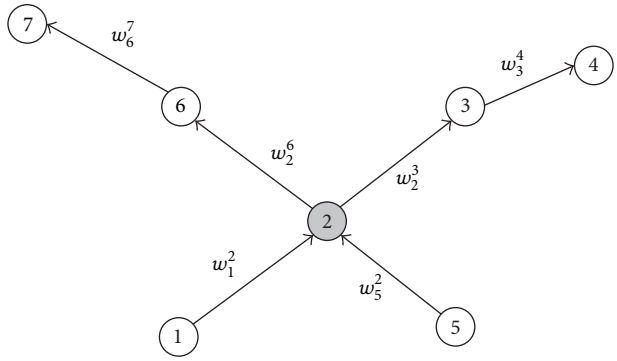
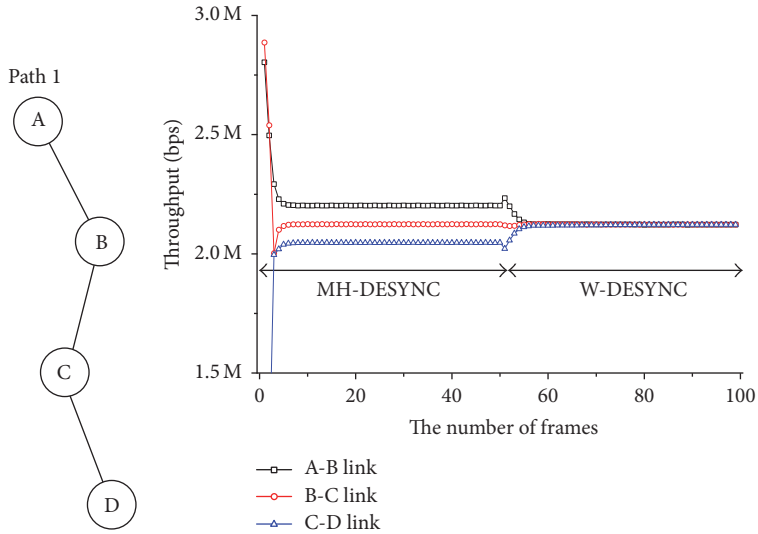


FIGURE 6: Node 2 on the multiple routing paths.



(a) Network topology: a routing path

(b) Variation of allocated throughputs under MH-DESYNC and the proposed algorithm

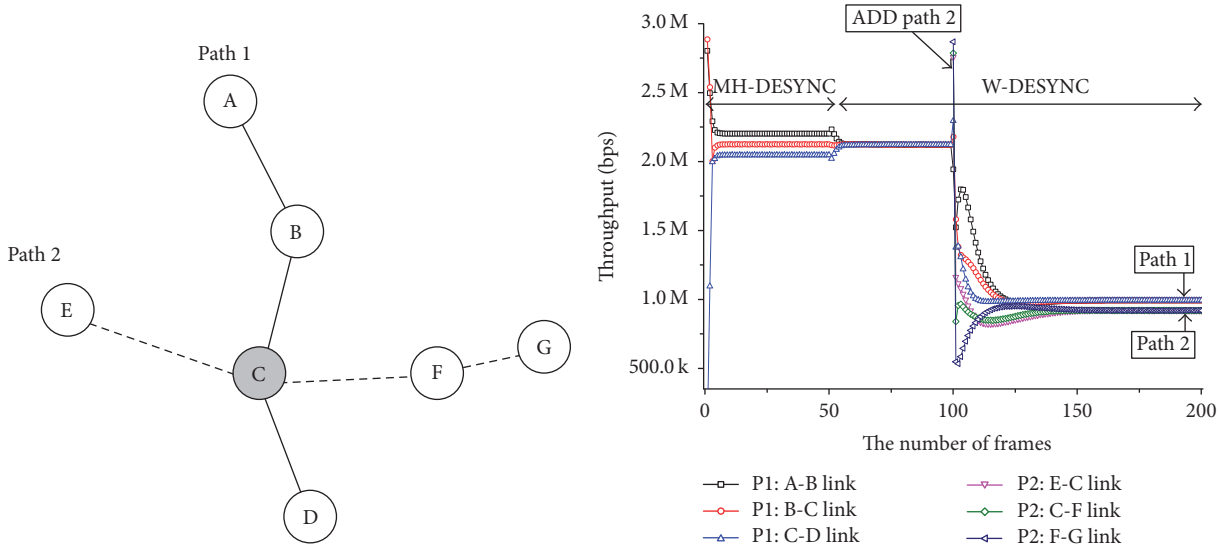
FIGURE 7: Variation of available bandwidth for a routing path.

