Ad Hoc Networks 25 (2015) 54–71

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

DICSA: Distributed and concurrent link scheduling algorithm for data gathering in wireless sensor networks

Behnam Dezfouli [a,b,*], Marjan Radi [a,b], Kamin Whitehouse [c], Shukor Abd Razak [a], Tan Hwee-Pink [b]

[a] Department of Computer Science, Faculty of Computing, Universiti Teknologi Malaysia (UTM), Johor 81310, Malaysia
[b] Networking Protocols Department, Institute for Infocomm Research (I^2R), A*STAR, Singapore 138632, Singapore
[c] Department of Computer Science, University of Virginia, Charlottesville, VA, USA

ARTICLE INFO

Article history:
Received 5 September 2013
Received in revised form 17 August 2014
Accepted 16 September 2014
Available online 27 September 2014

Keywords:
Scheduling algorithm
Interference
MAC

ABSTRACT

Although link scheduling has been used to improve the performance of data gathering applications, unfortunately, existing link scheduling algorithms are either centralized or they rely on specific assumptions that are not realistic in wireless sensor networks. In this paper, we propose a distributed and concurrent link scheduling algorithm, called DICSA, that requires no specific assumption regarding the underlying network. The operation of DICSA is managed through two algorithms: (i) Primary State Machine (PSM): Enables each node to perform its own slot reservation; (ii) Secondary State Machine (SSM): Enables each node to concurrently participate in the slot reservation of its neighbors. Through these algorithms and a set of forbidden slots managed by them, DICSA provides concurrent and collision-free slot reservation. Our results show that the execution duration and energy consumption of DICSA are at least 50% and 40% less than that of DRAND, respectively. In terms of slot assignment efficiency, while our results show higher spatial reuse over DRAND, the maximum slot number assigned by DICSA is at least 60% lower than VDEC. In data-gathering applications, our results confirm the higher performance of DICSA in terms of throughput, delivery ratio and packet delay. We show that the network throughput achievable by DICSA is more than 50%, 70%, 90% and 170% higher than that of DRAND, SEEDEX, NCR and FPS, respectively.

1. Introduction

The fundamental traffic pattern observable in sensor networks is to send the data sampled by the nodes towards a common destination called sink node. This many-to-one traffic pattern is referred to as *convergecast* [1–4]. In order to make accurate and quick decisions, data gathering applications usually require high delivery ratio with minimum end-to-end delay [5–8]. To this aim, data convergecast has been investigated and improved from various perspectives (e.g., tree structure [9,10], data aggregation [11,12], and channel access). From the channel access point of view, employing random access mechanisms results in significant packet collisions, which is the result of traffic direction, multihop transmissions and hidden-node problem. Since packet collisions reduce network performance in terms of effective throughput and delivery ratio [1,13], scheduling algorithms have been proposed to eliminate the negative effects of collisions on the performance of data gathering applications [14]. In particular, in contrast to random access mechanisms, scheduled access mechanisms (a.k.a., time division multiple access (TDMA)) divide

* Corresponding author at: Department of Computer Science, Faculty of Computing, Universiti Teknologi Malaysia (UTM), Johor 81310, Malaysia.
E-mail address: dezfouli@ieee.org (B. Dezfouli).

http://dx.doi.org/10.1016/j.adhoc.2014.09.011
1570-8705/

time into slots, and arbitrate channel access through slot assignment to the nodes or links. In *node scheduling algorithms*, transmission time slots are assigned to the nodes assuming each node's transmission should be received by all of its neighbors. In contrast, *link scheduling algorithms* aim to provide collision-free packet reception only at the intended receiver of each transmitter. Therefore, since link scheduling algorithms apply fewer constraints to slot assignment, they can potentially improve channel utilization, compared to node scheduling approaches [15,16]. Besides, as the convergecast traffic pattern indicates an almost static child-parent relationship between nodes [1,17], it justifies the benefits of employing link scheduling to improve data gathering performance [2,3,14]. Nevertheless, it has been shown that time slot assignment to the nodes or links of a graph is a NP-complete problem [18,19].

Based on the decision type made for channel access scheduling, the existing scheduling algorithms can be categorized into probabilistic (e.g., [20–23]) and deterministic algorithms (e.g., [2,3,19,24]). Using the probabilistic algorithms, nodes determine their behavior at each slot probabilistically (e.g., using a pseudo-random function). Therefore, although the main advantage of these algorithms is the ease of distributed implementation, they have no guarantee on a node's channel access delay. Additionally, since each node is only aware of its two-hop neighborhood, priority chaining may appear and these algorithms cannot effectively utilize the channel. For example, a node may refrain from transmission while no node in its two-hop neighborhood is transmitting. Using the deterministic algorithms, either the transmission slots of each node within the frame is known [24], or the total data gathering duration is predetermined [2,3]. However, while most of these algorithms are centralized (e.g., [25–29]), the rest (except DRAND [24]) rely on specific assumptions that are not realistic in sensor networks (e.g., the requirement to have an interference-free tree topology [30,31]). In particular, while significant research has been conducted on the theoretical aspects of improving network capacity through link scheduling, much less attention has been paid to the design of practical scheduling algorithms.

Among the deterministic algorithms, DRAND [24] is a distributed implementation of RAND [18], and is suitable for networks with no significant mobility. This algorithm requires no specific assumption regarding the network, and the complexity of its execution duration and message exchange is $O(\max N^{1,2})$ (assuming no packet loss), where $\max N^{1,2}$ is maximum two-hop neighborhood.[1] In addition, DRAND employs node scheduling because it does not consider any particular traffic pattern. However, considering the convergecast traffic pattern, this algorithm cannot achieve the potential improvements of link scheduling. For example, even if the transmissions of two neighboring nodes to their parents do not cause packet collision, DRAND prevents concurrent transmission of these nodes (exposed node problem). Furthermore, using DRAND, when a node is applying for a slot reservation, no one-hop or two-hop neighbor of this node is allowed to apply for slot reservation, therefore, only those nodes with distance more than two hops can be concurrently involved in slot reservation. This low level of concurrency increases the execution duration and energy consumption of DRAND.

In this paper, we propose the DIstributed and Concurrent link Scheduling Algorithm (DICSA), which provides distributed link scheduling without requiring any specific assumption regarding the underlying network. This algorithm relies on network layer information and performs slot assignment based on child-parent relationship. DICSA is composed of two algorithms: The first algorithm enables each node to perform its own slot reservation, the second one enables the nodes to be involved in the slot reservation of their neighbors. In particular, in contrast with DRAND (which requires at least three-hop distance between those nodes applying for reservation), DICSA does not require any specific distance for concurrent slot reservation, and enables the nodes to be involved in the slot reservation of more than one neighbor at a time. This mechanism significantly reduces the execution duration and energy consumption of DICSA compared to DRAND. Additionally, it results in lower slot updating cost during the network operation. Both of the DICSA's algorithms manage a common set of *forbidden slots lists* to achieve collision-free slot assignment. Using these lists, each node is also aware of the slots in which it should receive from its children or send to its parent. Therefore, establishing the forbidden slots lists also simplifies MAC design to achieve energy efficiency. Considering probabilistic and deterministic link scheduling and node scheduling algorithms, we perform comprehensive performance evaluations from four main perspectives: (i) algorithm execution cost, (ii) slot assignment efficiency, (iii) slot updating cost, and (iv) data gathering performance. All these evaluations confirm the high performance of DICSA.

It is worth mentioning that child-parent packet transmission is not the only traffic pattern in sensor networks. For example, link estimation and route updates may require packet exchanges that do not follow the child-parent scheme [17,32]. Nevertheless, the frequency of control packet transmissions is considerably lower than that of data packet transmissions; therefore implying the importance of collision-free unicast transmissions. Additionally, employing link scheduling algorithms in hybrid CSMA-TDMA MAC protocols enables collision-free unicast transmissions, as well as supporting other traffic types. Therefore, this paper does not aim to propose a sophisticated low-power MAC protocol, rather, the proposed scheduling algorithm can be used in the design of TDMA and hybrid MAC schemes. It should also be noted that this paper neglects data aggregation during convergecast; therefore, each generated packet should be individually delivered to the sink node. This type of convergecast is referred to as *raw-data convergecast* [3] and is different from the approaches proposed in [11,12,33].

The rest of this paper is organized as follows. Prior works are given in Section 2. We present the formal presentation and requirements of link scheduling and node scheduling algorithms in Section 3. The design and implementation of DICSA is described in Section 4. We present

[1] Notice that the set of the two-hop neighborhood of a sample node i, which is denoted by $N_i^{1,2}$, includes those neighbors that are one-hop or two-hop away from i. In other words, $N_i^{1,2} = N_i^1 \cup N_i^2$.