on each of the k different routing paths executes firing k times in a single frame. Thus, the resources of a node on multiple routing paths are exclusively allocated as many times as the number of routing paths that the node belongs to, in a single frame. We therefore allocate the resources of the node on multiple routing paths to be used for each routing path. Let $w_i^j(n)$ be the weight factor of the link from node i to node j. In Figure 6, node 2 is on two routing paths (path 1-2-3-4 and path 5-2-6-7) and performs firing twice so as to obtain the bandwidth for the routing paths of 1-2-3-4 and 5-2-6-7 by using its two weight factors $w_2^3(n)$ and $w_2^6(n)$, respectively. In Table 1, the concepts of C-S flocking model and W-DESYNC are mapped. Cucker-Smale flocking model synchronizes direction and velocity from position and velocity. W-DESYNC uses Cucker-Smale flocking model to equalize the throughput from the weight factor.

5. Performance Evaluation

In this section, we evaluate the performance of the proposed W-DESYNC method compared with the MH-DESYNC method. Here, it is noticed that the performance of the DESYNC method is not considered, because it was proposed for the use in fully connected networks, so it cannot be applied to the multihop environment unlike to the W-DESYNC and MH-DESYNC methods.

5.1. Application of the Proposed Algorithm to Single Routing Path. Figure 7(a) shows the routing path that consists of nodes A, B, C, and D in a wireless multihop network. The available bandwidth of each node is determined by using MH-DESYNC when the frame number is up to 50 frames and after that, the proposed algorithm using W-DESYNC is applied to the same routing path. Figure 7(b) shows that node



(a) Network topology: two routing paths

(b) Variation of allocated throughputs under MH-DESYNC and the proposed algorithm

Figure 8: Variation of available bandwidth for two routing paths.

C is allocated the lowest bandwidth when MH-DESYNC is used for the path A-B-C-D. The end-to-end throughput of the path A-B-C-D is therefore limited by the bandwidth of the link of bottleneck node C. On the other hand, we can verify that the amount of bandwidths allocated to each link converges to the same when the proposed algorithm is applied, which shows that the bottleneck-node problem is resolved with the use of the proposed algorithm.

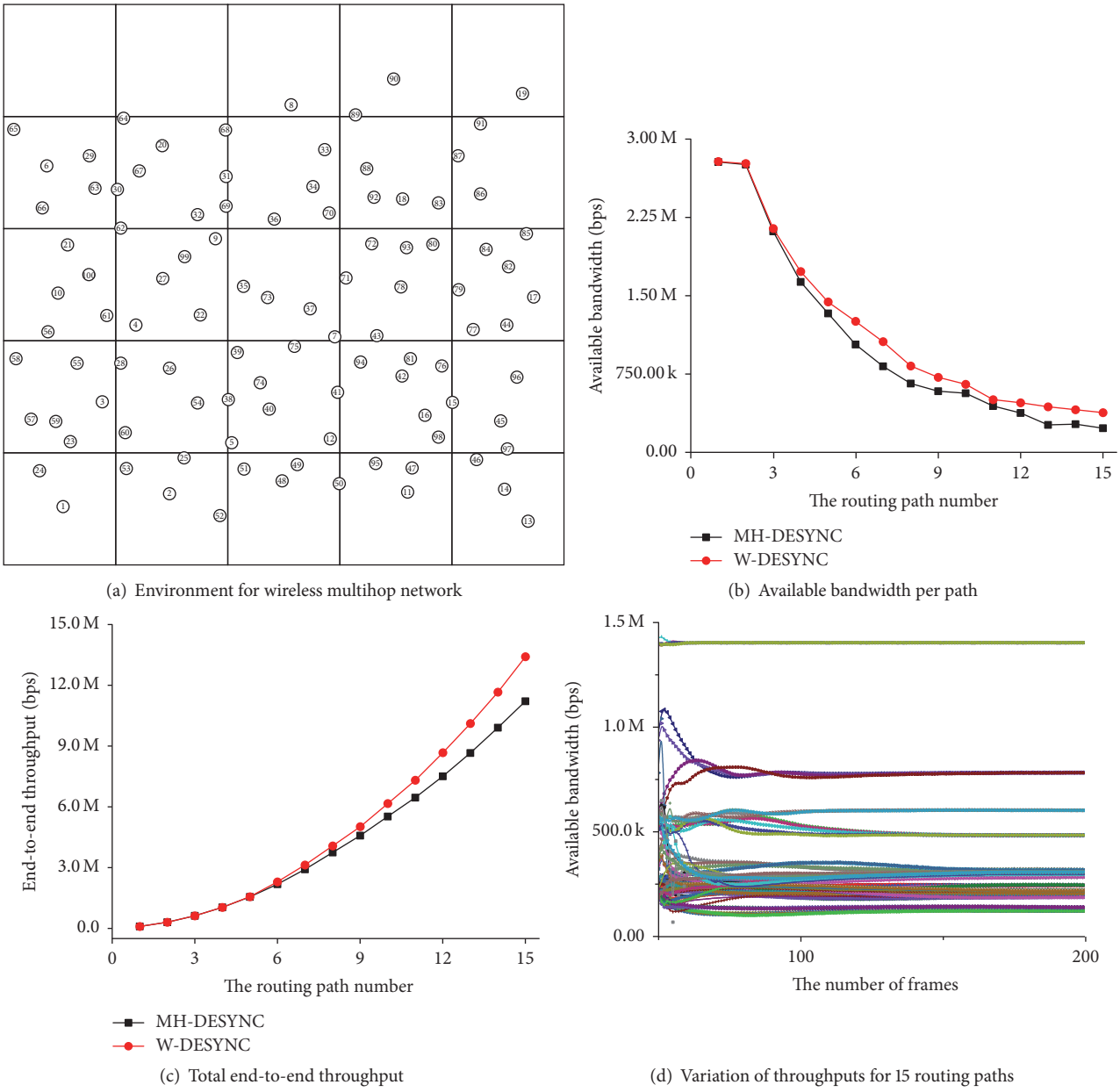
5.2. Application of the Proposed Algorithm to Multiple Routing Paths. Figure 8(a) shows the network topology with two routing paths; specifically the new path E-C-F-G is added to the path A-B-C-D in Figure 7(a). In this figure, node C becomes a bottleneck node where two paths share. Figure 8(b) shows the variation of available bandwidth when the new path E-C-F-G is added at the frame number of 100. The result shows that the link throughputs in each routing path converge to the same value although the converged link throughput of each routing path is different. From the result, we can confirm that the proposed algorithm can prevent nodes from being bottleneck nodes that decrease end-to-end throughput of the routing path even when some nodes are shared by multiple routing paths.

5.3. Network Level Simulation. Finally, we measure and compare the performances of the proposed W-DESYNC method and the MH-DESYNC method in an environment where 100 nodes are randomly deployed and the network size is $500 \times 500\ m^2$, as shown in Figure 9(a). Source and destination nodes are arbitrarily chosen, and the routing paths are set up through ad hoc on-demand distance-vector routing (AODV). Simulation parameters used in the simulation runs are summarized in Table 2.

Table 2: Simulation parameters.

Parameter	Value
The number of nodes	100
Routing protocol	AODV
System bandwidth	11 Mbps
Transmission range	100 m
Network size	500 m by 500 m
The number of control slots	40
Control slot size	0.0747 ms
The number of data slots	10,000
Data slot size	0.8988 μs
Frame size	11.235 ms
Packet size	324 bytes (including MAC header of 20 bytes and IP header of 24 bytes)
Packet generation interval	20 ms

Figure 9(b) shows the average end-to-end bandwidth of a routing path as a function of the number of routing paths while adding a routing path one by one. The results show that as the number of paths increases, more nodes share the limited resource among neighbors, and thus the average end-to-end bandwidths of a routing path decrease under both methods. However, the result shows that the proposed W-DESYNC method achieves higher end-to-end bandwidth than the MH-DESYNC method. Figure 9(c) shows the total end-to-end throughput as a function of the number of routing paths in the network when the proposed and MH-DESYNC methods are used. Here, routing paths are added one by one in the network, similar to the case of Figure 9(b), and packets are generated every 20 ms at the source node and transmitted via the given routing path to the destination



(a) Environment for wireless multihop network

(b) Available bandwidth per path

(c) Total end-to-end throughput

(d) Variation of throughputs for 15 routing paths

FIGURE 9: Wireless multihop performance.

node. According to Figure 9(c), the throughput fairness among all the nodes in the routing path via the W-DESYNC method enhances the total end-to-end throughput in the network. Figure 9(d) shows the variation of the throughput of each routing path when there are 15 routing paths in the network. The end-to-end throughput of the routing paths converges over time, indicating the stability of the proposed method.

6. Conclusion

In this method, we proposed an algorithm for resolving the performance degradation caused by the bottleneck node in a routing path. The W-DESYNC method is proposed as a technique to enable nodes to be allocated resources according to their weight factors. To obtain throughput fairness among the links on a routing path, a bioinspired flocking model is applied to the updating procedure of the weight factor, which is derived from the link-quality information. In addition, the multiple-firing procedures for facilitating multiple resource allocation for a node on multiple routing paths are introduced. Simulation results show that the proposed method achieves the throughput fairness of the routing paths and increases the network throughput by efficiently solving the bottleneck-node problem in multihop networks.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by Agency for Defense Development (ADD-IBR-245). This research was supported by the Chung-Ang University Research Scholarship grants in 2015.

References

[1] S. Abdel Hamid, H. S. Hassanein, and G. Takahara, *Routing for Wireless Multi-Hop Network*, Springer, 2013.

[2] B. Melander, M. Bjorkman, and P. Gunningberg, "New end-to-end probing and analysis method for estimating bandwidth bottlenecks," in *Proceedings of the IEEE Global Telecommunication Conference (GLOBECOM '00)*, vol. 1, pp. 415–420, November 2000.

[3] M.-F. Tsai, N. Chilamkurti, J. H. Park, and C.-K. Shieh, "Multi-path transmission control scheme combining bandwidth aggregation and packet scheduling for real-time streaming in multi-path environment," *IET Communications*, vol. 4, no. 8, pp. 937–945, 2010.

[4] W.-Y. Shin, S.-Y. Chung, and Y. H. Lee, "Parallel opportunistic routing in wireless networks," *IEEE Transactions on Information Theory*, vol. 59, no. 10, pp. 6290–6300, 2013.

[5] A. Orda and A. Sprintson, "Precomputation schemes for QoS routing," *IEEE/ACM Transactions on Networking*, vol. 11, no. 4, pp. 578–591, 2003.

[6] R. Hou, K.-S. Lui, F. Baker, and J. Li, "Hop-by-hop routing in wireless mesh networks with bandwidth guarantees," *IEEE Transactions on Mobile Computing*, vol. 11, no. 2, pp. 264–277, 2012.

[7] X. Jia, D. Li, and D. Du, "QoS topology control in ad hoc wireless networks," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '04)*, vol. 2, pp. 1264–1272, March 2004.

[8] A. V. Babu and L. Jacob, "Fairness analysis of IEEE 802.11 multirate wireless LANs," *IEEE Transactions on Vehicular Technology*, vol. 56, no. 5, pp. 3073–3088, 2007.

[9] K. Ronasi, V. W. S. Wong, and S. Gopalakrishnan, "Distributed scheduling in multihop wireless networks with maxmin fairness provisioning," *IEEE Transactions on Wireless Communications*, vol. 11, no. 5, pp. 1753–1763, 2012.

[10] F. Yu, T. Wu, and S. Biswas, "Toward in-band self-organization in energy-efficient MAC protocols for sensor networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 2, pp. 156–170, 2008.

[11] X. Lin, N. B. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1452–1463, 2006.

[12] S. Tang, "Distributed multiuser scheduling for improving throughput of wireless LAN," *IEEE Transactions on Wireless Communications*, vol. 13, no. 5, pp. 2770–2781, 2014.

[13] J. Alonso-Zarate, C. Verikoukis, E. Kartsakli, A. Cateura, and L. Alonso, "A near-optimum cross-layered distributed queuing protocol for wireless LAN," *IEEE Wireless Communications*, vol. 15, no. 1, pp. 48–55, 2008.

[14] X. Wang and K. Kar, "Cross-layer rate optimization for proportional fairness in multihop wireless networks with random access," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1548–1559, 2006.

[15] J. Degesys, I. Rose, A. Patel, and R. Nagpal, "DESYNC: self-organizing desynchronization and TDMA on wireless sensor networks," in *Proceedings of the 6th International Symposium on Information Processing in Sensor Networks (IPSN '07)*, pp. 11–20, April 2007.

[16] J. Degesys, I. Rose, A. Patel, and R. Nagpal, "Self-organizing desynchronization and TDMA on wireless sensor networks," in *Proceedings of the 1st Workshop on Bio-Inspired Design of Networks (BIOWIRE '07)*, pp. 192–203, Cambridge, UK, April 2007.

[17] A. Patel, J. Degesys, and R. Nagpal, "Desynchronization: the theory of self-organizing algorithms for round-robin scheduling," in *Proceedings of the 1st International Conference on Self-Adaptive and Self-Organizing Systems (SASO '07)*, pp. 87–96, July 2007.

[18] Y.-J. Kim, H.-H. Choi, and J.-R. Lee, "A bioinspired fair resource-allocation algorithm for TDMA-based distributed sensor networks for IoT," *International Journal of Distributed Sensor Networks*, vol. 2016, Article ID 7296359, 10 pages, 2016.

[19] F. Cucker and S. Smale, "Emergent behavior in flocks," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 852–862, 2007.